

# Co-PoeTryMe

Aplicação co-criativa de apoio à geração  
de poesia e letras de música

Tiago João Ribeiro Mendes  
2011144069



Mestrado em Design e Multimédia  
Faculdade de Ciências e Tecnologias  
Universidade de Coimbra

## Resumo

Nos últimos anos têm sido desenvolvidos vários sistemas que, de forma autônoma, geram poesia, através de diferentes técnicas, mas muitas vezes com limitações ao nível da intenção e significado. Por isso, uma abordagem alternativa à composição de poesia passa por tirar partido do computador para algumas tarefas, mas permitir também ao autor humano ter um papel no processo criativo.

O objetivo deste projeto é exatamente o de proporcionar aos utilizadores uma ferramenta interativa de apoio à composição de poesia e letras de música. Como forma de complemento, este documento pretende transmitir, num primeiro momento, o estudo realizado acerca das várias disciplinas que este trabalho engloba, e num segundo momento, clarificar todo o trabalho feito tanto ao nível do design da aplicação como da implementação propriamente dita.

Numa primeira fase é apresentado o estudo sobre as diversas áreas presentes no trabalho, que são a poesia, a geração de poesia, a criatividade computacional, a co-criatividade e o design de *interface*, de interação e gráfico. São também analisadas algumas aplicações relacionadas com a geração e co-geração de poesia.

Numa segunda fase é apresentado e analisado o PoeTryMe (Gonçalo Oliveira, 2012), enquanto sistema gerador de poesia e enquanto sistema criativo e co-criativo. É ainda apresentada a aplicação atualmente disponível, o TryMe.

Numa terceira etapa é apresentada a plataforma co-criativa desenvolvida, o Co-PoeTryMe, que tira partido das funcionalidades do PoeTryMe para geração de poemas, de versos e de palavras relacionadas com determinado termo ou com determinadas restrições silábicas ou fonéticas. É apresentado o levantamento de requisitos, tanto ao nível visual como do próprio funcionamento da aplicação, e são descritas e justificadas as escolhas feitas, tanto relativamente ao design como às tecnologias escolhidas para os cumprir.

O produto final desta dissertação é uma aplicação *Web* interativa de apoio à composição de poesia e letras de música, destinada não só aos praticantes dos domínios envolvidos, como para fins educativos ou lúdicos.

## Glossário

**API:** “*Interface de Programação de Aplicações*”, é um conjunto de funções estabelecidas por um *software* que podem ser utilizadas por outras aplicações.

**Anti-Aliasing:** Método de redução de *aliasing*, que é o efeito em forma de serra que se cria ao desenhar uma reta inclinada num computador – por isso também conhecido por serrilhamento.

**Browser:** Programa desenvolvido para permitir a navegação pela *Web*.

**Background:** “Ambiente de fundo”, neste caso aplicada à imagem, corresponde à parte da imagem que não representa o interveniente/sujeito principal.

**CSS:** “Folha de Estilos em Cascata”, é uma linguagem de marcação utilizada para estilizar os conteúdos de uma página *Web*.

**Drag & Drop:** “Arrastar e Largar”, é a ação de fazer click num objeto virtual e “arrastá-lo” para uma posição diferente ou para “cima” de um outro objeto virtual.

**Feedback:** Resposta ou reação do sistema a uma interação do utilizador.

**Função de Fitness:** Função que permite que o sistema auto-avalie o seu desempenho, e por consequência, os artefactos gerados.

**HTML:** “Linguagem de Marcação de Hipertexto”, consiste numa linguagem de marcação utilizada para produção de páginas *Web*.

**Hover:** Quando o rato está por cima de determinado elemento/objeto virtual.

**Interface:** Parte visual de um sistema que permite que o utilizador interaja com ele. É o elemento de ligação entre o utilizador e o sistema.

**JavaScript:** É uma linguagem de programação baseada em *scripts* amplamente utilizada no desenvolvimento de páginas *Web*.

**JSON:** Formato textual para a representação de dados através de listas e pares nome/valor, leve e fácil de ler por humanos e por máquinas.

**Landscape:** Termo que se refere à orientação de dispositivos móveis, neste caso *smartphones* e *tablets*, quando o aparelho está em posição horizontal.

**N-gramas:** Sequência contígua de n itens de uma determinada sequência de texto ou discurso.

**REST:** “Transferência de Estado Representacional”, é uma arquitectura que permite comunicar com serviços através de pedidos que são feitos através de URLs. Estes pedidos incluem a localização do serviço e um conjunto de parâmetros, e as respostas podem estar num formato textual, habitualmente XML ou JSON.

**Portrait:** Termo que se refere à orientação de dispositivos móveis, neste caso *smartphones* e *tablets*, quando o aparelho está em posição vertical.

**Screenshots:** Capturas de ecrã.

**Scripts:** Conjunto de instruções textuais para serem executadas por um programa de computador.

**Sentece case:** Caixa baixa. Refere-se à utilização de texto onde apenas é capitalizada a primeira letra depois de um ponto final, ou a primeira letra de nomes próprios.

**Software:** Programas de computador que comandam o seu funcionamento.

**Template:** Os templates são excertos de textos originais, de onde são retiradas algumas palavras, que são substituídas por outras no processo de geração.

**Testes de Turing:** Também designados por testes-cegos, são textos onde um conjunto de artefactos, alguns gerados por computador e outros produzidos por humanos, é apresentado a um conjunto de pessoas que não conhecem a sua origem. Pretende-se que as pessoas tentem identificar que artefactos foram gerados por computador e produzidos por humanos.

**Tomcat:** Servidor de aplicações *Web* Java.

**Tweets:** Publicações do Twitter.



**Testing:** Fase após a primeira implementação de uma aplicação, onde é pedido a um conjunto de pessoas que desempenhem determinadas tarefas. O processo é supervisionado, permitindo identificar alguns erros de usabilidade.

**Uppercase:** Caixa alta. Refere-se à utilização de texto apenas em letras maiúsculas.

**Versão Beta:** Aplicação que ainda está em estado de desenvolvimento e testes.

**Web:** “Teia” ou “Rede”, significa um sistema de informações ligadas através de hipermídia que permitem aceder a uma infinidade de conteúdos através da Internet.

# Índice

<b>1. Introdução</b>	<b>11</b>
1.1 - Âmbito	12
1.2 - Motivação	12
1.3 - Objectivos	13
1.4 - Contribuições	14
1.5 - Planeamento	15
<b>2. Estado da Arte</b>	<b>19</b>
2.1 Poesia	19
2.1.1 - Definição de Poesia	20
2.1.2 - Geração de Poesia	21
2.2 Criatividade e Co-Criatividade	23
2.2.1 - Criatividade	23
2.2.2 - Criatividade Computacional	25
2.2.3 - Co-Criatividade	27
2.3 Plataformas Relacionadas	31
2.3.1 - Sistemas autónomos de geração de poesia	32
2.3.2 - Plataformas de suporte criativo	34
2.3.3 - Sistemas co-criativos de geração de poesia	38
2.3.3 - Avaliação das plataformas analisadas	45
<b>3. Design de Aplicações interativas</b>	<b>47</b>
3.1 - <i>Interfaces</i> com o utilizador	47
3.2 - Design de Interação	51
3.3 - Usabilidade	52
<b>4. PoeTryMe: a base do Co-PoeTryMe</b>	<b>55</b>
4.1 - PoeTryMe como gerador de poesia	56
4.2 - PoeTryMe como sistema criativo	58
4.3 - PoeTryMe e a co-criatividade	59
4.4 - Limitações do PoeTryMe	59
<b>5. Implementação do Co-PoeTryMe</b>	<b>61</b>
5.1 - Requisitos Funcionais	61
5.2 - Escolhas tecnológicas	65
5.3 - Funcionalidades Implementadas	68

<b>6. Design da Aplicação</b>	<b>73</b>
6.1 - Requisitos Visuais	73
6.2 - Design da <i>Interface</i>	74
6.2.1 - Estrutura	74
6.2.2 - Identidade Visual	79
6.2.3 - Tipografia	81
6.2.4 - Paleta de Cores	84
6.2.5 - Ferramentas	87
6.2.6 - Fragmentos Poéticos	90
6.2.7 - Representação do processo co-criativo	92
6.3 - Design de Interação:	102
6.3.1 - Interação com as Ferramentas	102
6.3.2 - Interação com os Fragmentos Poéticos	107
6.4 Protótipos / Ecrãs	115
<b>7. Avaliação</b>	<b>126</b>
7.1 - Testes de Usabilidade	126
7.2 - Alterações feitas à Aplicação	131
7.3 - Utilização por terceiros	134
7.4 - Co-PoeTryMe e a co-criatividade	136
<b>8. Conclusão e Perspectivas Futuras</b>	<b>138</b>
<b>9. Anexos</b>	<b>140</b>
<b>10. Bibliografia/Webgrafia</b>	<b>145</b>

## Lista da Figuras

<b>Fig.1</b> - Plano seguido no primeiro semestre.	15
<b>Fig.2</b> - Plano seguido no segundo semestre.	17
<b>Fig.3</b> - <i>Interface</i> visual do FloWr, com o fluxograma de geração de poesia criado.	36
<b>Fig.4</b> - <i>Interface</i> visual do ConCreTe Flows, com o fluxograma do sistema discrito.	38
<b>Fig.5</b> - <i>Screenshot</i> da <i>Web-interface</i> do jGnoetry.	40
<b>Fig.6</b> - <i>Screenshot</i> do protótipo em “modo-escrita”, com a ferramenta “robot” aberta.	42
<b>Fig.7</b> - <i>Screenshot</i> da aplicação DeepBeat.	45
<b>Fig.8</b> - <i>Screenshot</i> da <i>interface</i> do TryMe.	56
<b>Fig.9</b> - Quadra gerada pelo PoeTryMe a partir das sementes “mar”, “praia” e “sol”, com o grau de surpresa igual a 0.	57
<b>Fig.10</b> - Soneto gerado pelo PoeTryMe a partir das sementes “amor” e “sentimento”, com o grau de surpresa igual a 0.5.	57
<b>Fig.11</b> - Exemplo de um pedido HTTP.	68
<b>Fig.12</b> - Módulo “Instruções” no estado reduzido, à esquerda e expandido, à direita.	76
<b>Fig.13</b> - Numa primeira fase com o módulo Instruções expandido e sem a possibilidade de fechar. Após a geração de conteúdos, disponibilização de módulos que suportam interação com esse mesmo conteúdos.	77
<b>Fig.14</b> - Arquitetura dos módulos.	78
<b>Fig.15</b> - Grelha na base da construção da <i>interface</i> do Co-PoeTryMe.	79
<b>Fig.16</b> - Logotipo desenvolvido.	80
<b>Fig.17</b> - Exemplos de merchandising com a marca Co-PoeTryMe.	81
<b>Fig.18</b> - <i>Speciemen</i> da fonte “Fabrik”.	82
<b>Fig.19</b> - <i>Speciemen</i> da fonte “Inconsolata”.	83
<b>Fig.20</b> - <i>Speciemen</i> da fonte “SchollBell”.	84
<b>Fig.21</b> - Interação com o botão Ativar/Desativar janelas de ajuda e o seu comportamento crômatico em relação à interatividade.	85
<b>Fig.22</b> - <i>Screenshot</i> da aplicação com a numeração dos botões.	87
<b>Fig.23</b> - Módulos magnéticos com palavras, colados no frigorífico.	91
<b>Fig.24</b> - Fragmentos poéticos gerados como peças possíveis de jogar.	92
<b>Fig.25</b> - Tipos de botões rádio disponíveis.	104
<b>Fig.26</b> - Janelas de ajuda nos <i>inputs</i> de texto.	105
<b>Fig.27</b> - Janelas de ajuda nos botões.	106

<b>Fig.28</b> – Janelas de feedback a situações de erro.	106
<b>Fig.29</b> – <i>Hover</i> verso e <i>hover</i> palavra do poema-rascunho.	108
<b>Fig.30</b> – <i>Hover</i> linha gerada e <i>hover</i> palavra gerada.	108
<b>Fig.31</b> – Verso clicado vs Palavra clicada.	109
<b>Fig.32</b> – Editar Linha/ Editar Palavra / Editar Texto.	110
<b>Fig.33</b> – <i>Drag</i> linha / <i>drag</i> palavra.	113
<b>Fig.34</b> – Adicionar linha / Adicionar palavra.	114
<b>Figs.35, 36, 37</b> – <i>Screenshots</i> da primeira versão da aplicação.	116
<b>Fig.38</b> – <i>Screenshot</i> da segunda versão da aplicação.	117
<b>Fig.39</b> – <i>Screenshot</i> da terceira versão da aplicação.	118
<b>Fig.40</b> – Ferramentas de seleção e edição utilizadas na terceira versão da aplicação.	118
<b>Fig.41</b> – <i>Screenshot</i> da quarta versão da aplicação – linha selecionada.	119
<b>Fig.42</b> – <i>Screenshot</i> da quarta versão da aplicação – palavra selecionada.	120
<b>Fig.43</b> – <i>Screenshot</i> da quinta versão da aplicação.	121
<b>Fig.44</b> – <i>Screenshot</i> da quinta versão da aplicação – método de seleção de fragmentos poéticos.	121
<b>Fig.45</b> – <i>Screenshot</i> da quinta versão da aplicação – método de seleção do intervalo de alterações ao texto a visualizar.	122
<b>Fig.46</b> – Screenshoots da última versão da aplicação.	124
<b>Fig.47</b> – Screenshoots da última versão da aplicação – representação dos conteúdos (neste caso versos) sugeridos pelo sistema.	124
<b>Fig.48</b> – <i>Screenshot</i> da última versão da aplicação – visualização da animação, neste caso, da troca da posição de duas palavras.	125
<b>Fig.49</b> – Conjunto de tarefas propostas aos utilizadores/testers.	127
<b>Fig.50</b> – Formulário preenchido aquando dos testes de usabilidade.	127
<b>Fig.51</b> – <i>Frame</i> de um dos vídeos dos testes de usabilidade efectuados.	128
<b>Fig.52</b> – Novo botão para alteração do tipo de fragmentação do poema.	131
<b>Fig.53</b> – Nova representação da ferramenta de adição de linhas.	133
<b>Fig.54</b> - Nova representação da ferramenta de adição de palavras.	133
<b>Fig.55</b> – Nova representação dos fragmentos poéticos sugeridos pelo sistema.	134
<b>Fig.56</b> – Pauta da canção criada, cuja letra foi criada no Co-PoeTryMe.	135

## **Lista da Tabelas**

<b>Tab. 1</b> - Avaliação das plataformas analisadas.	46
<b>Tab. 2</b> - Símbolos de revisão de provas, relacionados com a gramática e ortografia.	93
<b>Tab. 3</b> - Símbolos de revisão de provas, relacionados com a formatação do texto.	94
<b>Tab. 4</b> - Símbolos de revisão de provas, relacionados com a tipografia.	95
<b>Tab. 5</b> - Símbolos de revisão de provas, relacionados com a pontuação.	97
<b>Tab. 6</b> - Lista de utilizadores cujos testes foram presenciais.	130
<b>Tab. 7</b> - Erros presenciados nos testes de usabilidade.	130

# 1 Introdução

A poesia é uma forma de escrita criativa e, como tal, um processo complexo e, regra geral, solitário. A geração automática de poesia é uma tarefa no âmbito da geração de linguagem natural e é uma área de investigação da Inteligência Artificial que tem vindo a ganhar relevância nos últimos anos. Também a co-criatividade aplicada a este tipo de sistemas, onde o computador assume o papel de parceiro criativo, tem vindo a tornar-se objecto de estudo. Isto porque, entre outras razões, a capacidade de processamento dos computadores tem vindo a aumentar e, em consequência, tem havido um aumento da preocupação de tirar partido destas potencialidades para agilizar o processo criativo humano.

O PoeTryMe (Gonçalo Oliveira, 2012) é uma plataforma de geração automática de poesia que, através de uma gramática e um conjunto de relações semânticas entre palavras (que podem ser alteradas pelo utilizador), é capaz de gerar frases com significado. Para além disso, permite que o utilizador escolha uma forma poética, tema do poema, língua e ainda o “grau de surpresa” que corresponde à distância semântica entre as palavras usadas e o tema. Posto isto, e dada a versatilidade da arquitetura do sistema, é possível construir uma plataforma de geração de poesia que tenha por base o PoeTryMe. O sistema dispõe, desde há pouco tempo, de uma API REST, que permite o desenvolvimento de aplicações que comuniquem com o PoeTryMe, sob a forma de pedidos e respostas.

Actualmente existe uma aplicação disponível, o TryMe, que permite gerar poemas através da definição do conjunto inicial de parâmetros acima referidos, e da comunicação com o PoeTryMe, através da sua API REST. Podemos dizer que existe uma pequena componente co-criativa nesta aplicação, visto que o utilizador pode gerar poemas com as mesmas características (ou não)  $n$  vezes, no entanto, esta componente é muito limitada. Para além disso, muitas das vezes, os poemas gerados não vão de encontro às reais intenções do utilizador e este sente a necessidade de ter um papel mais ativo no processo de criação do poema.

Posto isto, surge a necessidade de criar uma plataforma co-criativa de geração de poesia, o Co-PoeTryMe, que utilize as potencialidades do PoeTryMe como parceiro criativo, e que tenha como principais objectivos a agilização do processo criativo humano e a adaptabilidade aos novos meios digitais. No âmbito desta tese foram desenvolvidas algumas funcionalidades extra para o PoeTryMe, por parte do autor do sistema Gonçalo Oliveira, para que a aplicação pudesse cum-

prir os requisitos estabelecidos. Isto porque, embora existam algumas plataformas de geração de poesia, não existe nenhuma que permita a qualquer um interagir com o resultado da geração de forma fácil e intuitiva, dando a possibilidade de moldar o poema à sua vontade. O resultado desta tese será uma aplicação que cumpra estes requisitos e, para isso, é necessário efectuar um estudo sobre os temas cruciais deste projecto: poesia, co-criatividade computacional e design de aplicações interativas.

Neste capítulo é descrita a origem do projecto, bem como os objectivos e metodologias necessárias ao seu desenvolvimento.

## 1.1 Âmbito

Este projecto enquadra-se no âmbito da dissertação de mestrado em Design e Multimédia da Universidade de Coimbra. Para a sua realização são cruciais as competências adquiridas durante o curso, bem como a investigação paralela sobre os temas vitais deste projecto: poesia, co-criatividade computacional e design de aplicações interativas. Este projecto vai ao encontro dos objectivos do curso de MDM, visto que engloba uma forte componente de design e de programação.

Na componente de Design destacam-se várias tarefas: desenvolvimento de uma identidade visual; desenvolvimento de uma aplicação interativa, que envolve o estudo sobre design de *interface*, design de interação e usabilidade.

Na componente de programação são necessárias as competências adquiridas em JavaScript, para programação do lado do cliente, bem como para a utilização de serviços *Web*, neste caso a API do PoeTryMe, e em HTML e CSS para a criação e estilização dos conteúdos.

## 1.2 Motivação

A escrita criativa é um processo moroso. Este projecto nasce da necessidade de existência de uma ferramenta capaz de agilizar o processo criativo, ou pelo menos resolver a “síndrome da folha branca” (Kantosalo, Toivanen, & Toivonen, 2015), isto é, a dificuldade que muitos escritores têm em começar um texto, mesmo sabendo sobre o que vão escrever.



A área da criatividade computacional têm-se debruçado nos últimos anos, entre outras coisas, sobre a geração de linguagem natural. Também a noção de co-criatividade, neste caso entre o computador e o ser humano, tem vindo a ser cada vez mais explorada, com o objectivo de tirar partido das capacidades do computador como parceiro criativo. Apesar de existirem algumas aplicações que utilizam o computador para esse fim, não foi identificada uma aplicação suficientemente capaz e intuitiva na área da composição de poesia e letras de música.

Para além disso, visto que cada vez mais as pessoas têm ligação à Internet e os computadores portáteis são mais comuns, há a necessidade que a ferramenta esteja on-line, para que o utilizador lhe possa aceder sem ter a necessidade de descarregar qualquer *software*, eliminando também todos os problemas de compatibilidade que poderiam existir.

O PoeTryMe é um sistema de geração de poesia automática, que está em desenvolvimento há alguns anos, e é capaz de gerar poemas de forma autónoma, através da definição de um conjunto inicial de parâmetros. Como é de esperar num sistema completamente autónomo, ele apresenta algumas lacunas, possíveis de colmatar através de uma intervenção humana mais ativa no processo de criação. Este foi um dos factos constatados por compositores humanos que recentemente mostraram interesse na utilização do sistema (Gonçalo Oliveira, 2012), destacando a importância de ser dada a possibilidade ao utilizador de alterar o poema gerado, através da interação com o sistema.

### 1.3 Objectivos

Existem poucas aplicações de geração de poesia e a maioria das existentes apenas permitem que o utilizador defina algumas regras iniciais, tais como a forma poética e o tema do texto. No entanto, muitas vezes o resultado apresentado não satisfaz as necessidades do utilizador. O objetivo principal deste projecto é suprir essa lacuna, através do desenvolvimento de uma aplicação que permita ao utilizador, para além de definir as regras iniciais de geração, interagir com o texto gerado, possibilitando assim que o molde à sua vontade, com um alto nível de personalização. Esta é uma das principais lacunas que as poucas plataformas de co-criação de poesia existentes possuem: baixo nível de personalização. O projecto tem ainda o objectivo implícito de resolver o “síndrome da folha branca” (Kantosalo et al., 2015) e agilizar o processo criativo do utilizador.

A par disto é pretendido que os vários tipos de interação com o texto sejam fáceis de entender e utilizar. Isto é, pretende-se que a aplicação seja intuitiva e que a experiência de navegação seja positiva. Isto passa por, por exemplo, evitar situações em que o resultado da utilização de uma ferramenta não corresponde ao previsto pelo utilizador, situações estas que podem conduzi-lo à frustração.

Outro objectivo é que a aplicação seja portátil e se adapte aos vários tipos de ecrã de computador. O facto da aplicação ser acedida através de um *browser* faz com que seja possível usá-la em qualquer lado, desde que haja conexão à Internet. Embora nesta fase a aplicação apenas esteja disponível para computador pessoal, a sua *interface* deve moldar-se às características do ecrã do computador que lhe está a aceder, bem como às características do *browser* utilizado.

Para além disso, pretende-se aproveitar esta oportunidade para criar uma identidade visual para a aplicação e para o projecto de onde ela deriva, o PoeTryMe, que os permita identificar de forma clara.

Um objectivo secundário será o de tornar a API do PoeTryMe mais completa e flexível para os utilizadores deste serviço. Isto passa pelo levantamento de requisitos para a nova aplicação, Co-PoeTryMe, que necessita do desenvolvimento de funcionalidades não disponíveis anteriormente.

## 1.4 Contribuições

A principal contribuição deste projecto é uma aplicação de apoio à composição de poesia e letras de música. O sistema permite que o utilizador personalize o poema tanto numa fase inicial, onde restringe a geração à forma poética e tema escolhidos, como numa fase posterior, interagindo com o poema-rascunho gerado. A aplicação pretende fornecer ao utilizador um alto nível de personalização do resultado. Paralelamente, a aplicação deve fornecer uma interação intuitiva, que permita ao utilizador focar-se nas tarefas a desempenhar, ao invés de ter que se preocupar com a aprendizagem do sistema.

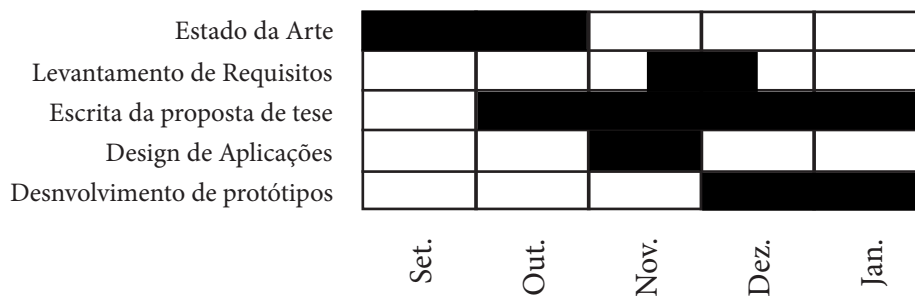
A aplicação será, por conveniência, acedida através de um *browser*, visto que deverá permitir ao utilizador que use o sistema sem ter necessidade de descarregar e instalar nenhum *software* adicional, estando assegurada a compatibilidade com todos os dispositivos. Esta solução permite que o utilizador use a aplicação

apenas tendo acesso à Internet, o que é cada vez mais comum nos dias de hoje. A aplicação é ainda complementada com esta tese que inclui um estudo acerca dos vários temas que este projecto aborda: poesia, criatividade computacional, co-criatividade, design de *interface* e interação, bem como um levantamento de plataformas existentes tanto na área de geração de poesia como da co-criatividade ligada à geração de poesia.

## 1.5 Planeamento

Visto que este é um projecto interdisciplinar, a fase de desenvolvimento passou pelo estudo das áreas da Poesia da Criatividade Computacional, da Co-Criatividade e do Design de Aplicações. Esta fase envolveu ainda a familiarização com o funcionamento do PoeTryMe e sua API.

No primeiro semestre foi dada prioridade à escrita da proposta de tese, e para tal foi necessária a elaboração de um estudo acerca dos principais temas que o projecto aborda. Para além disso, e visto que o projecto consiste numa aplicação *Web*, foi necessário fazer um levantamento de requisitos que permitisse perceber que acções o utilizador poderá executar, bem como, numa fase posterior, como é que essas ferramentas lhe serão apresentadas. Isto permite elaborar os primeiros esboços da aplicação, através da disposição das várias ferramentas e, posteriormente, estudar a maneira como estas são usadas. Envolve ainda o estudo de cada interação necessária para que o utilizador use cada funcionalidade, para que a utilização das ferramentas seja intuitiva e natural. Na Figura 1 está representada a distribuição temporal das várias fases do projecto, a desenvolver no primeiro semestre.



**Fig.1-** Plano seguido no primeiro semestre.

As actividades representadas na Figura 1 são agora explicadas em mais detalhe:

**Estado da Arte:** Esta fase envolve o estudo de diferentes áreas:

- **Poesia:** Estudo da poesia no contexto da geração de poesia artificial, através de sistemas computacionais criativos. É feito um levantamento de métricas que permitem avaliar tanto os processos que levam à geração dos textos, enquanto condutores ou não condutores à geração de poesia, como dos textos gerados, enquanto poesia ou não-poesia.
- **Criatividade Computacional:** Estudo da criatividade enquanto objecto de estudo da Inteligência Artificial. Este estudo envolve um levantamento de métricas que permitem avaliar sistemas computacionais como criativos ou não-criativos, através da análise tanto dos processos de geração dos textos, como dos artefactos gerados.
- **Co-Criatividade:** Estudo da co-criatividade, em situações que o computador é utilizado como parceiro criativo. É feito um levantamento de possíveis papéis que o computador pode desempenhar enquanto parceiro criativo e é avaliado o grau de criatividade e intervenção do sistema no processo de criação.
- **Geração Computacional de Poesia:** Estudo de vários sistemas, com foco no PoeTryMe por ser a base do sistema a desenvolver. Esta avaliação é feita através da aplicação das métricas apresentadas anteriormente: avaliação dos processos que levam à geração dos textos, e dos textos propriamente ditos enquanto poesia ou não-poesia; avaliação tanto dos processos usados pelo sistema para gerar artefactos, como dos artefactos em si, enquanto criativos ou não; avaliação do Co-PoeTryMe ao nível do papel que o computador assume no processo criativo. Para tal ser possível, é necessário um levantamento de todas as possibilidades de interação com a aplicação (levantamento de requisitos) e avaliar o nível de criatividade e de intervenção do sistema no processo criativo.

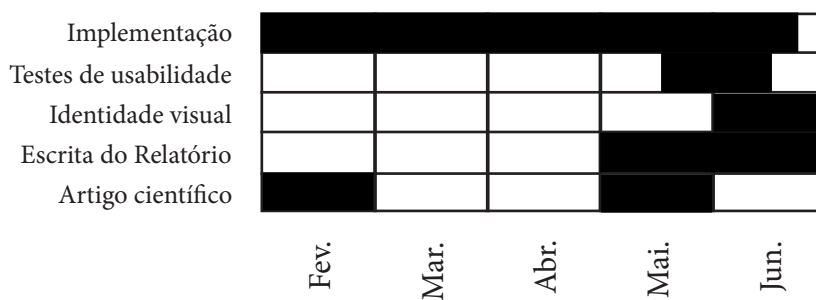
**Design de Aplicações:** Levantamento de regras e diretrizes a considerar no âmbito do desenvolvimento de aplicações interativas. Este estudo foca-se em primeiro lugar no design de *interfaces*, isto é, do design dos cenários/ecrãs estáticos, e posteriormente no design de interação, que estuda a forma como o utilizador interage com as diferentes componentes, bem como o comportamento destas

componentes quando sujeitas aos vários tipos de interação possíveis. Sendo o texto o principal agente da aplicação, foi feito um estudo de regras de utilização da tipografia em meios digitais.

**Levantamento de Requisitos:** Levantamento de todas as possibilidades de interação do utilizador com o sistema. Isto envolve identificar todas as tarefas que a plataforma deve permitir desempenhar.

**Protótipos:** Estudo da disposição e apresentação das funcionalidades ao utilizador. Em primeiro lugar foram feitos protótipos de baixa-fidelidade, que permitem definir que conteúdos devem estar presentes em cada ecrã/cenário, bem como a disposição desses conteúdos no espaço. Segue-se o desenvolvimento de um protótipo funcional, que permite avaliar a interação propriamente dita com mais detalhe. No âmbito do desenvolvimento de um protótipo funcional são levadas em conta as diretrizes resultantes não só do estudo de design de *interface*, mas também de design de interação.

No segundo semestre, foi dada prioridade à implementação da aplicação. Esta fase envolve um conjunto de outras tarefas: a implementação, a elaboração de testes de usabilidade e desenvolvimento de uma identidade visual. Na Figura 2 está representada a distribuição temporal das diversas fases pelas quais o projecto passou, ao longo do segundo semestre. Esta planificação sofreu algumas alterações relativamente à planificação do segundo semestre apresentada no relatório intermédio, nomeadamente nas fases de testes de usabilidade e início da escrita, que acabaram por começar mais tarde do que o que se previa inicialmente.



**Fig.2-** Plano seguido no segundo semestre.

**Implementação:** Desenvolvimento da aplicação levando em conta os requisitos definidos e utilizando as tecnologias apresentadas e descritas acima. A fase da implementação pode ser dividida, bruscamente, em três fases:

- Desenvolver do *layout* da aplicação de acordo com os protótipos realizados;
- Programar a aplicação de maneira a que o layout se desdobre e ajuste às características do ecrã do computador que está a aceder à aplicação;
- Tornar a aplicação funcional através da comunicação com o PoeTryMe.

A fase de implementação engloba ainda os testes de usabilidade, o que permite a “re-implementação” de algumas funcionalidades ou módulos que não estejam tão visíveis ou cuja utilização não seja tão intuitiva.

**Testes de Usabilidade:** Disponibilização de uma versão beta da aplicação, a ser usada por um conjunto de utilizadores, num ambiente supervisionado. É-lhes dado um conjunto de tarefas a desenvolver e analisada a maneira como eles interagem com o sistema. São recolhidos dados relativos às grandezas levadas em conta nas métricas apresentadas para “medição” da usabilidade.

**Identidade Visual:** Definição da marca e conceito da aplicação, que envolve desenvolvimento do logo, restrições da sua utilização, escolhas ao nível tipográfico e cromático, teste de escalas e desdobramento ao longo da aplicação.

**Artigo Científico:** Colaboração na escrita de dois artigos científicos, nomeadamente para o ProSocrates 2017, em Fevereiro, e em Maio de um artigo demonstrativo da aplicação.

## 2 Estado da Arte

Este capítulo apresenta um estudo dos principais temas do projecto, e está dividido em várias secções: a primeira que se prende com o estudo e definição de poesia; a segunda que aborda a criatividade computacional enquanto ramo da Inteligência Artificial; a terceira que se debruça sobre a co-criatividade, neste caso quando o computador é utilizado como parceiro criativo e, finalmente, a quarta que analisa e avalia algumas plataformas relacionadas.

Quanto à poesia, numa primeira fase, vai ser definida em termos mais formais e abrangentes, definição essa que numa segunda fase vai sendo restringida, de maneira a se adequar melhor ao contexto do projecto. São ainda identificadas algumas abordagens automáticas para a geração de poesia, bem como um conjunto de métricas que permitem avaliar tanto os processos de geração de texto, enquanto produtores ou não de poesia, como dos artefactos gerados enquanto poesia ou não poesia.

Na segunda secção, é definido o conceito de criatividade computacional enquanto objecto de estudo da Inteligência Artificial. É feito um levantamento de métricas que permitem avaliar tanto a presença de criatividade nos processos geradores de textos como nos textos gerados.

De seguida são analisadas abordagens à co-criação de poesia e avaliadas de acordo com o grau de criatividade necessário por parte do sistema e o seu grau de intervenção no processo criativo.

Na última secção é feito um levantamento de plataformas relacionadas, entre elas sistemas autónomos de geração de poesia, plataformas de suporte criativo e plataformas co-criativas de geração de poesia, que são avaliadas de acordo com as métricas apresentadas.

### 2.1 Poesia

A definição de poesia é muito ambígua e é difícil encontrar uma definição correta e que agrade a toda a gente. Fabb (2015), define poesia como “um texto composto por linguagem e dividido em secções que não são determinadas por características sintáticas ou pela estrutura prosódica”. Na definição, estas secções identificam

linhas, pares de versos, estrofes, que embora possam apresentar características da prosa ou do diálogo comum, não são determinadas por estes aspetos.

No entanto, esta definição tende a olhar para a poesia de uma forma mais filosófica, tornando-a muito abrangente. Engloba abordagens à poesia mais experimentais, tais como o verso livre, que não exige necessariamente a presença de rima nem o cumprimento de uma métrica, bem como a poesia concreta, que se foca no aspeto visual do poema, isto é, na forma da mancha gráfica.

No entanto, no âmbito deste projecto, é necessário uma definição mais científica, que permita avaliar, identificar e distinguir, com mais rigor, a poesia da não-poesia.

### 2.1.1 Definição de Poesia

Na busca de uma definição mais virada para o estudo científico da poesia, há várias abordagens que se apresentam sob a forma de requisitos que o texto deve cumprir para ser considerado deste estilo literário. Uma das mais populares e consensuais é a de Manurung (2003), que apresenta a definição de poesia que mais se adequa ao contexto deste projecto, e que permite avaliar os textos gerados diretamente. Segundo a sua definição, um texto é um poema quando apresenta três características inerentes a este estilo literário:

**Poeticidade:** A presença de propriedades que caracterizam este estilo literário, tais como ritmo, métrica, padrões fonéticos e linguagem figurativa. O ritmo e a métrica estão relacionados com padrões de acentuação silábica, isto é, com a posição de sílabas acentuadas e não acentuadas nos versos. Os padrões fonéticos englobam a rima, aliteração e assonância:

- **Rima:** ocorre entre duas palavras, quando as vogais acentuadas e os fonemas que as seguem são iguais.
- **Aliteração:** consiste na repetição de sons de consoantes em palavras consecutivas dos versos de um poema.
- **Assonância:** consiste na repetição de sons de vogais em palavras consecutivas dos versos de um poema.



A linguagem figurativa é uma característica comum de vários géneros literários, não sendo exclusivo da poesia. Consiste no jogo de palavras usado por parte do autor para enriquecer a experiência de interpretação do poema. Este é um dos aspetos mais difíceis de validar, especialmente se só analisarmos os textos produzidos, porque não sabemos quando uma figura de estilo é uma metáfora ou uma sequência de palavras aleatória mal conseguida.

Em comparação com a definição de Nigel Fabb (2015), este primeiro requisito da definição de Manurung exclui o verso livre da poesia, pois não tem qualquer preocupação rítmica ou métrica.

**Gramaticalidade:** Um poema deve seguir regras sintáticas, definidas por uma gramática e dicionário. Este é talvez o requisito mais óbvio que, por definição, todos os textos gerados por sistemas de geração de linguagem natural devem cumprir. Embora a noção de gramaticalidade na poesia seja menos restrita do que na prosa, a poesia não deixa de ser regrada, evitando simples sequências de palavras aleatórias.

**Significado:** Um poema deve transmitir uma mensagem conceptual possível de ser interpretada. Esta é uma restrição que não se aplica apenas aos textos deste género literário, mas a todos os tipos de texto.

Esta definição é mais virada para a avaliação de sistemas de geração de linguagem natural. Estes sistemas, cujo objectivo é a produção de texto possível de ser interpretado pelo ser humano, não têm a mínima preocupação com a forma da mancha gráfica. Por consequência, a poesia concreta não se enquadra nesta definição, nem no trabalho desta tese.

## 2.1.2 Geração de poesia

A geração automática de poesia é um tema muito popular no seio da comunidade da Criatividade Computacional. Vários sistemas que geram poesia de forma autónoma são propostos todos os anos, explorando diferentes recursos e abordagens, tais como *case-based reasoning* (Gervás, 2001), algoritmos evolucionários (Manurung 2003), *constraint programming* (Toivanen et al. 2013), sistemas multi-agente (Misztal and Indurkha, 2014), sumarização generativa (Yan et al, 2013), etc.

No entanto, e mais importante do que olhar para a estratégia computacional de geração de artefactos propriamente dito, é olhar para os tipos de restrições impostas ao processo de geração de poesia para que o output cumpra os requisitos e apresente as características deste estilo literário. Isto é, verificar se estão a ser impostas regras que garantam tanto o cumprimento de determinada métrica, como a correção gramatical do poema gerado, e ainda, se obrigam o sistema, de alguma maneira, a conferir um significado ao poema possível de ser interpretado por um ser humano. Isto porque, para além de uma ferramenta de geração de poesia poder combinar várias estratégias de geração das acima enumeradas, o mais importante quando analisamos um texto gerado por um sistema de geração de poesia, é perceber se esse texto apresenta as características do estilo literário e se, por consequência, pode ser considerado poesia. Uma categorização que vai de encontro a este tipo de avaliação da poesia é a de Manurung (2003), e distingue as abordagens de geração de poesia em três grupos, da seguinte forma:

**Salada de palavras:** Tentativas de geração de poesia que se limitam a criar sequências de palavras aleatórias, o que resulta num texto sem sentido e sem preocupações em termos gramaticais ou métricos.

**Preocupado com a forma:** Nesta abordagem, a escolha das palavras é regida pelo cumprimento de regras de métrica e gramaticalidade, salvaguardando o seu cumprimento. No entanto, ao nível do significado, os poemas gerados por esta estratégia podem revelar algumas fragilidades.

**Sistemas de geração de poesia:** A escolha das palavras é regida por regras gramaticais e de métrica e, para além disso, devem fazer sentido e transmitir uma mensagem conceptual. Os versos devem também cumprir restrições impostas pela forma da estrofe, nomeadamente no que diz respeito à presença de rima.

Segundo esta avaliação, apenas os resultados da geração dos sistemas que se enquadram na última categoria são capazes de produzir poesia, visto que são os únicos que, durante o processo de geração, impõem regras tanto ao nível do cumprimento da métrica, como gramatical, e ainda na escolha das palavras para conferir determinado significado ao poema.

## 2.2 Criatividade e co-criatividade

O termo co-criatividade computacional é um termo composto, e por isso é necessário analisar os seus vários constituintes.

Em primeiro lugar é necessário definir criatividade no âmbito do projecto, através da definição de uma métrica que permita identificar quando é que um artefacto gerado por um programa de computador pode ser considerado criativo ou não. Neste caso, como estamos a avaliar sistemas de geração de artefactos – e o computador é visto como o criador dos artefactos – estamos a verificar a presença, ou não, de criatividade computacional. Esta avaliação deve tomar em consideração não só os textos gerados, mas também os processos criativos que levam à geração dos artefactos.

É também necessário definir co-criatividade, que como o termo indica é um processo criativo com mais do que um interveniente. Como estamos a admitir o computador como parceiro criativo, na prática estamos a avaliar o grau de intervenção do computador no processo criativo.

Posto isto, é necessária a definição de uma métrica que permita avaliar tanto a presença de criatividade no sistema, como a capacidade do sistema funcionar como parceiro criativo, questões essas que vão ser abordadas nas secções seguintes.

### 2.2.1 Criatividade

A definição de criatividade é muito ambígua e, tal como para a poesia, não é possível estabelecer uma definição formal que satisfaça toda a gente. Rhodes (1961), numa fase inicial, e posteriormente Runco (2007), tentaram decompor o conceito de criatividade em factores que permitem avaliar a sua presença. Esta abordagem é chamada de “Os quatro P’s da Criatividade” e discrimina os factores que estão presentes no âmbito da criação de artefactos criativos:

**Pessoa** (person): características do indivíduo criador do artefacto;

**Produto** (product): características do produto que o tornam criativo;

**Contexto** (press): factores contextuais, ambientais e sociais;

**Processo** (process): técnicas de pensamento e processo cognitivo.

No entanto, esta definição serve melhor o estudo filosófico do termo “criatividade” do que o seu estudo científico. Para além disso, leva em conta características da pessoa na origem da criação de artefactos criativos.

Uma definição alternativa (Newell, Shaw, & Simon, 1962), mais direccionada para o estudo científico da criatividade, estipula quatro requisitos que o artefacto gerado deve apresentar para que seja atribuída criatividade ao sistema gerador:

A resposta deve ter **novidade e utilidade no domínio que representa**, tanto para o indivíduo como para a sociedade;

A resposta exige a **rejeição de ideias previamente aceites**;

A resposta resulta de **persistência e motivação intensa**;

A resposta **vem esclarecer um problema originalmente vago**.

Este tipo de abordagem é típica da Inteligência Artificial, em que o resultado pretende responder uma pergunta, desvalorizando o processo que leva à resposta. Esta métrica é mais virada para a avaliação dos artefactos gerados, no entanto a maioria dos sistemas foca-se apenas no primeiro requisito e apenas é capaz de criar respostas novas e com utilidade individual e social. Ainda assim, estes requisitos de novidade e utilidade são importantes na avaliação de artefactos gerados, e vão ser abordados mais à frente.

No entanto, ambas as definições são difíceis de aplicar na avaliação de sistemas computacionais. A definição de Wiggins (2006) é a mais interessante para o estudo da criatividade no ambiente da criatividade computacional. Wiggins define criatividade como uma procura e tenta formalizar conceitos da hierarquia da criatividade de Margaret Boden (1992) (Liapis, Yannakakis, 2016). Aplicando estes conceitos a um sistema de geração de poesia podemos dizer que o termo universo  $U$  refere-se ao espaço de todos os poemas possíveis de gerar. No entanto, é imposto um conjunto de regras  $R$  (que habitualmente se prende com a definição da forma poética e tema desejado, por parte do utilizador) que reduz o espaço de procura, visto que todos os artefactos de  $U$  que não cumprem as regras  $R$  são descartados. Um conjunto de regras  $T$  define como o sistema opera o espaço de

procura (técnicas de geração de artefactos) e outro conjunto de regras E avalia os artefactos produzidos, vulgarmente através de uma função *fitness*, que pode ser mais ou menos completa – por exemplo no caso do PoeTryMe, sistema de geração de poesia criado por Gonçalo Oliveira (2012), a função de fitness apenas pontua o cumprimento da métrica e a ocorrência de rima, ficando ao cargo do utilizador a avaliação do significado.

Este modelo permite assim olhar para o processo co-criativo entre o humano e o computador e estudar a interação no processo (Kantosalo, Toivonen, 2016) o que é importante no contexto deste trabalho, e vai ser discutido mais à frente.

## 2.2.2 Criatividade Computacional

A criatividade computacional é uma área multidisciplinar que sobrepõe aspectos da Inteligência Artificial a muitos outros domínios, tais como artes visuais, música, matemática, etc. Este é um campo de investigação relativamente recente, visto que apenas nas últimas décadas se começou a explorar o potencial de sistemas computacionais que exibem comportamentos criativos.

Embora a criatividade computacional seja um ramo da Inteligência Artificial, está assente num paradigma diferente, visto que pretende, através da execução de tarefas inteligentes, produzir artefactos com valor cultural (Colton, Wiggins, 2012). Os sistemas criados pretendem “copiar” a criatividade humana, através da formulação de algoritmos que reproduzam comportamentos criativos humanos, ou melhorar a criatividade humana, sem necessariamente haver criatividade por parte do sistema.

A criatividade, durante muitos anos, não foi considerada um ramo de pesquisa da Inteligência Artificial, visto que existe um certo cepticismo na atribuição de criatividade a um programa de computador. Inclusivamente, muitos autores defendem que um programa de computador apenas tem a capacidade de “seguir instruções específicas fornecidas pelo programador”, e para eliminar esse cepticismo seria necessário que o *software* fosse capaz de escrever as suas próprias instruções (Charnley, Colton, & Llano, 2016). No entanto, outros autores defendem que para ser atribuída criatividade a um programa de computador, basta que este “exiba comportamentos que para observadores imparciais seriam considerados criativos” (Colton, Wiggins 2012). Esta técnica de comparação de resultados

obtidos através de sistemas computacionais com resultados obtidos por pessoas (testes de Turing), é bastante comum em Inteligência Artificial. Nestas duas últimas abordagens à criatividade computacional, é dada importância não só aos artefactos gerados, mas também ao processo criativo, e segundo Maher (2012) a criatividade computacional pode ser descrita através da identificação dos processos criativos e a forma como o processo altera o espaço conceptual que opera”. Esta preocupação com o processo criativo também está presente na avaliação da arte moderna, onde o artefacto é avaliado não só pelo valor do produto final, mas também pelo processo, que tem um valor comparável ao do artefacto.

Vários autores, como Maher (2012) e Dan Ventura (2016), defendem a criação de uma métrica que permita avaliar a presença de criatividade sem levar em conta a sua origem (computacional ou humana), e que permita medir o progresso de comportamentos criativos, que é uma das maiores dificuldades nesta área.

Boden (2003), Gero (2000) e Maher (2012) defendem que é possível avaliar a presença de criatividade pela análise do processo criativo do sistema computacional. Segundo estes autores, um processo de geração de artefactos considerado criativo está assente num dos seguintes processos:

**Combinação** de conceitos presentes em artefactos já existentes;

**Exploração** de conceitos, num determinado domínio, nunca explorados nos artefactos existentes;

**Transformação** de conceitos presentes em artefactos de um determinado domínio, alargando as fronteiras desse mesmo domínio;

**Analogia**, que consiste na transferência de conceitos de outros domínios para a criação de um artefacto num determinado domínio.

No entanto, outros autores defendem que a simples avaliação do processo criativo usado no âmbito da geração de artefactos não é suficiente, e por isso defendem a criação de uma métrica que permita identificar a presença de criatividade computacional através da análise dos artefactos gerados (Kantosalo, Toivanen, Xiao, & Toivonen, 2010) (Ventura, 2016) (2007) (Gervás, 2002). Estas métricas estão assentes em três factores:

**Novidade**, que avalia a originalidade do artefacto gerado em relação à população de artefactos do mesmo domínio;

**Valor**, que avalia se o artefacto tem importância e utilidade para os praticantes do domínio;

**Intencionalidade**, que avalia se o artefacto é resultado de um objectivo do sistema, isto é, avalia se o artefacto é deliberado.

Este método de identificar a presença de criatividade computacional pretende distinguir a geração de conteúdos criativos, da simples geração aleatória e sem intencionalidade (em inglês, chamada Mere Generation), e para que isso seja possível, é necessário um método de auto-avaliação dos artefactos gerados.

Amabile (Maher, 2012) argumenta que não há um critério específico que conceptualmente fundamente a avaliação. Ela defendeu e introduziu um método onde a avaliação é feita por um conjunto de júris com conhecimento do domínio operado pelo sistema computacional.

Embora não haja um critério suficientemente claro que distinga um artefacto criativo de um não criativo, acredito que a avaliação humana dos resultados tende a ser imparcial (principalmente nos últimos anos, onde a criatividade se tem afirmado como ramo da Inteligência Artificial). Critérios como os de Dan Ventura (2016) e de Maher (2012), assentes em factores como na novidade, no valor, e na intencionalidade, são, na minha opinião, os mais próximos do ideal para avaliar a presença de criatividade em artefactos.

### 2.2.3 Co-Criatividade

A co-criatividade computacional é um sub-campo da criatividade computacional que utiliza as potencialidades do computador como parceiro do processo criativo humano. Isto é, a co-criatividade (Kantosalo & Toivonen, 2016) entre um ser humano e um computador é uma forma colaborativa de criatividade entre estes, onde ambos tem responsabilidade criativa (Kantosalo et al., 2010). Posto isto, é crucial estudar o papel e o grau de intervenção que o computador pode ter no processo criativo.

Num sistema co-criativo de geração de artefactos, o agente computacional pode desempenhar vários papéis. Já foram feitas várias tentativas no sentido de distinguir os possíveis papéis que o sistema pode desempenhar. Numa fase inicial, vão ser analisadas duas abordagens de categorização de papéis desempenháveis pelo sistema :a de Lubart (2005) e a de Maher (2012). Lubart (2005) identifica quatro papéis possíveis para o sistema desempenhar:

O sistema **gere o trabalho e tempo gasto em atividades criativas**, e apenas desempenha tarefas rotineiras, tais como gravar e mostrar informação;

O sistema **facilita a circulação de informação** entre o artista e a audiência ou entre outros co-autores;

O sistema **toma o papel de treinador** e aconselha o utilizador a utilizar técnicas/ferramentas que simulam o processo criativo;

O sistema **funciona como um colega que pode ser criativo**, contribuindo com ideias no diálogo com os utilizadores.

No entanto, à luz desta categorização, nas primeiras três categorias não é necessário que o sistema seja criativo para que possa desempenhar a sua função. Apenas a última categoria exige criatividade por parte do computador. Mary Maher (2012), define três papéis possíveis para o sistema desempenhar:

O sistema **proporciona ferramentas de suporte criativo**;

O sistema **potencia a capacidade criativa do utilizador** por disponibilizar conhecimento ou incentivar à criação cognitiva;

O sistema **gera ideias que são avaliadas e podem ser integradas**, ou não, no artefacto final.

Também nesta abordagem, as duas primeiras tarefas não exigem comportamento criativo por parte do agente computacional. No entanto, ambas as abordagens tendem a olhar para o computador como um instrumento/ferramenta interativa com potencial criativo para o utilizador dominar. Por isso o conhecimento base do domínio por parte do utilizador conduz, por norma, a melhores resultados. No entanto, Maher (2012) defende que tanto os humanos como os agentes computa-



cionais devem agir como indivíduos participantes no processo de criação. Kantosaló e Toivonen (2016) definem dois tipos de co-criatividade:

**Co-criatividade Alternada:** a pessoa e o agente computacional tomam turnos de criação que resultam num único conceito;

**Co-criatividade dividida em tarefas:** o utilizador e o agente computacional só desempenham tarefas específicas. Assim sendo os agentes não assumem turnos iguais.

Na **co-criatividade alternada**, o objectivo é produzir um artefacto que agrade ambas as partes através de turnos de iteração (avaliação e validação) a um conjunto de artefactos inicial. Este processo restringe o conjunto de artefactos iniciais, e dá origem um conjunto de artefactos que satisfaça ambos os agentes (humano e computacional). Podem-se distinguir dois tipos de co-criatividade alternada, e a diferença está no comportamento quando a intersecção entre os conjuntos de conceitos iniciais dos dois agentes (humano e computacional) é o conjunto vazio. A co-criatividade alternada diz-se simétrica quando o agente computacional é capaz de adaptar o seu conjunto de regras inicial de maneira a combater as incompatibilidades. Diz-se assimétrica quando o agente computacional descarta os seus turnos de iteração.

Na **co-criatividade dividida em tarefas** o processo de criação de um artefacto é dividido em sub-tarefas (definir conceitos, gerar conteúdos e avaliar conteúdos). O computador e a pessoa desempenham sub-tarefas específicas, assumindo assim turnos desiguais. No entanto, segundo a definição de co-criatividade de Yannakakis (2014), não é necessário que os dois agentes tenham o mesmo grau de envolvimento, desde que haja iniciativa criativa de ambas as partes. Também neste tipo de co-criatividade se distinguem duas categorias diferentes: o sistema usa algoritmos que lhe permitem pesquisar o espaço de conceitos interessantes possíveis e é capaz de os avaliar — agente completo; o sistema é incapaz de definir, gerar ou avaliar os seus próprios conceitos — agente incompleto.

A principal diferença entre estas duas abordagens à co-criatividade é o facto da co-criatividade alternada exigir mais do agente computacional. Para um sistema suportar co-criatividade alternada tem que ser completo, e por isso ser capaz de desempenhar tarefas mais complexas que os agentes incompletos, tais como ser capaz de identificar, gerar e avaliar conceitos num determinado domínio.

Kantosalo e Toivonen (2016) identificam um conjunto de potenciais problemas que surgem no âmbito da criação de plataformas de co-criatividade alternada. Estes desafios apresentam semelhanças com os conceitos de “aberração” e “desinspiração” introduzidos por Wiggins (2006). O conceito de “aberração” identifica situações em que o sistema gera conceitos fora do seu espaço de artefactos possíveis e válidos. O conceito de “desinspiração” pretende descrever situações onde o sistema é forçado a operar áreas do espaço de procura que não têm valor para ele, com objectivo de produzir resultados que agradem ao utilizador, ou quando o agente computacional gera constantemente conceitos que não têm valor para o utilizador. Os problemas identificados por Anna Kantosalo e Hannu Toivonen são:

**Incompatibilidade Universal:** O utilizador insere um conceito que não pertence ao universo de ação do agente computacional ou o conceito gerado pelo computador está fora do universo do utilizador. Por exemplo, no âmbito da geração de poesia, quando o utilizador sugere ao sistema um poema com determinada aparência visual, mas o sistema apenas se rege por regras de semântica, gramaticalidade e métrica.

**Incompatibilidade Conceptual:** Quando o sistema não reconhece os dados do utilizador como conceitos válidos, ou quando o utilizador não reconhece o conceito gerado pelo sistema como válido. Por exemplo, quando o utilizador pretende gerar um poema numa determinada métrica, não suportada pelo agente computacional. Este problema vem de encontro ao conceito de aberração de Wiggins.

**Discordância Artística:** Quando o sistema não reconhece valor nos conceitos gerados pelo utilizador, ou vice-versa. Este problema vem de encontro ao conceito de “desinspiração” apresentado por Wiggins, onde o sistema pode ser forçado a gerar conceitos aos quais não atribui valor, ou gera sistematicamente conceitos sem valor do ponto de vista estético do utilizador.

**Impotência Generativa:** Quando o sistema é incapaz de continuar a gerar artefactos no domínio do conceito introduzido pelo utilizador. Por exemplo, numa aplicação de geração de poesia, quando é sugerida como semente uma palavra que o sistema desconhece.

## 2.3 Plataformas Relacionadas

Nesta secção vão ser analisadas algumas plataformas, entre elas sistemas autónomos de geração de poesia, plataformas de suporte criativo e plataformas co-criativas de geração de poesia.

Dentro dos sistemas autónomos de geração de poesia que vão ser analisados estão: O ASPERA, sistema desenvolvido por Gervás que usa técnicas de raciocínio baseado em casos (*case-based reasoning*); O McGonnagall, sistema evolucionário descrito por Manurung (2003); O sistema descrito por Toivonen, baseado numa abordagem “Corpus-Based”, que vai ser avaliado aquando da avaliação do “Poetry Machine”, plataforma co-criativa desenvolvida por Anna Kantosalo (Kantosalo et al, 2015), que o utiliza para gerar o poema-rascunho; O “BlackBoard System” desenvolvido por Misztal (Misztal et al, 2016), baseado numa abordagem multi-agentes. Vão ser ainda avaliados o FloWr e o ConCreTe FloWs enquanto plataformas de suporte criativo, e o jGnoetry, o Poetry Machine, o LyriSys e o DeepBeat enquanto plataformas co-criativas de geração de poesia.

Numa primeira fase vai ser feita uma análise mais detalhada, tanto ao nível do processo que leva à geração de artefactos como dos artefactos propriamente ditos. As aplicações vão ainda ser enquadradas nas diferentes métricas apresentadas anteriormente: a de Manurung (2003) para avaliar os artefactos como poesia ou não-poesia; a de Dan Ventura (2016) para avaliar a presença de criatividade; e a de Anna Kantosalo (2016) para avaliar as aplicações enquanto plataformas co-criativas.

Num segundo momento, de maneira a tornar a avaliação mais fácil e útil, considera-se que um sistema gera poesia se se limita a produzir texto que cumpre os requisitos de poeticidade e gramaticalidade de Manurung. Para além disso, o sistema é considerado criativo se é capaz, autonomamente, de criar artefactos, através da definição de um conjunto inicial de parâmetros. O sistema é considerado co-criativo se permite interação com o poema-rascunho gerado, através de turnos de geração, tanto do utilizador como do sistema. O último parâmetro prende-se com a existência, ou não, de uma *interface* gráfica que permita aos utilizadores interagir com o sistema.

## 2.3.1 Sistemas autónomos de geração de poesia

### **ASPERA**

O ASPERA é um sistema de geração automática de poesia, desenvolvido por Gervás (2001), que funciona como um “tradutor de prosa para poesia”. Este sistema “combina técnicas de geração de linguagem natural e raciocínio baseado em casos, para aplicar um conjunto de heurísticas de construção obtidas” através da análise de um conjunto de poemas de inspiração, neste caso espanhóis.

O utilizador define um conjunto de parâmetros iniciais, que são: tema do poema (através da inserção de uma descrição da mensagem que se pretende transmitir); tamanho, que corresponde ao número de linhas (que Gervás designa de fragmentos); “mood”, que neste caso é mais polaridade, que pode ser positiva ou negativa.

O sistema, numa primeira fase, tenta combinar, da melhor maneira, a forma estrófica e um conjunto de poemas de inspiração. Após a definição da forma estrófica, o sistema planeia um poema-rascunho através da distribuição da mensagem introduzida pelo utilizador pela forma desejada para a estrofe. Posto isto, o sistema carrega um vocabulário específico e conjunto de versos-exemplo e gera cada linha do poema por copiar a estrutura frásica (partes do diálogo) de uma das linhas pré-selecionadas, combinada com palavras do vocabulário pré-selecionado. Depois de o poema-rascunho estar produzido, é apresentado ao utilizador para que seja validado. Caso isso aconteça, o sistema analisa linguisticamente o poema, e adiciona a informação correspondente à sua base de dados.

Analisando os artefactos através da métrica de Manurung, verificamos que estes cumprem os três requisitos. As palavras são combinadas de acordo com as regras gramaticais da língua espanhola, e esta combinação deve fazer sentido, transmitindo a mensagem inserida pelo utilizador. Para além disso, existem requisitos adicionais, impostos pela forma estrófica, para garantir a presença da rima e cumprimento da métrica. Os artefactos são ainda considerados criativos, porque apresentam novidade, valor e intencionalidade.

### **McGonnagall**

Este sistema, descrito por Manurung (2004), tem como principal objectivo a produção de poemas com significado, embora também pretenda que estes cumpram as regras gramaticais do idioma e apresentem características da poesia, como métrica e rima. Este sistema representa o processo de geração de poesia como um

problema de procura e utiliza um algoritmo de procura trepa colinas estocástico para encontrar o poema melhor pontuado, visto que cada estado representa um potencial texto de output. O sistema é assim capaz de gerar poesia regida por um conjunto de restrições métricas, usando uma determinada gramática, e através da introdução de um tópico. O processo de geração assente em duas fases:

**Fase de Avaliação:** onde um grupo de indivíduos é formado através da informação inicial, e estes indivíduos são avaliados de acordo com o cumprimento da métrica e da semântica desejadas.

**Fase de Evolução.** O sub-conjunto deste grupo inicial de indivíduos, neste caso os melhores pontuados, são selecionados para reprodução, ou seja são combinados com a esperança de serem produzidos melhores versões do poema.

O sistema chega a um estado onde os poemas gerados cumprem os três requisitos de Manurung, visto que a sua função de fitness avalia a distância semântica entre os poemas candidatos e a semântica alvo, e o cumprimento da métrica. A utilização de uma gramática base faz com que os textos produzidos também cumpram os requisitos sintáticos do idioma em que o sistema opera. Aplicando a métrica de Dan Ventura, em relação à presença de criatividade, podemos verificar que os resultados são originais, até porque resultam de técnicas similares às de combinação e transformação de Gero(2000) e Boden(2003), têm valor porque cumprem as regras do domínio, e são intencionais.

### **BlackBoard System**

O BlackBoard System é um sistema de geração de poesia desenvolvido por Joanna Misztal-Radecka e Bipin Indurkha (2016), cuja arquitetura pretende representar, metaforicamente, um conjunto de humanos, com conhecimentos diferentes, a discutir ideias num “quadro negro”. Os humanos são representados por “módulos independentes e especializados que cooperam durante o processo de geração por partilharem um espaço de trabalho comum”. Este tipo de arquitetura permite o desenvolvimento individual de módulos, bem como adicionar ou remover módulos do processo de geração. Cada módulo é visto como um especialista a desempenhar determinada função. Existem vários tipos de módulos:

**Módulos de análise textual**, que recolhem expressões e emoções de textos de inspiração (introduzidos pelo utilizador, no exemplo são usados *posts* de *blogs*) e a colocam no quadro, para serem usadas por outros módulos.

**Módulos de geração de palavras**, que geram palavras relacionadas com o tópico e as inserem no quadro negro. São gerados sinónimos, antónimos e hiperónimos dos nomes e adjectivos usados nas expressões recolhidas, bem como palavras frequentemente usadas e palavras que descrevem o mesmo estado emocional.

**Módulos de geração de poesia**, que produzem linhas para o poema, através das palavras geradas anteriormente. Estes módulos têm conhecimento gramatical, produzindo assim frases gramaticalmente correctas, onde se inclui a geração de comparações, de metáforas e de perguntas retóricas.

**Módulos de seleção**, que escolhem as melhores soluções de acordo com os requisitos inseridos inicialmente pelo utilizador. Existem módulos de contagem de sílabas, que escolhem as frases que melhor encaixam na métrica desejada, e módulos de rima que seleccionam os melhores conjuntos de frases que rimam.

Este processo de geração de poesia apresenta dois tipos de co-criatividade. Em primeiro lugar, apresenta co-criatividade entre o humano e o sistema, sendo esta componente bastante limitada. Isto porque o utilizador apenas pode definir um conjunto de parâmetros iniciais, neste caso os textos de inspiração e a forma poética desejada. Em segundo lugar, apresenta co-criatividade entre módulos, que operam todos o mesmo conjunto de ideias, no mesmo espaço de trabalho, e cooperam uns com os outros com o objectivo de gerar um poema.

Ao nível dos artefactos gerados, podemos dizer que são considerados poesia, através da métrica de Manurung (2003), por cumprirem as regras de poeticidade, gramaticalidade e significado, visto que têm módulos especialistas a avaliar estas componentes. O sistema é também considerado criativo por apresentar novidade, valor e intencionalidade. Este sistema gera intencionalmente linguagem figurativa, eliminando qualquer discussão na atribuição de intencionalidade ao sistema.

## 2.3.2 Plataformas de suporte criativo

### FloWr

O FloWr<sup>1</sup> (Charnley, Colton, & Llano, 2016) é um sistema que foi construído com o objectivo de permitir, aos investigadores do ramo da criatividade computacio-

---

<sup>1</sup><http://ccg.doc.gold.ac.uk/research/flowr/>

nal, criar novos sistemas através de uma *interface* visual, construindo fluxogramas que passam informação através de nós de processamento, como mostra a Figura 3. Com a criação desta aplicação, os autores pretendem criar uma comunidade de utilizadores que contribua tanto no desenvolvimento dos módulos de processamento iniciais, como dos próprios fluxogramas.

A geração do código está organizada em módulos, que podem ser combinados com outros, com uma descrição da ordem de execução de cada módulo (paradigma de programação visual). O FloWr é uma aplicação on-line, e permite partilha de fluxogramas ou utilização de fluxogramas públicos em projectos pessoais.

Charnley, Colton, Llano e Corneli (2014) criaram, através deste processo de geração de fluxogramas, uma aplicação geradora de poesia. O processo iniciava com a recolha de um conjunto de palavras mais frequentes de um determinado dicionário. Era feita uma seleção por adjetivos, que posteriormente eram avaliados pela sua conotação sentimental (positiva/negativa). Recolhiam-se depois sinónimos desses adjetivos, e um deles, de conotação negativa, era escolhido aleatoriamente como tema do poema. Posteriormente era feita uma pesquisa e recolha de *tweets* que contivessem a palavra-tema, e eliminados tanto os repetidos como os com expressões indesejadas. De seguida eram analisados os *tweets* com pronomes pessoais, e era feita uma pesquisa por pares de *tweets* com os mesmo fonemas no final, bem como com o mesmo número de sílabas. Através destes pares de *tweets* eram compostas quadras, das quais eram escolhidas aleatoriamente duzentas e cinquenta e formatadas para uma determinada estrutura poética. O poema gerado com maior conotação negativa atribuída era escolhido como resultado e guardado num ficheiro.

Se analisarmos este sistema de acordo com as métricas acima apresentadas, é discutível tanto a criatividade do sistema como a classificação dos resultados como poesia. É discutível a atribuição de criatividade porque a maior parte das escolhas do sistema são aleatórias. Ainda assim o sistema tem a intenção de gerar artefactos com um valor sentimental negativo associado, e que sejam sobre determinado tema. Para além disso utilizam os processos de combinação e transformação apresentados tanto por Boden (2003) como por Gero (2000). Embora os artefactos gerados cumpram os requisitos de gramaticalidade e poeticidade, é discutível que o sistema atribua intencionalmente um significado ao poema. Ainda assim, o sistema procura *tweets* que contenham o tema escolhido e tem intenção construir um poema de conotação negativa.



O FloWr é uma plataforma de suporte criativo, no sentido em que auxilia o processo criativo humano, visto que permite estruturar ideias e processos sem uma preocupação tão “de baixo-nível” e de forma rápida. Para além disso, o facto de correr num *browser* torna a aplicação portátil.

No entanto, e no âmbito deste projecto, acho mais interessante a questão da automatização da geração de fluxogramas. Na prática, isto significa que temos um *software* a criar *software*, o que elimina qualquer cepticismo no que toca à atribuição de criatividade a um sistema. Por isso achei mais interessante as tentativas de geração de poesia, através da combinação aleatória de nós de processamento. No entanto, de um conjunto de 200 *scripts* gerados automaticamente pelo sistema, apenas 17 criavam artefactos que cumpriam as regras da estrutura poética. Ainda assim considero esta funcionalidade muito interessante, porque o sistema deixa de apenas seguir instruções, para a partir de um determinado conjunto de instruções possíveis, escolher as que pretende usar (embora neste caso a escolha seja aleatória, ou seja, não há qualquer tipo de intencionalidade na escolha de um nó em detrimento de outros).

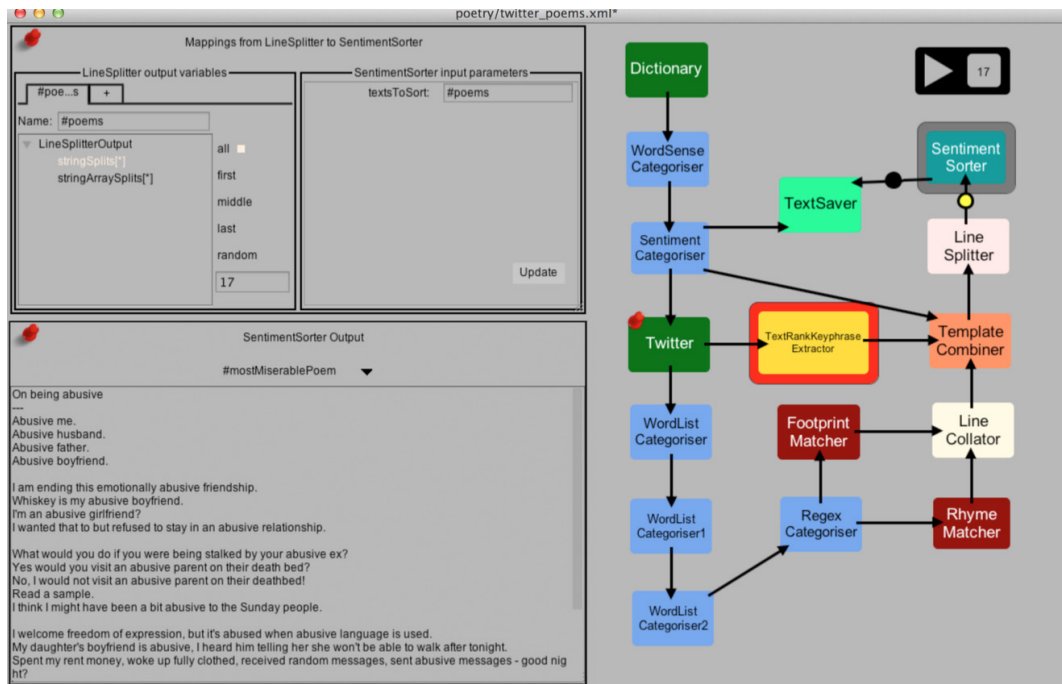


Fig.3- Interface visual do FloWr, com o fluxograma de geração de poesia criado.



## ConCreTe Flows

À semelhança do FloWr, o ConCreteFlows<sup>2</sup> é uma ferramenta de suporte criativo, disponível *on-line*, que também apresenta a componente de partilha e reutilização de fluxogramas, como demonstra Figura 4. Esta aplicação tem o principal objectivo de ajudar os utilizadores a criar sistemas de execução de tarefas computacionais criativas. Tal como no FloWr, inicialmente temos à disposição um conjunto de componentes/módulos que podem ser combinados de modo a criar um fluxograma por onde a informação é passada e transformada. O ConCrete Flows também usa o paradigma da programação visual, através de uma *interface* que representa de forma abstrata o fluxo de trabalho e os seus componentes (representa procedimentos complexos através de um arranjo visual de blocos).

Znidarsic (et al, 2016) desenvolveram, por exemplo, um sistema que através da definição de dois conceitos, é capaz de os fundir e gerar um artefacto que pode tomar a forma de um grafo, de um texto, de uma imagem, etc. Isto passa por, no âmbito da geração do artefacto, utilizar características dos dois conceitos inicialmente introduzidos.

Mais tarde, foi introduzido um módulo do PoeTryMe, um sistema de geração de poesia, que produz um poema de acordo com os conceitos introduzidos inicialmente. O facto de este tipo de aplicação permitir a inclusão de serviços *Web* externos, neste caso o PoeTryMe, abre o leque de possibilidades de sistemas possíveis de criar.

Sendo o PoeTryMe o gerador de poesia, neste caso, a análise deste sistema corresponde à análise do PoeTryMe. Podemos dizer que os artefactos são criativos porque apresentam novidade, valor e intencionalidade, porque escolhe palavras do domínio semântico do tema. Os processos que o sistema usa correspondem aos de combinação e transformação de Boden (2003) e Gero (2000). Os artefactos são considerados poesia porque cumprem os três requisitos da métrica de Manurung (2003). O sistema tem ainda uma *interface Web* e é co-criativo, visto que permite que o utilizador defina um conjunto de parâmetros iniciais, no entanto, esta componente co-criativa é muito limitada.

---

<sup>2</sup><http://concreteflows.ijs.si/>

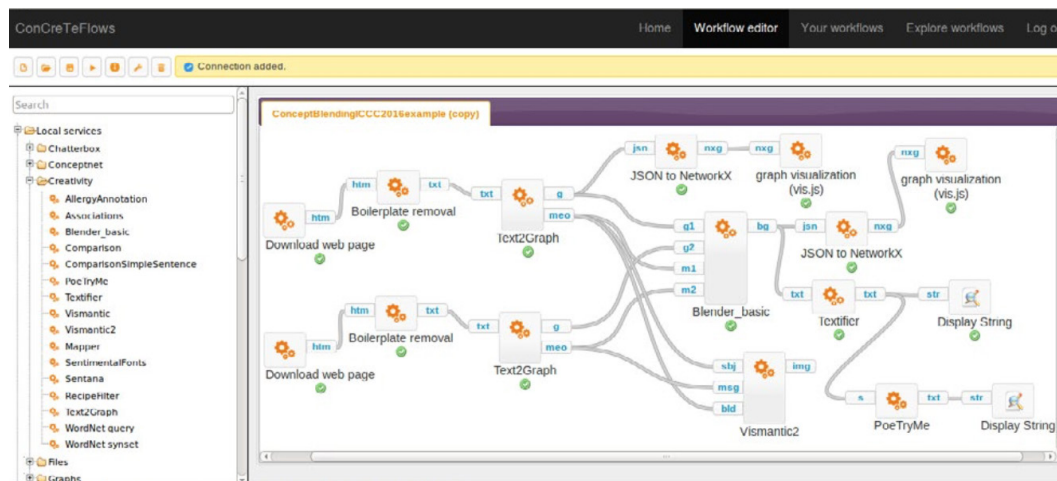


Fig.4- Interface visual do ConCreTe Flows, com o fluxograma do sistema descrito.

## 2.3.3 Sistemas co-criativos de geração de poesia

### jGnoetry

O jGnoetry<sup>3</sup> é uma plataforma co-criativa de geração de poesia (Figura 5), que permite gerar um poema, a partir de textos de exemplo inseridos pelo utilizador e permite personalizar a forma silábica bem como mais algumas características do poema final.

Relativamente ao processo de geração, o sistema usa os textos inseridos para “analisar como as palavras são usadas nos textos e tenta criar padrões”, através de técnicas de Modelação Estocástica da Linguagem (n-gramas). O jGnoetry permite inserir vários textos de inspiração e a influência que cada texto terá no poema final. Com relação à forma silábica, os utilizadores podem escolher uma forma poética das disponíveis: dístico, quadra, soneto, verso livre (quadra), haiku, tanka e renga (os três últimos são formas de poesia tradicional japonesa). Os utilizadores podem ainda editar a forma escolhida, por inserir sílabas ([s]) ou quebras de linha([n]) diretamente no *template*.

O sistema permite editar algumas opções de interação com o poema, tais como método de seleção de palavra (passar o rato em cima – *hover* – ou *click*), pontuação;

<sup>3</sup><https://gnoetrydaily.wordpress.com/2011/12/27/presenting-jgnoetry/>

início de linha no poema-rascunho (baseado na pontuação ou na quebra de linha dos textos de inspiração); uso da pontuação de acordo com o conjunto de inspiração (sim, não, ou apenas não usar aspas); pontuação no final do poema (ponto final, ponto de interrogação, ponto de exclamação ou nenhum); e capitalização (como nos textos originais, nenhuma ou personalizada com os seguintes parâmetros selecionáveis: capitalização do início de frases, no início de linhas e da letra “i” quando aparece sozinha). Após a geração do poema-rascunho, o sistema apresenta alguns detalhes da geração, como por exemplo o template usado e a sequência de palavras escolhidas.

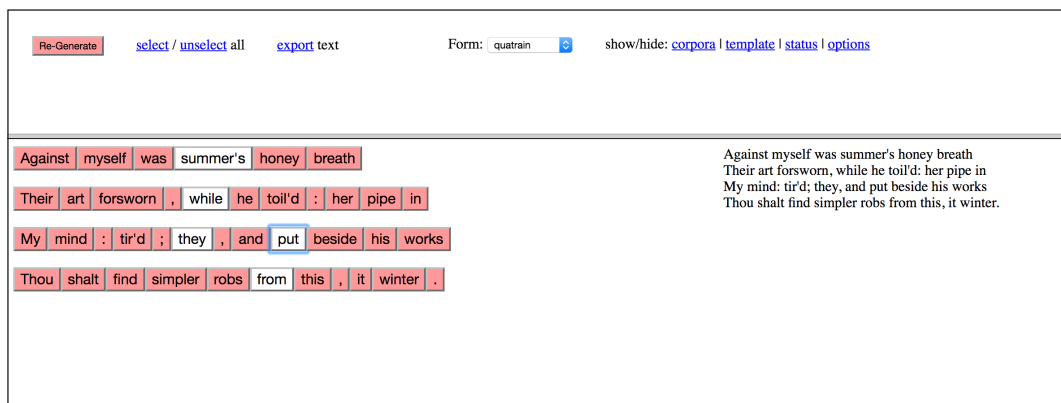
Na geração do poema-rascunho o utilizador escolhe o tema através da inserção de textos relacionados e limita, através da forma poética selecionada, a escolha das palavras por parte do sistema. No entanto, é discutível a intencionalidade do sistema na geração do poema-rascunho. Numa análise dos artefactos gerados, verificamos que estes cumprem os requisitos métricos e gramaticais. O sistema “não contém nenhum conhecimento *à priori* da gramática” que opera, mas “analisa quão frequentes são as aparições de várias combinações de palavras ao longo dos textos e usa a informação para gerar sequências de palavras”. Esta abordagem, por norma, resulta em versos com significado, embora possa fraquejar na coesão entre versos. Visto que estamos a falar de uma plataforma co-criativa, este tipo de situações podem ser resolvidas pelo utilizador, conferindo a consistência desejada ao poema e validando a intencionalidade na transmissão de um significado.

A meu ver, o mais interessante nesta aplicação é o processo co-criativo após a geração do poema-rascunho. Cada palavra do poema gerado é apresentada num rectângulo selecionável. A ideia é selecionar todas as palavras que pretendemos substituir (pode selecionar-se apenas uma palavra, várias palavras de uma linha ou de linhas diferentes, um verso inteiro ou mesmo todo o poema) e carregar no botão de re-geração. O sistema vai manter todas as palavras não selecionadas, embora estas possam mudar de linha ou de posição na mesma linha, e substituir todas as palavras selecionadas por outras que cumpram os mesmos requisitos métricos e da forma poética. Este processo de seleção e regeneração de palavras por parte do utilizador ocorre ciclicamente até o este ficar satisfeito com o resultado.

Se analisarmos o sistema co-criativo à luz da categorização de Anna Kantosalo (2016), percebemos que o sistema se enquadra na co-criatividade dividida por tarefas. O sistema e a pessoa não tomam turnos iguais no que toca à geração de conteúdos. O utilizador define o domínio a operar, através da introdução de tex-

tos de inspiração, e restringe o sistema a cumprir determinadas regras de métrica e de forma poética. O sistema assume o papel de gerador de conceitos/conteúdos, porque o utilizador não pode inserir ou modificar palavras diretamente, isto é, apenas pode substituir uma palavra por outra gerada pelo sistema, que pode não corresponder à que o utilizador pretendia inserir. No que diz respeito à avaliação e validação, é apenas o utilizador que desempenha esses papéis, visto que o sistema apenas leva em conta regras métricas e gramaticais.

Analisando o design do sistema, nota-se que houve pouca preocupação no sentido de tornar a aplicação mais acessível para os utilizadores. Não houve estilização dos elementos HTML, a não ser as palavras seleccionáveis. Há alguns tópicos do menu que não são os mais fáceis de associar à área da aplicação a que correspondem. No entanto, em termos de interação, achei interessante o facto das palavras serem seleccionáveis, e a distinção entre palavras a manter e palavras a substituir parece-me perfeitamente perceptível, e sendo esse processo, do meu ponto de vista, o processo co-criativo principal, acho que foi bem conseguido.



**Fig.5-** Screenshot da Web-interface do jGnoetry.

### Poetry Machine

“The Poetry Machine”(Kantosalo et al., 2015) é uma aplicação co-criativa de geração de poesia, cujos principais objectivos são a ajuda de crianças no estudo da poesia e a resolução da síndrome da folha branca (Figura 6).

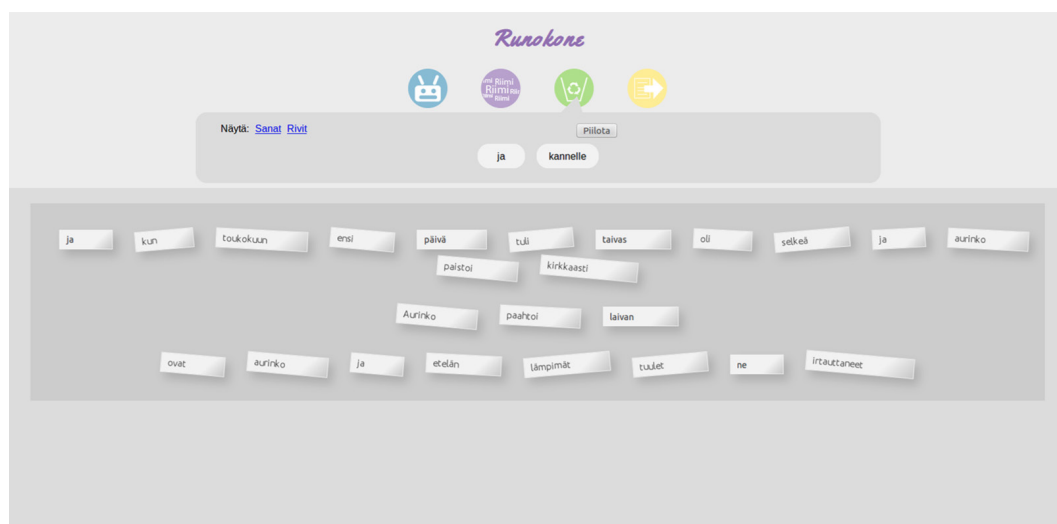
Numa fase inicial, o utilizador pode escolher o tema do poema (de um conjunto de oito possibilidades), e o sistema gera um poema-rascunho sob a forma de segmentos. Estes segmentos podem ser arrastados, removidos ou editados. O utilizador pode ainda pedir ao sistema que gere novas palavras ou linhas com determinadas características de rima. Estes fragmentos são extraídos automaticamente de um *corpus* que contem literatura infantil do Projeto Gutenberg.

O processo de geração de poesia é feito através da utilização dos métodos desenvolvidos por Toivanen (2012), embora a versão descrita (Kantosalo et al., 2015) use apenas parte desses métodos. Após a atribuição de um tema, por parte do utilizador, o sistema “seleciona aleatoriamente um pedaço de texto com o tamanho desejado do conjunto de inspiração” (Toivanen et al, 2012) e analisa morfológicamente as palavras que o constituem. As palavras são substituídas por outras relacionadas com o tema, ou ficam intactas caso estas não existam com a mesma morfologia. Caso o número de palavras substituídas ultrapasse um determinado limite, o poema é selecionado como resultado; caso contrário, o processo é repetido para outro excerto selecionado aleatoriamente do conjunto de inspiração. Posto isto, podemos considerar que os poemas gerados apresentam valor e intencionalidade, sendo a novidade discutível, visto que basta que “metade das palavras do poema de inspiração sejam substituídas” para que o poema seja apresentado como resultado. No entanto os artefactos apresentam sempre algum grau de novidade.

Analisando os artefactos, percebemos que o sistema cumpre os requisitos de poeticidade e gramaticalidade. Quanto ao significado, o sistema substitui palavras do texto de inspiração por palavras do campo semântico desejado, por isso pode-se atribuir intencionalidade à atribuição de significados. Mais uma vez, por ser uma plataforma co-criativa, podemos considerar que o utilizador é capaz de conferir significado ao poema e, por isso reforçar o significado que pretende transmitir.

Enquadrando o sistema na categorização de plataformas co-criativas de Anna Kantosalo (2016), verificamos que o agente computacional e humano assumem turnos desiguais de interação ao poema – co-criatividade dividida em tarefas. Na verdade, embora ambos os agentes possam gerar conteúdos, é sempre o utilizador que define o domínio a operar, e também é o utilizador que serve de validador dos conteúdos gerados. Embora o sistema também avalie os artefactos, em relação ao nível de novidade conseguido, essa avaliação é muito limitada.

Olhando para o design da aplicação, e embora não esteja disponível uma *interface Web* para utilização do sistema, podemos ver, através da análise de *screenshots* da aplicação, que a representação do poema-rascunho em segmentos é muito interessante. Cada um dos segmentos pode ser re-gerado ou editado, o que representa uma novidade quando comparamos esta plataforma com o jGnoetry.



**Fig.6-** Screenshot do protótipo em “modo-escrita”, com a ferramenta “robot” aberta.

### LyriSys

O LyriSys é uma aplicação de apoio à composição de letras de música, que permite que utilizadores, através da interação com o sistema, criem uma letra de música de “forma incremental e experimental”, tanto em inglês como em japonês. Esta aplicação foi desenvolvida por Kento Wantanabe, Yuichiroh Matsubayashi, Kentaro Inui, Tomoyasu Nakano, Satoru Fukayama e Masataka Goto.

Em primeiro lugar o sistema permite a definição da história da letra, através da definição de um conjunto de tópicos semânticos (não há liberdade total na escolha de tópicos, mas é possível escolher vários de uma lista de possibilidades), bem como pela definição da ordem de aparição destes tópicos ao longo da letra. Para além disso, é possível definir, para cada um destes tópicos, a estrutura poética/métrica desejada, através da definição do número de linhas, sílabas e pala-

vas. Após a definição destas características, o sistema é capaz de criar, de forma autónoma, uma letra de musica, que cumpre o conjunto de requisitos definido inicialmente. A escolha das palavras é feita através do uso de um algoritmo probabilístico, que aprende a probabilidade de certas palavras aparecerem juntas ou transmitirem determinada emoção/sentimento através da análise de um conjunto de letras de musica escritas por humanos.

Após a geração da letra-rascunho, o utilizador pode substituir cada uma das linhas por outras linhas candidatas geradas pelo sistema. Para além disso, o utilizador pode ainda editar as linhas, individualmente, através da introdução directa de texto.

Visto que não está disponível nenhuma *interface Web* que permita interagir com o sistema, é difícil de deduzir o seu funcionamento, mesmo havendo alguns *screenshots* da aplicação.

Analisando os artefactos gerados à luz da métrica de Manurung (2003), podemos verificar que estes cumprem os requisitos da poeticidade, gramaticalidade e pretendem transmitir uma mensagem conceptual. Esta última componente de transmissão de um significado foi até mais explorada do que nas plataformas anteriores, visto que é possível transmitir uma história através da definição da ordem de aparição dos tópicos semânticos escolhidos. Para além disso, os artefactos podem também ser considerados criativos porque apresentam novidade, valor e intencionalidade.

Analisando a aplicação à luz da categorização de Anna Kantosalo (2016), esta plataforma enquadra-se na co-criatividade dividida em tarefas. Isto porque embora ambos os agentes sejam capazes de gerar conteúdos, é apenas o utilizador que define o conjunto de parâmetros iniciais e avalia os conteúdos gerados.

### **DeepBeat**

O DeepBeat<sup>4</sup> é uma aplicação co-criativa de apoio à composição de letras de música, neste caso mais direccionada para a geração de letras de Rap, criada por Eric Malmi, Stephen Fenech e Pyry Takala. Esta aplicação usa estratégias de Inteligência Artificial para combinar linhas de letras de música Rap já existentes, de maneira a fazerem sentido (a nível semântico) juntas, e de maneira a rimarem.

---

<sup>4</sup><http://deepbeat.org/>

Num primeiro momento, o DeepBeat permite que o utilizador defina o idioma das letras a serem geradas, bem como um conjunto de palavras-chave que definirão o tema da letra. Permite ainda ativar ou desativar o “Deep Learning”, funcionalidade que quando ativa melhora os resultados produzidos ao nível da coerência entre linhas, embora, por outro lado torna o processo de geração mais lento.

Depois da definição destes parâmetros iniciais, é possível optar por duas estratégias de criação da letra: gerar a letra na íntegra, ou linha a linha. Caso se pretenda gerar a letra toda de uma vez, é possível definir quais das palavras-chave pretendemos que estejam presentes. Depois da letra ser gerada na totalidade, podemos passar a um processo de edição das linhas, semelhante ao processo que utilizaríamos se gerássemos a letra linha a linha. Para cada linha da letra é possível escrever texto diretamente, ou pedir ao sistema que gere um conjunto de linhas, que contenham determinado subconjunto (ou a totalidade) das palavras-chave. É ainda possível definir se pretendemos uma linha que rime com a anterior ou simplesmente uma linha, sem restrições de rima. O sistema abre uma janela com um conjunto de linhas sugeridas, e o utilizador pode selecionar uma para incluir na letra. Quando o utilizador estiver satisfeito com a letra composta, pode gravá-la e partilhá-la tanto no Facebook como no Twitter.

Analisando os artefactos gerados pelo sistema, verificamos que cumprem os três requisitos de Manurung (2003). Embora o sistema não tente seguir uma métrica específica quando gera determinada letra de música, ele privilegia a ocorrência de rimas. Quanto à presença de criatividade, podemos dizer que o sistema usa técnicas semelhantes às transformação e combinação descritas por Boden (2003) e Gero (2000) e que os artefactos gerados apresentam novidade, pois embora usem linhas de letras já existentes, estas surgem combinadas de uma maneira nunca antes explorada, valor e intencionalidade.

Enquadrando esta plataforma na categorização de Anna Kantosalo (2016) verificamos que embora ambos os agentes possam gerar conteúdos, tanto a definição de conceitos como a avaliação/validação de conteúdos são tarefas que ficam a cargo do utilizador. Posto isto, o DeepBeat enquadra-se na co-criatividade dividida em tarefas.

Quanto ao design da aplicação, penso que este capta a essência deste estilo musical, e a aplicação é muito simples e fácil de utilizar, visto que é preciso pouco tempo para dominar todos os aspetos da mesma (Figura 7).



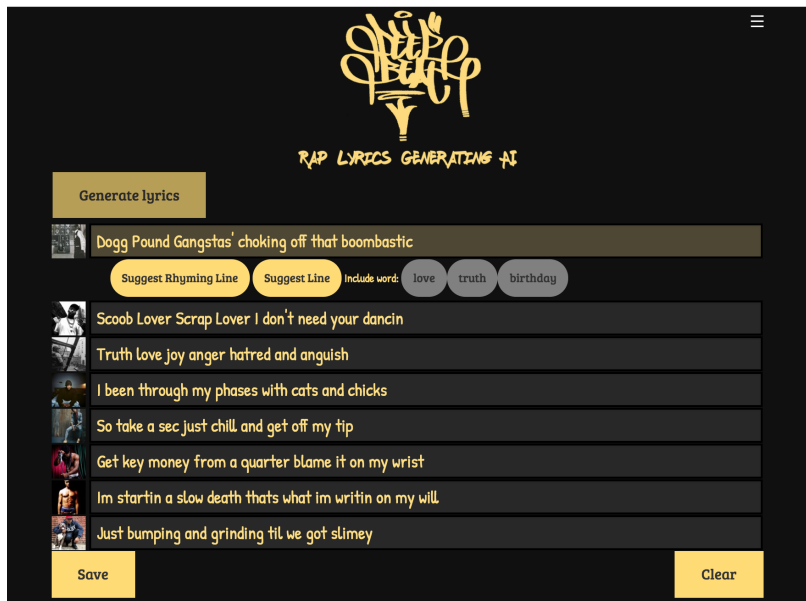


Fig.7- Screenshot da aplicação DeepBeat.

### 2.3.4 Avaliação das Plataformas Analisadas

De maneira a facilitar a avaliação das plataformas analisadas, as métricas vão ser agora simplificadas, com o objectivo de eliminar situações em que a avaliação pode ser subjectiva. Posto isto, um texto gerado por um programa de computador é considerado poesia se cumpre as regras de poeticidade e gramaticalidade de Manurung (2003), é considerado criativo se é capaz de criar artefactos, de forma autónoma, através da definição de um conjunto de parâmetros, e é considerado co-criativo se permite interação com o resultado da primeira geração. É ainda registado se a plataforma em questão dispõe ou não de uma *interface* gráfica *Web* que permita aos utilizadores experimentarem a aplicação. A Tabela 1 mostra o resultado da avaliação feita.

	Poesia	Criativo	Co-criativo	Interface Gráfico
ASPERA	Sim	Sim	Não	Não
McGonnagall	Sim	Sim	Não	Não
BlackBoard	Sim	Sim	Não	Não
FloWr	Não unicamente	Não	Não	Sim
ConCreTe Flows	Não unicamente	Não	Não	Sim
Poetry Machine	Sim	Sim	Sim	Sim
jGnoetry	Sim	Sim	Sim	Sim
LyriSys	Sim	Sim	Sim	Sim
DeepBeat	Sim	Sim	Sim	Sim

**Tab. 1-** Avaliação das plataformas analisadas.

## 3 Design de Aplicação Interativas

Tendo em conta que o objectivo deste trabalho envolve o desenvolvimento de uma aplicação *Web* que sirva de *interface* à versão mais recente da API do sistema PoeTryMe, é necessário fazer um estudo sobre design de *interface* e interação.

Numa fase inicial, é feito um levantamento de regras e diretrizes para o design de *interfaces*, mais concretamente para uma *interface Web*, que pretendem guiar o desenho dos ecrãs a mostrar ao utilizador.

Posteriormente, são identificados alguns requisitos do ramo do design de interação que têm como objectivo desenhar a interação do utilizador com o computador, de maneira a que a aplicação seja fácil de utilizar. O termo usabilidade é um dos mais utilizados quando falamos em design de interação, e também vai ser abordado mais adiante.

### 3.1 Interfaces com o utilizador

O termo design de *interface* refere-se ao desenho da *interface* de uma aplicação computacional, isto é, o design de *interface* não se preocupa com a interatividade, mas debruça-se sobre o desenho dos vários ecrãs estáticos que são apresentados ao utilizador. Para Schneiderman (1998), uma *interface* é bem desenhada quando desaparece, isto é, torna-se tão intuitiva e familiar, que o utilizador é capaz de prever o resultado da utilização de cada ferramenta, podendo-se assim focar no desenvolvimento de tarefas propriamente dito. Ben Schneiderman apresenta um conjunto de critérios, designados por “regras de ouro”, aos quais as *interfaces* devem obedecer:

**Manter a Consistência:** Deve haver consistência na terminologia usada, das sequências de ações, no uso de cor e da tipografia, que devem ser iguais para situações de utilização semelhantes;

**Universalidade:** Pressupõe um estudo diligente das necessidades dos vários tipos de utilizadores e a inclusão de funcionalidades que supram as necessidades desses diferentes tipos (por exemplo, incluir explicações/tutoriais ajuda os utilizadores novos a aprender a utilização da aplicação, enquanto que a inclusão de atalhos ajuda os utilizadores mais experientes a optimizarem a utilização da aplicação);

**Oferecer *feedback* informativo:** Todas as acções do utilizador devem ter um *feedback*, que pode ser mínimo em acções de menor importância, mas que deve ser mais substancial em situações de maior importância.

**Diálogos indicadores de final de acção:** O utilizador deve conseguir distinguir as várias fases de uma acção. O facto de haver um *feedback* informativo no fim de cada tarefa (que pode incluir várias acções), transmite ao utilizador uma sensação de conquista e satisfação, servindo como reforço para que este volte a usar a aplicação em tarefas semelhantes.

**Prevenir erros:** O design da aplicação deve impedir o utilizador cometer erros graves, e quando comete algum erro, este deve ser detectado e o sistema deve oferecer uma forma simples e eficaz de o resolver.

**Permitir reversão de acções facilmente:** Todas as acções devem ser reversíveis porque este factor ajuda a diminuir a ansiedade do utilizador no uso da aplicação. Este tipo de abordagem também encoraja a exploração e experimentação do sistema.

**Suportar o controle por parte do utilizador:** Os utilizadores, especialmente os mais experientes, gostam de sentir que dominam a *interface* e que o sistema só responde a acções.

**Reduzir informação a decorar:** O sistema não deve obrigar o utilizador a decorar grandes quantidades de informação num curto espaço de tempo. Pelo contrário, essa informação deve estar sempre disponível para o utilizador consultar sempre que sentir necessidade.

Neste caso, como o suporte da *interface* é um *browser*, há um conjunto de regras e directrizes dirigidas para este suporte. O *Web-design* é o ramo do design que se debruça sobre as *interfaces* quando o suporte é um *browser*. Visto que um *browser* pode ser acedido tanto por computadores, como por dispositivos móveis, é necessário que haja uma preocupação em adaptar a visualização do sistema ao dispositivo que lhe está a aceder.

Existem duas abordagens possíveis de seguir no desenvolvimento de *layouts* responsivos para *Web* (Lupton, 2014):

**Líquida:** O conteúdo ajusta-se continuamente à largura do ecrã.

**Adaptativa:** O conteúdo ajusta-se ao ecrã baseado em passos predefinidos de tamanhos de ecrã, que na prática corresponde à construção de um *layout* para computador, outro para *tablet* e outro para *mobile*. O *layout* mostrado está dependente do tamanho e orientação do dispositivo que lhe acede.

Visto que se trata de uma aplicação onde o texto é o agente principal, é necessário considerar algumas regras de utilização da tipografia em meios digitais.

No que toca à escolha do(s) tipo(s) de letra, é preciso levar em conta que o texto na página está dividido entre cabeçalhos e corpos de texto. Esta distinção pode ser feita utilizando sempre o mesmo tipo de letra, através da utilização de vários pesos e tamanhos, ou em alternativa, por combinar tipos de letra diferentes. Os cabeçalhos devem refletir a importância do texto que os segue, permitindo ao leitor analisar o documento ao seu ritmo e escolher o que pretendem ler ou não. Para os títulos, como são usados tamanhos maiores, pode-se usar um tipo de letra com detalhes apelativos “a escolha pode ser vista como o alho e a pimenta para a carne e as batatas” (Lupton, 2014), enquanto que a escolha do tipo de letra para o texto deve ser mais sóbria, com o objectivo de que o texto seja lido sem esforço, através de fontes compostas por formas bem desenhadas e subtis. “Em 2013, um estudo global revela que os sites continuam a tradição de usar tipos de letra serifados para o texto, e não serifados para os cabeçalhos” (Lupton, 2014). No entanto, e o mais importante salientar, é que o uso da tipografia deve reforçar a hierarquia dos elementos.

Ellen Lupton (2014), indica algumas directrizes que permitem avaliar um tipo de letra em meios digitais:

**Legibilidade** (Legibility): Quão diferentes são os caracteres. As formas mais orgânicas e individualizadas lêem-se melhor que as modulares e geométricas, pois acentuam a diferença entre caracteres;

**Leiturabilidade** (Readability): O tipo de letra é confortável e convida a leitura de longo prazo;

**Flexibilidade:** Verificar se o tipo de letra funciona bem em vários tamanhos e pesos, ou é preferível usar uma combinação de tipos de letra;

**Espetacularidade:** A fonte é memorável por conter detalhes apelativos, tais como números ou caracteres com detalhes únicos.

**Amfibiosidade:** Saber se a fonte foi otimizada para ecrã ou não.

Segundo Ellen Lupton (2014), o bom uso da tipografia faz com que esta se misture com o fundo, passando despercebida.

O tamanho do texto é um aspeto crucial no desenho de uma *interface*. Há vários tipos de medidas para tipografia aceites pelo CSS: pixels, pontos, percentagens e “ems”. As duas últimas são preferíveis em detrimento das duas anteriores por serem escaláveis, isto é, ajustam-se ao tamanho do ecrã e não quebram o design do site. Independentemente do tamanho do ecrã, é preciso pensar que a tipografia em meios digitais deve ser usada em tamanhos mais generosos em comparação com os tamanhos usados para impressão, devido à imprecisão do anti-*aliasing*. A capacidade do *browser* de mostrar e esconder informação também é um factor que permite que o designer utilize tamanhos de letra mais generosos, em comparação com ambientes estáticos. Ainda assim, a tipografia em dispositivos móveis pode ser mais pequena do que em computadores com ecrãs maiores, porque o utilizador pode ajustar a distância entre o ecrã e os seus olhos.

Os utilizadores de meios digitais estão habituados a um conjunto de interações, tais como *click*, arrastar, colocar o rato por cima, etc. Se levarmos em conta outros dispositivos, encontramos novas formas de interação, tais como: introduzir, premir, deslizar, etc. Quando a tipografia suporta este tipo de interação, devem ser dadas pistas ao utilizador de que este tipo de interação está disponível. Isto pode ser feito através de animações subtis, tais como a mudança de cor, sublinhado, sombras, etc.

Num site, um “caminho” é uma rota que conecta conteúdos, por isso os elementos de navegação, neste caso baseados em texto, devem ser explícitos, para o utilizador, de forma a que este perceba para que estado a interação com um elemento de navegação o vai levar. A tipografia pode servir como navegação de várias maneiras, no entanto, todo o texto que representa uma possibilidade de interação deve ser perfeitamente identificável e distinguível do resto da informação.

## 3.2 Design de Interação

O design de interação preocupa-se mais com as respostas do sistema ao conjunto de ações possíveis. Nielsen (1994) apresenta um conjunto de regras gerais a levar em conta na construção de uma aplicação interativa:

**Visibilidade do estado do sistema:** O sistema deve sempre informar o utilizador sobre o que está acontecer, através de feedback adequado;

**Ligação entre a linguagem do mundo real e do sistema:** O sistema deve “falar” a língua do utilizador, usando terminologia que lhe seja familiar;

**Controlo e liberdade de utilização:** O utilizador deve ter a possibilidade de desfazer ou refazer acções, para que se sinta livre para explorar o sistema.

**Consistência e standardização:** Deve ser possível padronizar a execução de tarefas semelhantes, logo deve haver consistência na aparência da aplicação em situações idênticas.

**Prevenção de Erros:** Moldar e restringir o *interface* de forma a prevenir situações de erro ou conflito, por exemplo não permitir a introdução de algarismos quando o sistema está à espera de receber um *input* textual.

**Reconhecimento em vez de memorização:** Não se deve obrigar o utilizador a memorizar grandes quantidades de informação, pelo contrario, deve-se mantê-la sempre disponível para ser consultada sempre que haja necessidade.

**Flexibilidade e eficiência no uso:** A aplicação deve permitir, nomeadamente aos utilizadores experientes, a utilização de atalhos/aceleradores do processo de maneira a agilizar o desempenho de tarefas;

**Estética e design minimalista:** A informação deve ser simples, sucinta e pertinente no âmbito da situação de utilização, e moderada no que toca a utilização de elementos cuja funcionalidade é simplesmente estética.

**Facilitar reconhecimento, análise e recuperação de erros:** O *feedback* deve ser explicativo e preciso na identificação de erros e respectivas causas, bem como na apresentação de soluções para os corrigir.

**Ajuda e Documentação:** O sistema deve fornecer ao utilizador informação de ajuda à sua utilização, informação esta que deve ser de fácil pesquisa e focada nas tarefas dos utilizadores.

Esta abordagem é completa e descritiva, e proporciona um raciocínio baixo-nível e detalhado no processo de criação de uma aplicação.

### 3.3 Usabilidade

Usabilidade é um dos termos mais utilizados quando o assunto é avaliação de *interfaces* com o utilizador e interação. Há muitas definições para o conceito de usabilidade. Na definição da International Standards Organization (ISO 9241-11), a usabilidade é uma extensão do produto que permite usá-lo, com a intenção de satisfazer determinados objectivos com eficácia, eficiência e satisfação (Schneiderman, 1998).

Steve Krug (2005) define usabilidade como simplesmente fazer com que alguma coisa funcione bem: fazer com que uma pessoa com uma habilidade e experiência normal consiga utilizá-la para o seu propósito sem ficar frustrada. Tom Tullis e Bill Albert (2013) definem usabilidade como sendo a habilidade de o utilizador usar uma coisa para desempenhar uma determinada tarefa com sucesso. Muitas vezes o termo usabilidade confunde-se com o conceito de experiência do utilizador. No entanto, a usabilidade foca-se mais no cumprimento de objectivos de forma simples e eficaz, enquanto que a experiência do utilizar é muito mais abrangente (nomeadamente inclui a usabilidade). O conceito de experiência de utilizador envolve também outros factores, incluindo factores humanos, de design, de interação, acessibilidade, etc. e pretende proporcionar ao utilizador uma experiência de navegação agradável e com significado ao longo da interação.

Embora os termos usabilidade e experiência do utilizador sejam distintos, na altura de avaliar a usabilidade também estamos, em parte, a avaliar a experiência do utilizador. Isto porque as métricas de usabilidade se aplicam às pessoas e aos seus comportamentos em consequência à interação com o sistema.

Na ISO 9241-11 (Schneiderman, 1998), são apresentados os seguintes requisitos da usabilidade:



**Eficácia:** Verificar se o utilizador é ou não capaz de desempenhar a tarefa a que se propõe ou a tarefa que lhe é proposta;

**Eficiência:** A quantidade de esforço necessário para o desempenho da mesma tarefa. Para este factor, uma das métricas usadas é, por exemplo, o número de *clicks* necessários para desempenhar essa mesma tarefa;

**Satisfação:** O grau de satisfação do utilizador por ter conseguido desempenhar a tarefa pretendida. Este factor é difícil de recolher e subjectivo, visto que é analisado na fase de testing, através da reacção dos utilizadores.

Scheiderman (1998), apresenta algumas medidas mais fáceis de recolher, no processo de interação, e que permitem averiguar se o utilizador consegue, com facilidade, cumprir os objectivos:

**Tempo de aprendizagem:** Tempo que um utilizador-tipo demora a desempenhar um determinado conjunto de acções relevantes para concluir a tarefa a que se propõe.

**Velocidade da Performance:** Tempo que o utilizador demora a realizar as tarefas pretendidas ou propostas.

**Taxa de erros:** Contagem do número de erros e respectiva caracterização. Permite identificar quais tarefas conduzem a um maior número de erros, e que por isso são menos intuitivas.

**Retenção:** Verificar se os utilizadores aprendem com os erros, ou cometem os mesmos tipos de erros repetidamente.

**Satisfação:** Verificar se os utilizadores gostaram do processo de interação ou foi uma experiência frustrante. Neste parâmetro, que também presente na métrica anterior, a avaliação da usabilidade e da experiência do utilizador funde-se. Isto porque a avaliação da satisfação do utilizador é feita pela avaliação de comportamentos e sentimentos no âmbito da interação com a aplicação.

No âmbito deste projecto penso que a métrica de Schneiderman (1998) é mais adequada e prática, porque estamos a recolher valores numéricos, que são mais fáceis de comparar do que avaliações qualitativas. Por exemplo, através

da comparação dos tempos de utilização entre utilizadores novos e utilizadores experientes, conseguimos ter a noção, pela disparidade dos valores obtidos, se é fácil ou não para o utilizador desempenhar determinadas tarefas, ou aprender a utilização do sistema.

## 4. PoeTryMe: a base do Co-PoeTryMe

O PoeTryMe (Gonçalo Oliveira, 2012) é uma sistema computacional capaz de gerar poesia automaticamente, em português, espanhol ou inglês, através da definição de um conjunto de dados de entrada, e usa um conjunto de coleções de poemas para definição da gramática, bem como uma rede que permite pesquisar relações semânticas entre palavras. Embora não seja possível, através da utilização da *interface* do TryMe<sup>5</sup> – *interface Web* que permita interagir com uma versão limitada do PoeTryMe (Figura 8) – alterar a gramática e a rede semântica, estes parâmetros podem ser alterados pelo utilizador. Aliás, basta que este altere o idioma de geração para que a gramática e a rede semântica sejam automaticamente alteradas. O utilizador pode ainda definir outros parâmetros, tais como a forma poética desejada, o tema do poema e a distância semântica entre as palavras-tema e as palavras escolhidas pelo sistema. Esta funcionalidade já pode ser testada através do TryMe, onde é possível definir estes últimos parâmetros através da interação com a sua *interface*.

“O sistema está assente numa arquitetura modular que permite desenvolvimento independente de cada módulo” (Gonçalo Oliveira, 2012), com o objectivo de ser “suficientemente versátil para permitir um alto nível de personalização. Para além disto, o PoeTryMe dispõe de uma API REST que permite que outras aplicações o usem enquanto serviço *Web*, isto é, através da comunicação sob a forma de pedidos ao sistema. No início deste trabalho, essa API tinha apenas a funcionalidade de geração de um poema completo, dada uma língua, um conjunto de palavras-chave, uma forma poética e um factor surpresa. No entanto, e visto que o PoeTryMe é o sistema que estará na base da nova aplicação, o Co-PoeTryMe, foi necessário adicionar funcionalidades à API, tais como a de geração de linhas, através da definição de um conjunto de palavras-chave e número de sílabas, e a de geração de palavras, através da definição de um relação com determinada palavra ou de um fonema final desejado ou de um determinado número de sílabas.

Nesta secção, o PoeTryMe é apresentado e enquadrado em algumas das métricas das apresentadas anteriormente, focando-se tanto no texto gerado como na presença de criatividade e no grau de co-criatividade. Serão ainda identificadas algumas limitações deste sistema, que motivaram a criação de uma nova aplicação co-criativa “em cima” do PoeTryMe.

---

<sup>5</sup><http://poetryme.dei.uc.pt/>

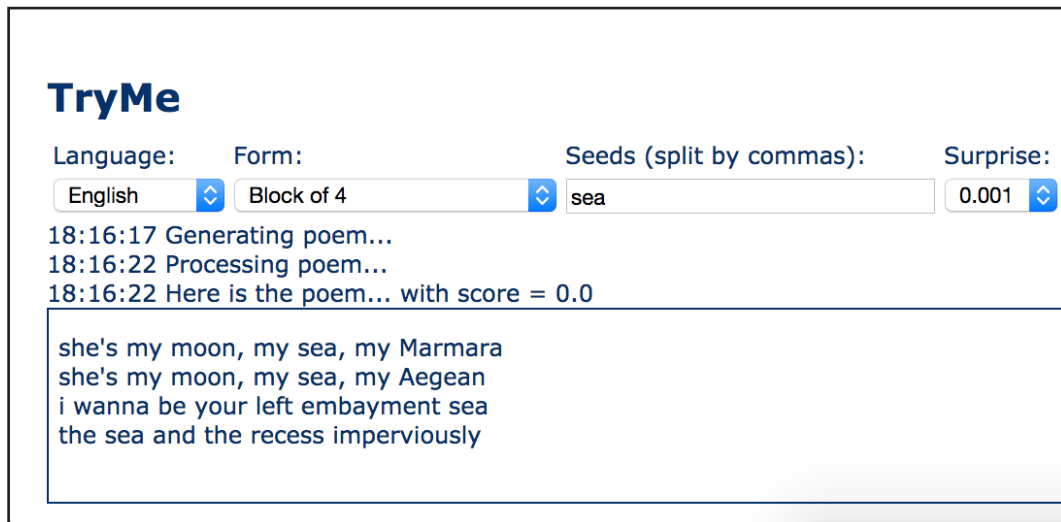


Fig.8- Screenshot da interface do TryMe.

## 4.1 PoeTryMe como gerador de poesia

Numa primeira fase vai ser descrita e avaliada a abordagem do sistema no que diz respeito à geração de poesia, e numa segunda fase serão avaliados os resultados gerados pelo sistema.

A arquitetura versátil do PoeTryMe permite a utilização de diferentes estratégias de geração. Gonçalo Oliveira (2012) refere três estratégias de geração possíveis:

**Básica:** Para cada linha do poema é gerada uma frase aleatória, usando os termos-chave do tema escolhido pelo utilizador.

**Geração e teste:** Para cada linha a ser criada, o sistema gera  $n$  linhas aleatórias e avalia-as de acordo com a métrica alvo e a presença de rima. Os melhores versos são usados, e os outros armazenados com a possibilidade de serem usados, caso seja necessário um novo verso com as mesmas restrições.

**Evolucionária:** Uma população inicial de poemas é gerada e avaliada, e os mais pontuados seguem para a próxima fase. A cada fase, os melhores poemas são combinados e transformados, gerando novos poemas candidatos. Na última fase de geração o melhor poema gerado é usado e apresentado como resultado.

Estas estratégias tiram partido de outro módulo crucial do sistema, que é o de geração de frases sintaticamente corretas e semanticamente coerentes. Isto é possível através da consulta de uma rede semântica e uma gramática. Resumidamente, a estratégia de geração recolhe as linhas criadas pelo módulo de geração de linhas e seleciona as que irão ser usadas no poema. A estratégia mais usada é a de geração e teste, devido à sua simplicidade e ao facto de não ter impacto negativo nos resultados ou no tempo de geração (Figuras 9 e 10).

de repente na praia numa margem  
canto à personagem do meu sol  
que se fez asa, oceano, mar  
orla atear, a praia aumentar

**Fig.9** - Quadra gerada pelo PoeTryMe a partir das sementes “mar”, “praia” e “sol”, com o grau de surpresa igual a 0.

virgindade e morte num humor  
foi álgico como a minha dor  
e o poder, o estado perdido  
personificação do meu cupido

este medo de emoções paradas  
humores poetas, fleumas delicadas  
amor é orgulho, amor é brio  
são silêncios de tranquilidade

de expressões sumptuosas e rostos  
visto que brios foram sentimentos  
sentimentos ou pressentimentos

esta sensibilidade são dois gostos  
e as dignidades são de orgulho  
se o filé é igual a palpito

**Fig.10** - Soneto gerado pelo PoeTryMe a partir das sementes “amor” e “sentimento”, com o grau de surpresa igual a 0.5.

Se analisarmos os poemas gerados pelo PoeTryMe e os avaliarmos de acordo com os parâmetros de Manurung (2003), percebemos que o tanto os requisitos de métrica como gramaticais são cumpridos. A atribuição intencional de significados é o factor que pode ser mais discutível, no entanto, o sistema pesquisa e escolhe palavras do mesmo domínio semântico do tema inserido pelo utilizador. A única questão que se pode colocar é a intencionalidade no uso de linguagem figurativa, que é a característica mais difícil de avaliar, devido à sua subjetividade. Como os poemas são construídos verso a verso, a coesão entre os eles pode não ser a mais desejada. No entanto, a construção de uma plataforma co-criativa, o Co-PoeTryMe, pretende, entre outras coisas, resolver essa lacuna. Assumindo o utilizador o papel de validar o significado dos poemas gerados, esta discussão deixará de existir.

## 4.2 PoeTryMe como sistema criativo

Os poemas gerados pelo PoeTryMe baseiam-se em templates, extraídos a partir de textos originais, onde duas palavras semanticamente relacionadas são substituídas por outras duas, com uma relação análoga. Esta análise é feita através da consulta de uma rede semântica e realizada no módulo de geração de linhas. A partir deste conjunto inicial de artefactos o sistema desempenha tarefas semelhantes às de combinação e transformação, descritas tanto por Boden (2003) como por Gero (2000).

No que toca à avaliação dos artefactos gerados, e recorrendo à métrica de Dan Ventura (2016), o sistema cria artefactos que apresentam novidade, valor e intencionalidade. Isto porque, em primeiro lugar, há sempre um distanciamento entre os poemas de inspiração e os artefactos gerados, logo os poemas produzidos apresentam novidade. Visto que o sistema têm a preocupação de fazer cumprir as regras sintáticas do idioma em que opera, pretende cumprir determinada métrica, e pontua a presença de rima, então os poemas gerados apresentam valor e relevância para os praticantes do domínio. Também podemos dizer que o sistema tem intencionalidade porque seleciona palavras do mesmo campo semântico do tema inserido e as insere em contextos onde estavam palavras que partilhavam a mesma relação.

## 4.3 PoeTryMe e co-criatividade

Na aplicação actual podemos identificar uma componente de co-criatividade, no sentido em que o utilizador pode gerar poemas, com as mesmas características (ou não), n vezes. No entanto os papéis que o sistema e o utilizador assumem são muito diferentes, e têm diferentes graus de influência tanto no processo criativo como no artefacto gerado.

O utilizador restringe o domínio a operar pelo sistema antes de se iniciar o processo de geração. Esta restrição é feita através da definição de um conjunto de parâmetros: o tema; a forma poética; o idioma; o nível de distância semântica entre as palavras-tema inseridas e as palavras a serem escolhidas pelo sistema. Na actual aplicação, tanto o conjunto de textos inicial, como a rede semântica e a estratégia de geração já estão definidas. É evidente que caso mudemos o idioma do poema a ser gerado, o conjunto de textos inicial e a rede semântica mudam. Apesar da arquitetura versátil do sistema permitir alterar estas e outras componentes, isso não está disponível através da aplicação TryMe. Para além disso, apenas o sistema gera conteúdos e, embora seja capaz de avaliar os artefactos gerados, fá-lo através de uma avaliação incompleta, visto que apenas pontua a métrica e a rima. Portanto, podemos dizer que o utilizador assume o papel de avaliador final no que toca à linguagem utilizada.

Do ponto de vista da co-criatividade, podemos dizer que esta componente está pouco explorada na aplicação actual, e é exactamente esse o objectivo deste projecto. Isto porque, embora o utilizador tenha influência na geração, através da introdução dos parâmetros iniciais acima referidos, não existe a possibilidade de interação com o poema gerado.

## 4.4 Limitações do PoeTryMe

Nesta secção vão ser indicadas algumas limitações do sistema que motivaram o desenvolvimento de uma aplicação co-criativa “em cima” dele. Em primeiro lugar, e visto que a aplicação é acedida através de uma *interface Web*, é necessário que as respostas sejam rápidas porque, entre outras razões, é importante que os utilizadores recebam feedback das suas acções de forma rápida, quase instantânea. Caso contrario, alguns das limitações poderiam ser minimizadas por usar todas as capacidades do PoeTryme, por exemplo, o aumento do número de fases

de geração conduz normalmente a melhores resultados e aumenta as probabilidades da existência de rima. No entanto, as limitações que vão ser apresentadas referem-se à atual implementação do PoeTryMe.

A principal limitação do PoeTryMe prende-se com a transmissão clara de uma mensagem conceptual. Embora o sistema pesquisa e escolhe palavras do mesmo domínio semântico do tema inserido pelo utilizador, como as linhas são geradas individualmente, muitas das vezes a coesão entre elas não é a melhor, ou a sequência de ideias apresentada não é a mais interessante. Por outro lado, algumas das palavras usadas podem ter mais do que um significado, o que por vezes pode ser interessante, mas muitas vezes altera o significado do poema. Esta limitação é a principal motivação para o desenvolvimento do Co-PoeTryMe, pois a intervenção humana, neste caso, ajudaria a conferir um significado ao poema. Esta foi também a opinião de alguns utilizadores, entre eles músicos e poetas, que expressaram a vontade de ter um papel mais ativo no processo de criação do PoeTryMe, e alguns deles confessaram mesmo ter gerado vários poemas para, a partir das melhores linhas, criarem novos poemas, mais direcionados para as suas necessidades e intenções.

Outra “limitação” prende-se com o facto do factor de surpresa ser um parâmetro difícil de definir. Se por uma lado usarmos um factor de surpresa muito baixo, é muito provável que pelo menos uma palavra de inspiração esteja presente em cada linha, mas se usarmos um factor de surpresa muito grande, começasse a perder a ligação entre o poema e as palavras-chave definidas. A intervenção humana, neste caso, pode avaliar quando um poema é muito literal ou é tão abstracto que não se percebe o tema que pretende transmitir.

Por fim, e devido ao método automático de renderização da gramática, ocasionalmente surgem inconsistências gramaticais nos poemas, ou palavras de outros domínios semânticos. Também neste caso, a intervenção humana é o suficiente para suprir esta lacuna.



## 5. Implementação do Co-PoeTryMe

O principal objectivo desta dissertação é fornecer aos utilizadores uma aplicação co-criativa de apoio à composição de poesia e letras de música, o Co-PoeTryMe, que lhes proporcione um elevado nível de personalização. Isto porque, tanto quanto se sabe, e analisando as plataformas identificadas na secção 2.3, não existe uma aplicação deste género suficientemente capaz e que dê inteira liberdade ao utilizador para interagir com o poema-rascunho após a geração. Esta aplicação usa o PoeTryMe, através da sua API, tanto para recorrer à ferramenta de geração de poemas, já presente na anterior aplicação, o TryMe, como à de geração de versos e palavras isolados e à de avaliação de poemas em relação a uma métrica e à presença de rima, que são funcionalidades desenvolvidas posteriormente por Gonçalo Oliveira, com o objectivo de serem incluídas no Co-PoeTryMe. Nesta secção vai ser explicado o levantamento de requisitos funcionais, isto é, as funcionalidades necessárias para que o sistema consiga proporcionar ao utilizador o grau de personalização pretendido.

Em segundo lugar, vão ser identificadas as tecnologias escolhidas para o desenvolvimento da aplicação Co-PoeTryMe, bem como as razões que levaram a essas escolhas. De seguida são apresentadas as funcionalidades desenvolvidas que possibilitam a resolução das tarefas que o Co-PoeTryMe se propunha a resolver.

### 5.1 - Requisitos Funcionais

O Co-PoeTryMe deve usar o serviço PoeTryMe, através da sua API REST, para permitir que o utilizador gere um poema-rascunho através da introdução de algumas restrições: idioma (português, inglês ou espanhol), forma poética (pode ser escolhida uma das formas disponíveis, ou introduzida uma personalizada, através da definição do número de linhas e número de sílabas por linha desejadas), sementes (palavras de inspiração que definem o tema do poema) e o grau de surpresa (distância semântica entre as palavras de inspiração e as geradas) – **requisito 1**. A utilização deste serviço pretende, numa fase inicial, resolver o síndrome da “folha branca” (Kantosalo et al., 2015), que se refere à dificuldade sentida pelos autores no início do processo de escrita de um texto criativo. No entanto, caso o utilizador não sinta esta dificuldade, o sistema deve permitir que este opte por escrever diretamente um poema-rascunho – **requisito 2**, ou por importar um poema – **requisito 3**, desde que esteja gravado num ficheiro “.txt”.

Após a geração do poema-rascunho, deve ser fornecido um conjunto de funcionalidades ao utilizador, que lhe permitam moldar e personalizar o poema à sua vontade. À semelhança do “The Poetry Machine”, o poema-rascunho é gerado sob a forma de fragmentos poéticos, isto é, cada palavra ou linha é um fragmento que pode ser editado individualmente. Esta técnica de apresentação do poema fornece ao utilizador um alto nível de personalização. O sistema deve permitir vários tipos de interação com estes fragmentos:

O sistema deve permitir que o utilizador **troque a posição de uma palavra com a de outra**, tanto para palavras da mesma linha como de linhas diferentes, e permitir que o utilizador **troque a ordem das linhas** do poema-rascunho – **requisito 4**;

O sistema deve permitir que o utilizador **edite uma palavra ou linha diretamente**, isto é, é possível o utilizador apagar ou escrever uma nova palavra ou linha, através da introdução directa de texto – **requisito 5**.

Para além das ferramentas que podem ser utilizadas sobre os fragmentos poéticos existentes, o sistema deve permitir a **geração de novos fragmentos**, tanto do tipo palavra como do tipo verso. Para isso, o Co-PoeTryMe recorre mais uma vez ao PoeTryMe enquanto serviço, desta vez para utilizar as ferramentas de geração de palavras e de versos, que podem ser integrados, ou não, no poema-rascunho.

No caso da geração de palavras – **requisito 6** – é possível definir um conjunto de parâmetros de maneira a restringir e direccionar o processo de geração:

**Relação com determinada palavra**, isto é, o utilizador pode introduzir ou escolher uma determinada palavra do poema-rascunho, e pedir que o sistema sugira palavras que se relacionam com a dada, sendo que a relação pode ser uma das disponíveis: sinónimo, antónimo, hiperónimo, hipónimo, co-hipónimo, ou qualquer uma delas;

**Rima desejada**, que na prática significa obrigar que as palavras sugeridas pelo sistema contenham determinado fonema no final. Esta funcionalidade ajuda os utilizadores a construir poemas com rima, que é uma das características particulares deste estilo literário;

**Sílabas Alvo**, ou seja, o número de sílabas que as palavras sugeridas pelo sistema devem ter. Esta funcionalidade ajuda os utilizadores a cumprir a métrica que desejam para o poema;

**Polaridade**, isto é, definição da conotação das palavras sugeridas pelo sistema, que pode ser positiva ou negativa. Esta ferramenta é útil para ajudar o utilizador a atribuir um sentimento ao poema.

Estas **restrições podem ser combinadas** para limitar mais o processo de geração, no entanto, pode-se correr o risco de o sistema não encontrar palavras que cumpram os requisitos definidos.

Também na geração de linhas – **requisito 7** – o utilizador deve poder definir um conjunto de parâmetros que direcionem o processo de geração para os resultados pretendidos:

**Número de sílabas**, que as novas linhas devem ter, factor que ajuda o utilizador a cumprir uma determinada métrica que deseje para o poema.

**Palavras-chave**, que são palavras introduzidas pelo utilizador que definem o tema das novas linhas a ser geradas. Esta funcionalidade ajuda o utilizador a conferir um determinado significado ao poema.

**O grau de surpresa**, que corresponde à distância semântica entre as palavras-tema inseridas pelo utilizador e a linha a ser gerada. Caso o grau de surpresa escolhido seja zero, é muito provável que uma das palavras-chave esteja presente na linha. Quanto maior o valor mais distante é a relação entre as palavras escolhidas e a palavra-chave (por exemplo se o grau de surpresa for 2, as palavras sugeridas podem ser sinónimos de antónimos de determinada palavra de inspiração inserida).

À semelhança da geração de palavras, estes **factores podem ser combinados** de maneira a restringir mais o processo de geração.

Tanto a geração de palavras como de linhas foram ferramentas desenvolvidas e adicionadas à API do PoeTryMe por Gonçalo Oliveira, no seguimento das reuniões realizadas no âmbito desta dissertação.

Os conteúdos gerados devem ser apresentados como fragmentos poéticos, embora com importância e destaque inferior aos do poema-rascunho. No entanto, tanto as palavras como linhas sugeridas podem ser integradas no poema-rascunho, e isto pode acontecer de várias maneiras:

O sistema deve permitir que o utilizador **troque a posição** de uma palavra ou linha **sugerida por uma** palavra ou linha **do poema-rascunho** – **requisito 8**.

Relativamente às palavras, deve ser possível que o utilizador **adicione uma das palavras sugeridas** a qualquer uma das linhas do poema-rascunho – **requisito 9**.

Em termos de linhas, o sistema deve permitir que o utilizador **adicione uma das linhas** sugeridas ao poema-rascunho – **requisito 10**.

O sistema deve providenciar um conjunto de **instruções de apoio à utilização da aplicação**, no entanto, o utilizador deve ter a possibilidade de as desativar, caso já esteja familiarizado com o seu funcionamento e não precise de as consultar – **requisito 11**. No entanto, estas ajudas devem estar acessíveis caso num determinado momento o utilizador sinta necessidade delas. Para além disso, o sistema deve permitir que o utilizador **desfaça ou refaça qualquer tipo de ação** sobre o poema, de maneira a convidar o utilizador à experimentação de várias possibilidades, sem sentir a responsabilidade de estar a tomar uma ação irreversível – **requisito 12**. Tanto as ferramentas de ativação/desativação de janelas de ajudas como de desfazer/refazer acções vêm ao encontro das regras de design de *interface* e interação apresentadas anteriormente, que defendem que o utilizador deve ter a liberdade e o controlo da utilização do sistema.

Deve ser possível que o utilizador **edite o poema como um todo**, caso sinta a necessidade de fazer uma alteração mais substancial – **requisito 13**. O sistema deve ainda permitir que, a qualquer momento, o utilizador gere, importe ou escreva um novo poema-rascunho, caso não esteja satisfeito com o actual.

O sistema deve permitir que o utilizador, a qualquer momento, possa **visualizar as alterações feitas ao poema**, tanto através de uma **representação estática**, que mostra todas as alterações feitas em simultâneo, como de uma **representação dinâmica**, que permite identificar a ordem/sequência de acções tomadas sobre o poema – **requisito 14**. Estas representações devem também refletir o grau de intervenção do utilizador e do sistema no produto final.

A dada altura, se o utilizador achar que tem um poema que o satisfaz, o sistema deve permitir que este o **salve – requisito 15 – ou partilhe – requisito 16** – nas redes sociais. Caso o utilizador queira guardar o poema gerado num ficheiro local, o sistema deve permitir que este o salve tanto em ficheiro de texto (.txt), que lhe permite guardar o poema num formato editável que pode voltar a ser importado para mais interação, ou numa imagem que para além de conter o poema produzido, inclui a representação estática de todas as acções efectuadas sobre ele. Quanto à partilha, o sistema deve permitir que o utilizador publique o poema gerado nas redes sociais, neste caso numa das disponíveis: Facebook e Twitter.

O sistema deve ainda incluir uma funcionalidade para **pontuar o seguimento da métrica e presença de rima** no poema – **requisito 17**. Esta ferramenta tenta reforçar a componente co-criativa da aplicação, incentivando o utilizador a criar um poema que lhe agrada, por um lado, mas que por outro lado siga uma métrica desejada e apresente rima, características comuns no texto poético. Por fim, deve ser possível alterar o idioma da interface, dentro das possibilidades Português e Inglês, para que a utilização possa ser utilizada por mais pessoas e chegar a mais gente – **requisito 18**.

## 5.2 Escolhas tecnológicas

O Co-PoeTryMe é uma aplicação *Web* que pode ser acedida por um *browser*. Cada vez mais, a Internet está acessível a partir de todo o lado e, para além disso, a escolha deste meio como forma de aceder à aplicação, traz algumas vantagens no cumprimento dos seguintes requisitos:

- A aplicação pode ser acedida em qualquer lugar, desde que o dispositivo tenha acesso à Internet, mais propriamente à *Web*.

- A aplicação deve ser executada a partir de um *browser* e não necessitar da instalação de qualquer *software* adicional, eliminando assim tanto requisitos de memória do disco rígido como problemas de compatibilidade com os aparelhos que lhe acedem.

No âmbito deste projecto, são utilizadas as seguintes tecnologias, das quais é difícil actualmente fugir quando se trata do desenvolvimento de uma aplicação em ambiente *Web*:

**HTML:** para criar a estrutura do site, ou seja, a linguagem HTML é usada para definir os diferentes contentores/módulos e os respectivos conteúdos. Permite ainda hierarquizar conteúdos e especificar o tipo de conteúdo que cada módulo receberá, por exemplo: texto, imagens, navegações, links, etc.

**CSS:** para estilizar os conteúdos definidos no HTML, tanto a aparência estática, tais como dimensão, cor, etc., ou a animação, caso sejam necessários conteúdos em movimento. Também é utilizado para o desenvolvimento da fluidez da *interface* (adaptação a diferente ecrãs de computador).

**JavaScript:** podemos dizer que toda a parte funcional da aplicação está implementada em JavaScript, nomeadamente:

- Para **desenvolver a interatividade** com os conteúdos, através da definição dos seus comportamentos de acordo com o tipo de interação a que estes estão a ser sujeitos. Visto que o dispositivo de acesso à aplicação é o computador, as interações poderão ser: rato por cima (*mouseover*), *click*, arrastar e largar (*drag & drop*), duplo *click* (*double-click*). De forma a facilitar este trabalho, foram usadas as bibliotecas jQuery<sup>5</sup> e jQuery UI<sup>6</sup>. Foi ainda usada a Tooltipster<sup>7</sup>, que é um plugin do jQuery que permite adicionar janelas de ajuda a elementos, e garante a compatibilidade com todos os *browsers*.
- Para **comunicar com a API do PoeTryMe**, alojada num servidor do Departamento de Engenharia Informática<sup>8</sup>. A comunicação é feita através de AJAX, sob a forma de pedidos à API REST, e os resultados dos pedidos vêm sob forma de objecto JSON, como demonstra a Figura 11. Estas respostas são tratadas e actualizadas na *interface*.
- Para **recolher as características do aparelho** que está a aceder à aplicação: dimensões do ecrã; reconhecer se é um dispositivo móvel, e no caso de ser, identificar a orientação do aparelho (*portrait* ou *landscape*). Permite apresentar os conteúdos de forma a se adaptarem aos dispositivo em questão, através da atribuição dinâmica de regras de CSS.

---

<sup>5</sup><https://jquery.com/>

<sup>6</sup><https://jqueryui.com/tooltip/>

<sup>7</sup><http://iamceege.github.io/tooltipster/>

<sup>8</sup><http://poetryme.dei.uc.pt:8080/>

- Para **representar o processo co-criativo**, tanto no seu estado estático como dinâmico. No âmbito da realização desta tarefa foram usadas algumas bibliotecas, nomeadamente a Raphael.js<sup>9</sup> para simplificar o processo de desenhar vectores e garantir a compatibilidade com os diversos *browsers*; e a Raphaels.handdrawn.js<sup>10</sup> que permite introduzir um grau de erro/imprecisão no desenho dos traços e corre no topo da Raphael.js.
- Para **exportar o poema para texto, imagem ou para o partilhar**. No caso da exportação para texto foi usada a FileSaver.js<sup>11</sup>, biblioteca que permite gravar ficheiros do lado do cliente, ideal para aplicações *Web* que precisam de gerar ficheiros. Para a exportação de imagens são usadas técnicas que permitem transformar um canvas numa imagem para que o download possa ser feito. Para a partilha nas redes sociais, neste caso Facebook e Twitter, são usados os plugins de “Share Button” e “Tweet Button”, respectivamente.
- Para **guardar dados localmente**, isto é, recorrer à API localStorage para armazenar dados que, por ficarem acessíveis mesmo depois de fechar a aplicação, melhoram a experiência de utilização. Neste caso são armazenados os dados do idioma escolhido para a *Interface* e do estado das janelas de ajuda. Isto permite que o utilizador, num segundo acesso à aplicação, encontre algumas definições iguais à sua anterior utilização.

---

<sup>9</sup><http://dmitrybaranovskiy.github.io/raphael/>

<sup>10</sup><http://handdrawn.clearcove.ca/>

<sup>11</sup><https://github.com/eligrey/FileSaver.js/>

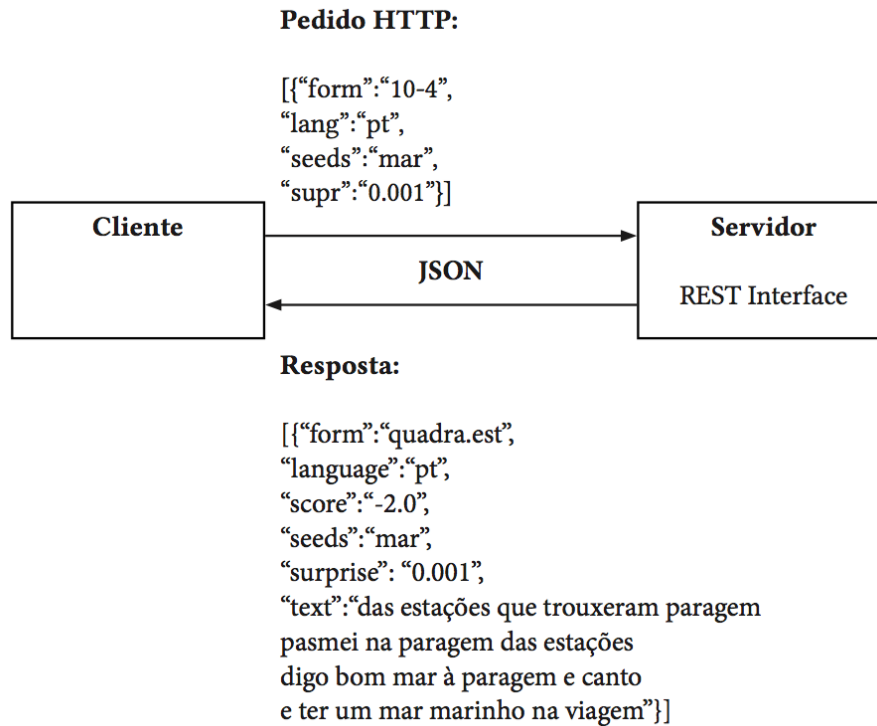


Fig.11 - Exemplo de um pedido HTTP.

## 5.3 - Funcionalidades Implementadas

Todos os requisitos (secção 5.1) foram implementados recorrendo as tecnologias referidas anteriormente (secção 5.2). Esta secção detalha a implementação dos vários requisitos.

### Gerar Poema-rascunho: requisito 1

Esta funcionalidade envolve, em primeiro lugar a disponibilização de um conjunto de opções para o utilizador preencher, que na prática são os parâmetros que o sistema PoeTryMe leva em consideração na geração de poemas completos: Idioma, Forma Poética, Grau de Surpresa e Palavras-Chave. No âmbito do desenvolvimento desta funcionalidade foi desenvolvida uma sub-funcionalidade, que permite introduzir formas poéticas personalizadas, através da introdução do número de linhas e número de sílabas por linha desejados. A funcionalidade de geração de poemas-rascunho usa nos dados inseridos pelo utilizador e faz um



pedido à API do PoeTryMe. Quando a resposta chega, esta funcionalidade está encarregue de receber o poema em formato de texto e transformá-lo em fragmentos poéticos, para que o utilizador possa interagir com ele.

### **Escrever Poema: requisito 2**

Esta funcionalidade permite que o utilizador escreva um poema-rascunho de raiz, através da disponibilização de uma área de escrita de texto. Após confirmação, a funcionalidade recolhe o texto inserido pelo utilizador, e à semelhança da geração de poemas, representa-o sob a forma de fragmentos poéticos interativos.

### **Importar Poema: requisito 3**

Esta funcionalidade permite que o utilizador carregue um poema-rascunho através da escolha de um ficheiro de texto (formato “.txt”) do seu computador. Após a escolha do ficheiro, a aplicação abre-o e lê o seu conteúdo. Posto isto, à semelhança das funcionalidades anteriores, esta ferramenta representa o texto do ficheiro sob a forma de fragmentos poéticos interativos. Esta funcionalidade permite também continuar o processo de interação com um texto, caso o poema carregado seja o resultado de uma exportação de texto feita anteriormente.

### **Trocar a posição de fragmentos poéticos no texto: requisito 4**

Esta funcionalidade permite que o utilizador, através de *drag & drop*, troque a posição de fragmentos poéticos que integram o poema-rascunho, sejam eles versos ou palavras. Isto acontece através de arrastamento de um fragmento poético para “cima” de outro. Assim que a troca é feita, o sistema armazena-a e representa o novo poema-rascunho. Sempre que um elemento é largado numa zona onde nenhum fragmento poético “habita”, este retorna à sua posição inicial.

### **Editar Fragmento Poético ou Poema-Rascunho: requisitos 5 e 13**

Esta funcionalidade envolve três possibilidades muito semelhantes: edição de palavras, edição de versos e edição de poema. Embora o processo de transformação destes três tipos de elementos textuais não seja exatamente igual, é muito semelhante e por isso aparecem agrupados. Esta funcionalidade permite que, através de *double-click* ou *enter* após um elemento estar selecionado, o utilizador edite ou elimine o conteúdo de determinado fragmento poético ou do poema-rascunho como um todo. Isto é possível através da disponibilização de uma área de introdução de texto, cujo conteúdo, após confirmação do utilizador, é formatado e representado sobre a forma de fragmentos poéticos interativos.

### **Gerar Palavras: requisitos 6, 8 e 9**

Esta funcionalidade é muito semelhante à de geração de versos, no entanto, a geração de palavras envolve a definição de um conjunto diferente de parâmetros: Palavra Alvo, Relação (com a palavra alvo), Rima, Sílabas Alvo e Polaridade. Os dados são enviados sobre a forma de pedido à API do PoeTryMe e as respostas são representadas sobre a forma de fragmentos poéticos interativos, que podem ser adicionados ao poema-rascunho, ou podem ser usados em substituição de alguma palavra pertencente ao poema-rascunho.

### **Gerar Versos: requisitos 7, 9 e 10**

Esta funcionalidade, em primeiro lugar, envolve a disponibilização de um conjunto de campos para o utilizador preencher, que correspondem aos parâmetros aceites pelo PoeTryMe para gerar versos, que são: Número de Sílabas, Grau de Surpresa, e Palavras-chave. Estes dados são formatados para terem a forma de pedido para a API do PoeTryMe. Quando a resposta chega, esta funcionalidade está encarregue de transformar as strings retornadas em fragmentos poéticos que podem ser adicionados ou substituídos pelos fragmentos que pertencem ao poema-rascunho.

### **Ativar/Desativar janelas de ajuda: requisito 11**

Esta funcionalidade permite que o utilizador ative ou desative as janelas de ajuda, o que na prática consiste em atribuir aos elementos da *interface* o atributo HTML `title`. No caso de as janelas de ajuda serem ativadas, ao passar com o rato em cima de cada elemento da *interface*, o conteúdo desse elemento, uma descrição textual da sua funcionalidade, é apresentado. Caso seja desativado, todos os conteúdos das janelas de ajuda passam a vazio, o que faz com que estas não apareçam. O estado deste parâmetro também é guardado no *browser*, neste caso um valor numérico que indica se na última visita através do *browser* à aplicação foram ou não usadas as janelas de ajuda.

### **Desfazer e Refazer alterações ao poema: requisito 12**

Estas funcionalidades são apresentadas como um conjunto visto que são muito semelhantes. A primeira permite ao utilizador desfazer uma alteração anteriormente feita, e a segunda permite recuperar um alteração que tenha sido desfeita. Estas funcionalidades envolvem identificar o tipo da última alteração feita e recuperar os conteúdos do estado anterior. Isto é feito através da consulta de um objecto que guarda todas as alterações feitas ao poema, que no caso de desfazer ação, elimina a última posição deste objecto. No caso de re-fazer ação, é consul-

tado um outro objecto que guarda todas as trocas que foram desfeitas, e caso haja alguma, o sistema adiciona esta alteração ao objecto que armazena todas as alterações feitas.

#### **Representar o processo co-criativo: requisito 14**

Para que esta representação seja possível, é necessário que toda a informação sobre as interações feitas ao poema sejam guardadas num objecto (o mesmo utilizado para desfazer e refazer acções). A visualização desta informação pode ser feita de duas maneiras: em primeiro lugar, permite que o utilizador veja, sob a forma de um aglomerado, todas as alterações feitas ao poema-rascunho; em segundo lugar, permite visualizar, sob a forma de animação, todas as trocas feitas, desta vez individualmente.

#### **Exportar Poema: requisito 15**

Esta funcionalidade está, na prática, dividida em duas partes, visto que a exportação pode ser feita para um ficheiro de texto, onde apenas o texto é preservado, ou sobre a forma de imagem, onde também é representado o processo co-criativo. A primeira possibilidade de exportação envolve “pegar” no texto presente na zona de edição do poema e escrevê-lo num ficheiro de texto, que é automaticamente guardado no computador do utilizador. Este ficheiro é guardado num formato que permite a sua futura importação. A segunda possibilidade envolve fazer um *screenshot* da área de edição do poema da aplicação, neste caso com o modo de representação do processo co-criativo ativado.

#### **Partilhar Poema: requisito 16**

Esta funcionalidade permite partilhar o texto presente na zona de edição do poema no Facebook ou no Twitter. Resumidamente, consiste em “pegar” no poema-rascunho, transformá-lo numa *string* e usar os *plugins* disponíveis para partilha, neste caso o plugin do botão de partilha do Facebook e o plugin do botão Tweet do Twitter.

#### **Pedir Avaliação: requisito 17**

Esta funcionalidade permite que a qualquer momento o utilizador avalie o seu poema-rascunho em relação ao cumprimento da forma poética inserida e à presença de rima. Esta ferramenta pretende reforçar a componente co-criativa da aplicação e incentivar o utilizador a criar um poema que agrade ambas as partes (utilizador e sistema). Esta funcionalidade envolve, mais uma vez, a comunicação com o sistema PoeTryMe. É fornecido um conjunto de parâmetros de entrada,

neste caso: idioma, forma poética e o poema propriamente dito, e o PoeTryMe retorna um valor numérico que representa a sua avaliação do poema enquanto cumpridor da métrica desejada, e quanto à presença de rima. Este valor é mapeado para um valor mais amigável para o utilizador, neste caso para um valor de zero a cinco estrelas.

### **Alteração do Idioma da *Interface*: requisito 18**

Esta funcionalidade envolve a criação de um objecto que contem todos os elementos textuais presentes na aplicação nos dois idiomas disponíveis: Português e Inglês. Sempre que o botão de alteração é carregado, o objecto é consultado e todos os textos são substituídos pelos textos do novo idioma seleccionado. Esta funcionalidade envolve também recolher e guardar informação no localStorage, isto é, no *browser* do utilizador. Isto permite que, por exemplo, se na primeira visita o utilizador preferiu usar a aplicação em inglês, da próxima vez que a aplicação for consultada no mesmo *browser*, já esteja com os textos em inglês por definição.

## 6-Design da Aplicação

Nesta secção vão ser apresentadas, com mais detalhe, as escolhas feitas relativamente ao design da aplicação. Em primeiro lugar será apresentado um conjunto de requisitos visuais da aplicação para que esta seja intuitiva, isto é fácil de aprender e fácil de utilizar. Vai ainda ser explicado como foram levadas em conta as diretrizes de design de *interface* e interação recolhidas, e as inspirações que influenciaram as escolhas feitas no design. Estas escolhas foram feitas tanto para a estrutura da *interface*, como para a tipografia, para a paleta de cores e para os símbolos utilizados para representar as ferramentas, e ainda ao nível do design dos próprios fragmentos poéticos. Vão ser também explicadas as escolhas relacionadas com a interação, isto é, a forma como os diversos elementos reagem às várias possibilidades de interação com o utilizador. Pretende-se também criar uma identidade visual para o sistema, de maneira a tornar a aplicação facilmente reconhecível e identificável.

A aplicação tem ainda o objectivo de representar o processo co-criativo, isto é, mostrar as acções efectuadas sobre o poema-rascunho e o grau de intervenção humana e computacional no poema final. Nesta secção serão também explicadas as decisões tomadas para representar os diferentes tipos de acções sobre o sistema, tanto para a representação estática como para a dinâmica.

### 6.1- Requisitos Visuais

Foi definido um conjunto de requisitos visuais para a aplicação Co-PoeTryMe, de maneira à *interface* da aplicação cumprir determinados objectivos. Em primeiro lugar pretendia-se que a aplicação fosse intuitiva e fácil de aprender. Um dos factores que contribui para isso é a utilização de uma estrutura modular, onde cada módulo representa uma funcionalidade. Isto ajuda o utilizador a perceber onde se executa cada uma das tarefas possíveis.

Para além disso, o design da *interface* é limpo e minimalista, sem elementos visuais puramente decorativos, ou seja, que não tenham utilidade informativa ou que não sejam parte integrante da resolução de alguma tarefa. Outro factor que contribui para esta “limpeza” da *interface* é o facto de existir muito espaço em branco, que permite que o utilizador se foque nos conteúdos propriamente ditos. Para além disso, o facto dos conteúdos (símbolos, margens de módulos e tipogra-

fia) se auto-sustentarem visualmente, contribui para que haja uma unidade visual e uma coesão que facilita a concentração do utilizador nas tarefas. Também é reforçada a consistência visual através do tamanho dos módulos, desenhados sobre uma grelha de 32 colunas, o que ajuda a reforçar a unidade visual, acima referida. O facto da tipografia utilizada na *interface* ser geométrica é outro factor que contribui para esta coesão e “casamento” entre os elementos. A paleta de cores utilizada, também ela simplificada (dispõe de quatro cores), contribui para esta unidade visual. Ainda assim, esta paleta de cores é suficiente, para representar tanto os vários tipos de estados de módulos e botões, como para identificar a possibilidade de interação com estes conteúdos.

Outro factor que também contribui para a rápida aprendizagem da *interface* é o facto da simbologia utilizada já estar presente em muitas outras aplicações, o que faz com que os utilizadores se sintam familiarizados e consigam prever o resultado da utilização de determinadas ferramentas.

## 6.2 - Design da Interface

Neste capítulo vai ser explicada a abordagem ao design da *interface*, isto é, as decisões tomadas no que diz respeito ao aspeto visual estático da aplicação. Estas decisões foram tomadas tanto para a estrutura da aplicação, como para a tipografia e paleta de cores utilizadas, e ainda para o design dos pictogramas que representam as ferramentas e dos fragmentos poéticos. Explica-se ainda de que maneira foram levadas em conta as directrizes recolhidas no âmbito do design de *interfaces*. Para além disso será apresentada a nova identidade visual da nova aplicação, o Co-PoeTryMe, a partir da qual se desenvolveram todas as possibilidades de visualização dos elementos presentes na aplicação.

### 6.2.1 - Estrutura

A *interface* do Co-PoeTryMe apresenta uma estrutura modular, onde cada módulo representa uma funcionalidade/ferramenta diferente da aplicação. Em primeiro lugar, esta arquitetura inspira-se na arquitetura do sistema PoeTryMe, serviço que é utilizado para gerar conteúdos (poemas, versos e palavras) e para avaliá-los de acordo com o cumprimento da métrica e presença de rima. Visto que o principal objectivo deste projecto, na prática, consiste no desenvolvimento

de uma *interface* para a mais recente versão da API PoeTryMe, achei interessante o facto da arquitetura da *interface* se inspirar e tentar representar, de alguma forma, a arquitetura do sistema “por baixo” da *interface*. À semelhança do PoeTryMe, que está dividido em módulos que podem ser desenvolvidos e substituídos individualmente, também a *interface* do Co-PoeTryMe está dividida em módulos, bem definidos, cada um destinado à realização de uma tarefa. Também por isso optei pela utilização de contornos, que delimitam a área correspondente a cada módulo e identificam a ação que é possível realizar no mesmo. No entanto, olhando para a *interface* como um todo, todos os módulos encaixam, como peças de um puzzle, em torno do módulo crucial da aplicação: o da edição do poema.

O facto da arquitetura ser modular traz algumas vantagens, tanto em termos da implementação, como do “desdobramento” da aplicação. Ao nível da implementação, facilita a aplicação de técnicas responsivas. Embora neste momento a aplicação apenas se adapte aos diferentes ecrãs de computador, seria relativamente directo re-organizar estes módulos para criar uma *interface* otimizada para utilização através de dispositivos móveis. Relativamente ao desdobramento da aplicação também há várias vantagens. Em primeiro lugar, e tirando partido das capacidades do *browser* de mostrar e esconder informação (Lupton, 2014), é possível fazer com que alguns módulos, em determinado momento, sejam escondidos ou apresentados em tamanho reduzido, e que sejam expandidos através de interação, revelando a informação que contêm. Esta técnica vai de encontro à última diretriz apresentada na secção 3.1, que diz que a informação deve estar disponível para ser consultada sempre que o utilizador sinta necessidade, ao invés de o obrigar a decorar essa informação. Na Figura 12 é dado o exemplo dos dois estados possíveis do módulo “Instruções”, do lado esquerdo num estado reduzido, onde não é visível a informação que este módulo disponibiliza, e depois no estado expandido, onde toda a informação está visível. O módulo passa de um estado para o outro através de *click*, neste caso no próprio módulo. Para passar do estado expandido para o reduzido, basta fechar o módulo por carregar no botão “x” no canto superior direito do mesmo.

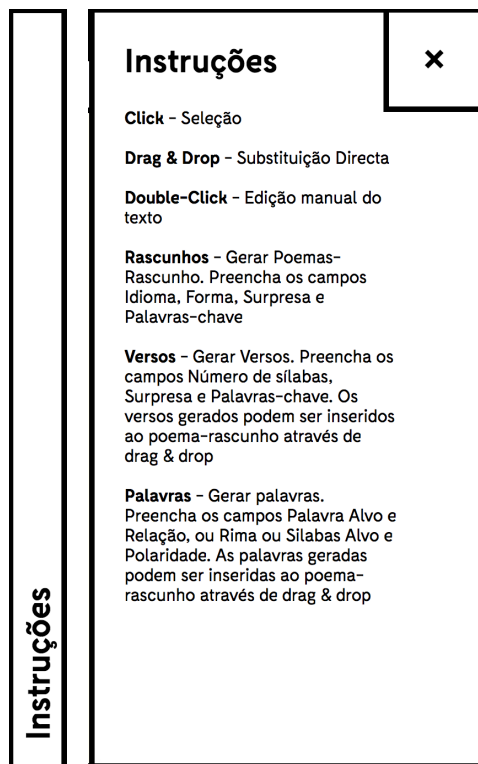
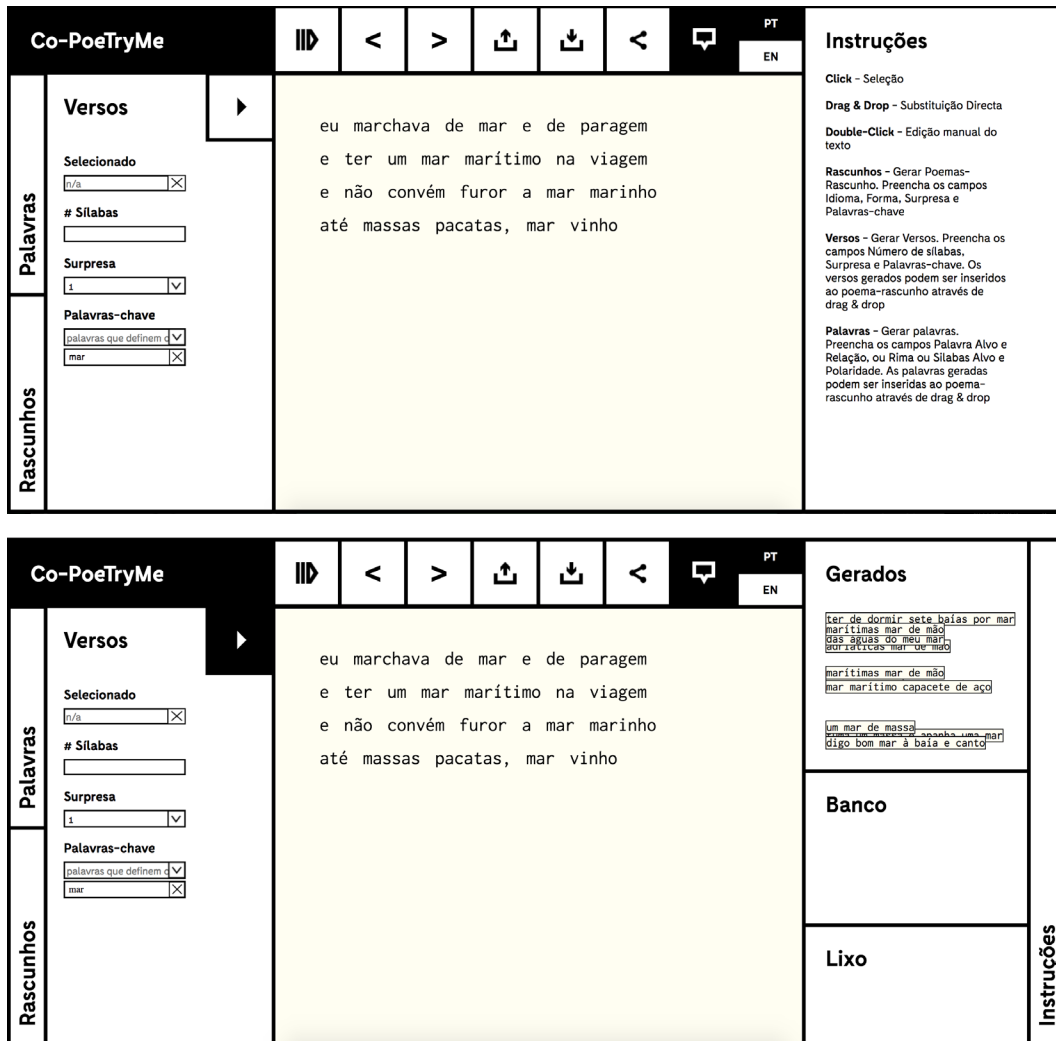


Fig.12- Módulo “Instruções” no estado reduzido, à esquerda e expandido, à direita.

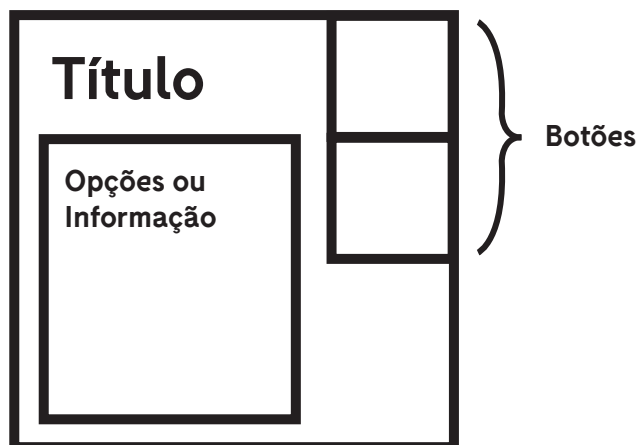
Para além disso permite que alguns módulos só apareçam quando as funcionalidades que fornecem estão disponíveis, ou em *feedback* a uma ação do utilizador. Isto faz com que a tarefa de criação de um poema seja dividida em sub-tarefas, e dá uma sensação de continuidade ao processo. Por exemplo, só é possível gerar versos ou palavras após a existência de um poema-rascunho, e por isso os módulos de geração de versos e palavras só são disponibilizados depois de haver um poema-rascunho. Também os módulos “Palavras/Linhas Geradas”, “Banco de Palavras/Linhas” e “Lixo” só são disponibilizados caso o utilizador tenha gerado palavras ou versos, isto porque não faz sentido estes módulos estarem presentes sem conteúdos, visto que não adicionam nenhuma funcionalidade ou possibilidade de interação se estiverem vazios (Figura.13). Caso fosse possível gerar estes conteúdos antes da existência de um poema-rascunho, a aplicação entraria num estado que levaria ao erro. Por isso, pode-se dizer que esta forma da aplicação se desdobrar ajuda a prevenir situações de erro, que é uma das diretrizes a levar em conta na construção de *interfaces*, apresentada na secção 3.1.





**Fig.13-** Numa primeira fase com o módulo Instruções expandido e sem a possibilidade de fechar. Após a geração de conteúdos, disponibilização de módulos que suportam interação com esse mesmo conteúdos.

Há um conjunto de características que se mantêm para todos os módulos: a posição do título, no canto superior esquerdo; os botões disponíveis, no canto superior direito; a posição das opções ou da informação que o módulo disponibiliza (Figura 14). Este factor facilita a utilização dos módulos porque permite que o utilizador “preveja” o comportamento de outros módulos após a utilização do primeiro. Isto vai de encontro à primeira directriz apresentada na secção 3.1 que diz que deve ser mantida a consistência para situações de utilização semelhantes.



**Fig.14-** Arquitetura dos módulos.

Outro factor que ajuda a manter a consistência entre os módulos é o facto de haver uma consistência nas dimensões utilizadas, tanto para os próprios módulos como para botões, títulos e textos. Foi criada uma grelha como base para o desenho da *interface*, neste caso assente em 32 colunas. Isto porque há situações onde os módulos são representados num estado muito reduzido (Figura 15), e por isso era necessária alguma diversidade e versatilidade na definição das dimensões dos elementos. Todos os botões têm a largura de duas colunas, enquanto que um módulo reduzido tem a largura de uma coluna. O módulo da Edição do Poema, visto ser módulo crucial, encontra-se centrado na página, ocupando um total de 16 colunas de largura, libertando assim 8 colunas para cada lado, para a representação dos módulos das ferramentas. Do lado esquerdo temos, inicialmente, o módulo de geração de poemas, que ocupa a totalidade das 8 colunas, mas que após a realização da tarefa a que se propõe, encolhe para uma coluna. Surge assim, por trás, o módulo de geração de linhas, que mede 7 colunas de largura. Este módulo pode ser substituído pelo módulo de geração de palavras, através do *click* no módulo reduzido “Palavras”, que também ocupa uma coluna, no lado esquerdo da aplicação. Estes dois módulos obrigaram a utilização de um número de colunas mais elevado, visto que o módulo expandido deve disponibilizar os seus componentes de forma clara e perceptível. Por isso tentou-se diminuir o tamanho dos módulos em estado reduzido para o mínimo possível, para não ocuparem espaço necessário à disponibilização das ferramentas do módulo expandido. O lado direito da aplicação foi tratado de forma semelhante ao lado esquerdo, mais uma vez com o objectivo de manter a consistência. Inicialmente temos o módulo Instruções a ocupar a totalidade das 8 colunas, que aquando da geração de Palavras ou Versos passa para um estado

reduzido, onde apenas ocupa uma coluna. Tanto os módulos de palavras/linhas geradas, como banco de palavras/linhas e lixo ocupam as 7 colunas restantes. Mais uma vez, devido ao facto de estes módulos poderem conter linhas, é importante que o espaço disponível seja suficiente para que uma linha sugerida não tenha que ser “partida” na sua representação, embora nalguns casos aconteça, nomeadamente quando são pedidos versos com muitas sílabas.

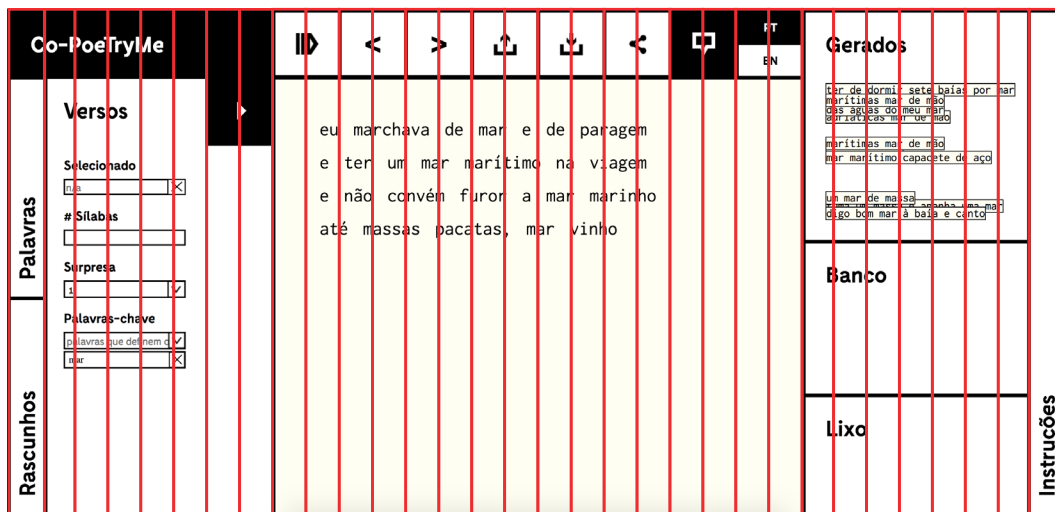


Fig.15- Grelha na base da construção da *interface* do Co-PoeTryMe.

## 6.2.2 - Identidade Visual

Devido ao objectivo de tornar a aplicação reconhecível, foi desenvolvida uma identidade visual para o projecto. Em primeiro lugar, foi definido o nome da aplicação: Co-PoeTryMe, que resulta da adição do prefixo ‘co’ (de co-criatividade, colaboração) ao nome da aplicação base, o PoeTryMe. Na prática, este projecto consiste no desenvolvimento de uma *interface* gráfica que permita interagir com a nova versão da API do PoeTryMe, tanto com a funcionalidade já de geração de poemas já existente, como com as novas funcionalidades de geração de palavras e linhas. Posto isto, pode-se dizer que a principal inovação do Co-PoeTryMe, em relação à aplicação anterior tem a ver com a introdução de uma forte componente co-criativa, isto é, a possibilidade dada ao utilizador de ter um papel activo no processo de criação do poema.

O logotipo é construído através da adição do prefixo “Co-”, representado num tipo de letra que remete para a escrita manual, ao nome da sistema existente “PoeTryMe”, desenhado num tipo de letra mais geométrico (Figura 16). A utilização destes dois tipos de letra tão diferentes pretende clarificar a grande novidade desta aplicação em relação à antiga, que é a componente co-criativa. Esta estética de escrita mais manual também foi introduzida na representação do processo co-criativo, visto que todas as alterações efectuadas ao poema-rascunho pelo o utilizador são apresentadas através de representações que também remetem para a escrita à mão. O logotipo aparece ao longo da aplicação tanto com fundo branco e tipografia a negro, como ao contrário (negativo), visto que o módulo que o contem dispõe de uma possibilidade de interação, e por isso, quando o rato lhe passa por cima, muda de estado.

## Co-PoeTryMe



Fig.16- Logotipo desenvolvido

O logotipo dispõe ainda de uma versão animada, mostrada logo quando o utilizador entra na aplicação. Numa primeira fase, o nome PoeTryMe é dividido em três módulos, “Poe”, “Try” e “Me”, apresentados de forma aleatória. Isto faz com que sejam formadas novas palavras/expressões, tais como: PoeMeTry, TryPoeMe, MePoeTry, etc. Numa segunda fase, é mostrada uma animação destes módulos a retornarem à sua posição original, formando o nome PoeTryMe. Após os módulos estarem na posição devida, o prefixo “Co-” aparece numa animação que pretende replicar a inserção de texto em qualquer editor de texto.

Dada a versatilidade da identidade visual desenvolvida, é possível aplica-la aos mais diversos objectos, como mostra a Figura 17.



Fig.17- Exemplos de merchandising com a marca Co-PoeTryMe

## 6.2.3 - Tipografia

Ao longo da aplicação são usados três tipos de letra diferentes, para situações diferentes: um para todos os títulos, textos e opções da *interface*, ou seja, para tudo o que são elementos estáticos da aplicação; um para os conteúdos gerados, sejam eles poemas, versos ou palavras, isto é, para todos os elementos que permitem interação; um para texto que é manualmente introduzido, que neste caso só aparece na fase de visualização do processo co-criativo, e pretende representar a introdução manual de texto.

A fonte utilizada para a *interface* é a Fabrik<sup>12</sup> (Figura 18), que é uma fonte desenhada por Fabian Fohrer, co-fundador da TIGHTTYPE, que tinha por objectivo criar uma fonte contemporânea com algumas formas de letra únicas. Isto porque algumas letras apresentam uma espécie de corte, o que faz com que este tipo de letra seja facilmente identificável. Na construção da *interface*, este tipo é usado tanto para títulos, como para opções e para informações, por isso é usado em tamanhos diferentes e nos dois pesos disponíveis: Regular e Negrito. Posto isto, a hierarquização dos conteúdos é feita através da utilização de diferentes pesos e tamanhos do mesmo tipo de letra, em vez da combinação de vários tipos diferentes. Em primeiro lugar, a fonte é usada para títulos, e por isso num tamanho

---

<sup>12</sup> <http://tighttype.com/fabrik/>

mais generoso, neste caso 30px e a negrito. Visto que esta fonte possui detalhes únicos, apresenta um grande grau de “espetacularidade”, um dos critérios da avaliação de fontes apresentado por Ellen Lupton, e vai de encontro à sua sugestão de utilizar fontes mais detalhadas e apelativas em tamanhos generosos. O peso e tamanho da fonte são utilizados de maneira a terem a mesma espessura dos contornos dos módulos, numa tentativa de que ambos trabalhem em conjunto na hierarquização dos conteúdos, mas também para conferir uma unidade e coerência visual à *interface*. A fonte é ainda usada para os títulos das opções e das informações disponibilizadas nos módulos, num tamanho de 18px e a negrito, e para os conteúdos das opcionais e informações propriamente ditas é usada a 18px no peso regular, ou seja, a este nível a hierarquização é feita através da alteração do peso da fonte.



Fig.18- *Speciemen* da fonte “Fabrik”.

Para os conteúdos gerados pelo sistema, é usada a Inconsolata<sup>13</sup> (Figura 19), fonte desenhada por Raph Levien com o objectivo de ser utilizada em ambientes de programação, ou seja, para ser vista em ecrã. Esta é uma fonte mono-espçada e por isso tem uma forte conotação tecnológica. Visto que os conteúdos são gerados por um agente computacional, faz sentido que o texto gerado tenha uma forte conotação tecnológica, já que pretende representar a sua comunicação com o utilizador. Esta fonte é usada, ao longo da aplicação, em dois tamanhos diferen-

<sup>13</sup><https://fonts.google.com/specimen/Inconsolata>

tes, sempre no peso regular. Num primeiro momento é usada num tamanho de 26px, para representar o poema-rascunho, embora este tamanho possa reduzir se as linhas do poema forem muito compridas e ultrapassarem o tamanho do módulo de edição do poema. Caso isso aconteça, o tamanho poderá reduzir até 18px, de maneira ao poema-rascunho poder ser visualizado na íntegra, ao invés de ter frações do poema-rascunho escondidas, só possíveis de visualizar através de *scroll*. Este tamanho pretende refletir a importância do módulo de “Edição do Poema”, o módulo principal da aplicação, e por isso, os fragmentos textuais nele presentes são os agentes principais da aplicação. Por outro lado, e visto que estes fragmentos permitem interação, o facto de estarem representados num tamanho mais generoso facilita essa tarefa, visto que, por exemplo, se tivermos a utilizar a ferramenta de *drag & drop*, torna as zonas de *drop* mais generosas. Num segundo momento a fonte é utilizada para representar versos e palavras geradas pelo sistema. Neste caso, a fonte é utilizada num tamanho mais pequeno, 18px, visto que estes fragmentos são de segunda importância, pois podem ou não ser integrados no poema-rascunho.

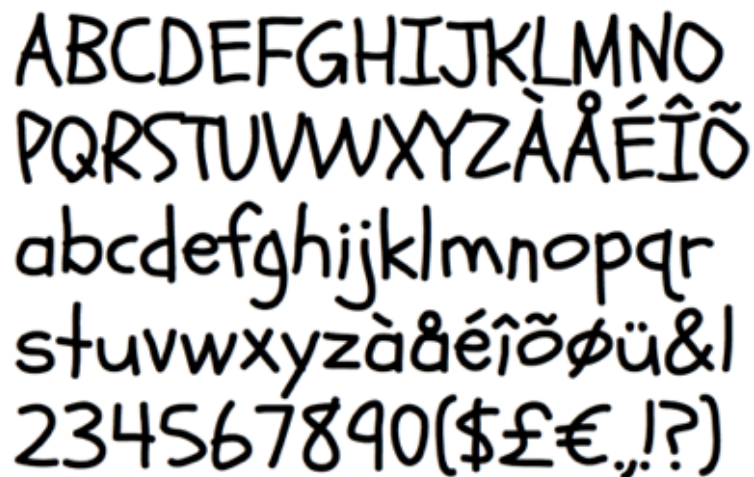
!"#\$%&'()\*+,-./0123456789:;<=>?  
 @ABCDEFGHIJKLMNPOQRSTUVWXYZ[\]^\_  
 `abcdefghijklmnopqrstuvwxyz{|}~  
 ¡¢£¥¦§¨ª«¬®¯°±²³´µ¶·¸¹º»¼½¾¿  
 ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞß  
 àáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ  
 ĀĂĄĆĈĊĎĚĜĹĽŃŇŔŖŠŢŤŨÚŹŻąăċďđęğ  
 κĹĺńňřśŝţţ'üűżη<>'','',...↑↓¼½¾Ⓜ

Fig.19- *Speciemen* da fonte “Inconsolata”.

Para representar o texto inserido manualmente pelo utilizador, é utilizada a fonte Schoolbell<sup>14</sup> (Figura 20), desenhada pela “Font Diner” e é um tipo de letra que pretende representar a escrita à mão. Foi introduzido um determinado grau de imprecisão no desenho das formas, que remete para as letras desenhadas por crianças ou à pressa. Visto que para a representação das trocas efectuadas foram

<sup>14</sup><https://fonts.google.com/specimen/Schoolbell>

utilizados traços que também incluem um determinado grau de imprecisão na sua representação, esta fonte liga-se perfeitamente com estas representações e ajuda os utilizadores a entender que os textos escritos nesta fonte pertencem às alterações feitas ao poema. Esta fonte é utilizada no mesmo tamanho que a Inconsolata, no poema-rascunho, para dar a entender que o texto é parte integrante do poema.



ABCDEFGHIJKLMNO  
PQRSTUVWXYZÀÁÊËÏ  
abcdefghijklmnoqr  
stuvwxyzàáêëïøü&  
234567890(\$£€.,!?)

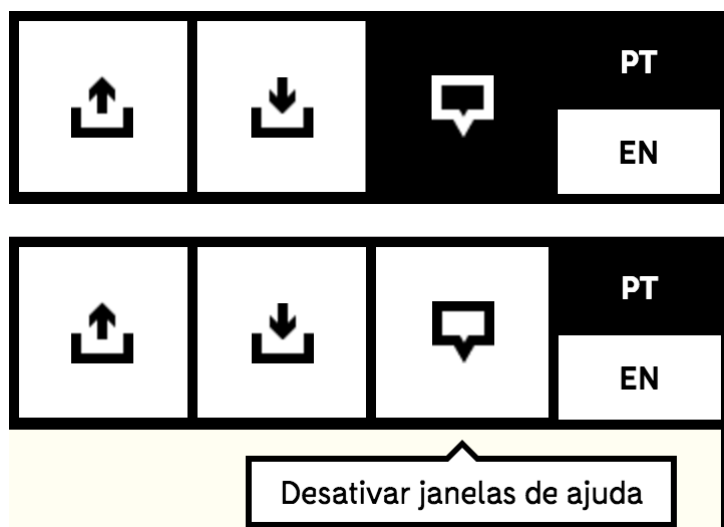
Fig.20- *Speciemen* da fonte “SchollBell”.

## 6.2.4 - Paleta de Cores

A paleta de cores utilizada na construção da *interface* dispõe apenas de quatro cores: Branco, Preto, Amarelo muito claro e Vermelho. A maior parte da aplicação é monocromática, e o uso apenas do preto e branco pretende maximizar o contraste entre texto e fundo, já que o agente principal desta aplicação é o texto. Esta situação, onde o contraste é máximo potencia a legibilidade do texto. A aplicação pretende representar o processo de escrita, neste caso de um poema, processo esse vulgarmente designado por “pôr o preto no branco”. Nas zonas estáticas da aplicação, o fundo é branco e a tipografia e contornos são pretos. No entanto, quando os módulos permitem interatividade, por exemplo quando estão no seu estado reduzido, as cores são trocadas (ficando o fundo a preto e a tipografia a branco, sendo que os contornos nunca alteram a sua cor) para ajudar o utilizador a perceber que há uma interação disponível. Também no caso dos



botões que têm dois estados, selecionado e não-selecionado, é usada esta técnica. Por exemplo, no caso do botão “Janelas de Ajuda”, caso as janelas de ajuda estejam ativadas o botão é representado em negativo, isto é, com fundo preto e o pictograma a branco, e quando estão desativadas apresentam fundo branco e pictograma a preto. Caso a funcionalidade do botão esteja ativada, quando o rato lhe passa por cima, a aparência troca para fundo branco e pictograma a preto para que seja perceptível que existe uma possibilidade de interação com o mesmo (Figura 21).

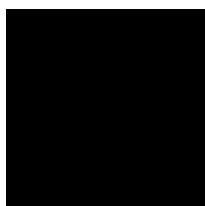


**Fig.21-** Interação com o botão Ativar/Desativar janelas de ajuda e o seu comportamento crómico em relação à interatividade

De maneira a que o módulo “Edição do Poema” ficasse destacado dos outros, a sua cor de fundo é ligeiramente amarelada. No entanto, esta diferença de cor é muito subtil. Isto porque, em primeiro lugar, um dos objectivos desta aplicação é resolver o síndrome da folha-branca, apontado por Kantosalo (2014) e que pode surgir no início do processo de escrita de um texto criativo. Por isso fazia sentido que a secção da aplicação que representa a folha fosse branca. No entanto, de maneira a destacar esta zona, é-lhe dada uma cor diferente, um amarelo muito claro, que lhe dá um aspeto de folha de papel reciclada. Visto que a reciclagem consiste, na prática, na utilização de determinados objectos/materiais para fazer outros materiais, achei o processo semelhante ao que ocorre neste módulo. Isto

porque os fragmentos, neste módulo, podem ser re-posicionados, editados ou apagados, formando assim novos poemas. Para além disso é possível, por exemplo, gerar palavras que usam uma das palavras do poema como base/restricção ao processo de geração, nomeadamente quando pretendemos gerar palavras com determinada relação com uma palavra do poema-rascunho. Posto isto, o amarelo atribuído a este módulo pretende, por um lado, destacar esta zona das restantes, e por outro lado, fazer uma comparação entre o processo de reciclagem e o de criação de um poema através da utilização da aplicação Co-PoeTryMe.

Para a representação das alterações feitas ao poema-rascunho por parte do utilizador é usada a cor vermelha. Visto que a representação das alterações feitas se inspira na fase de revisão da produção editorial, correções essas que são, geralmente, feitas à mão com um marcador vermelho, achei que a cor utilizada deveria reforçar esta inspiração. Esta cor é tanto utilizada para representar os traços e caixas que simbolizam trocas feitas na posição dos conteúdos do poema, bem como para representar o texto que foi introduzido directamente pelo utilizador.

**Preto:**

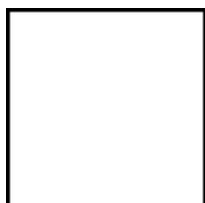
RGB - (0,0,0)

HEX - #000000

HSL - (0, 0%, 0%)

HWB - (0, 0%, 100%)

CMYK - (0%, 0%, 0%, 100%)

**Branco:**

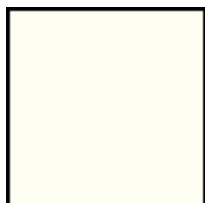
RGB - (255,255,255)

HEX - #ffffff

HSL - (0, 0%, 100%)

HWB - (0, 100%, 0%)

CMYK - (0%, 0%, 0%, 0%)

**Amarelo Claro:**

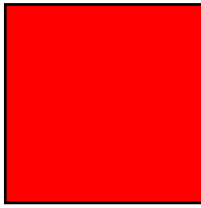
RGB - (255, 254, 242)

HEX - #fffef2

HSL - (55, 100%, 97%)

HWB - (55, 95%, 0%)

CMYK - (0%, 0%, 5%, 0%)



### Vermelho:

RGB - (255, 0, 0)

HEX - #ff0000

HSL - (0, 100%, 50%)

HWB - (0, 0%, 0%)

CMYK - (0%, 100%, 100%, 0%)

## 6.2.5 - Ferramentas

Ao longo da aplicação é apresentado um conjunto de pictogramas em representação de ferramentas disponíveis. Na Figura 22, todos estes botões aparecem acompanhados de um número, e a legenda é mostrada abaixo:

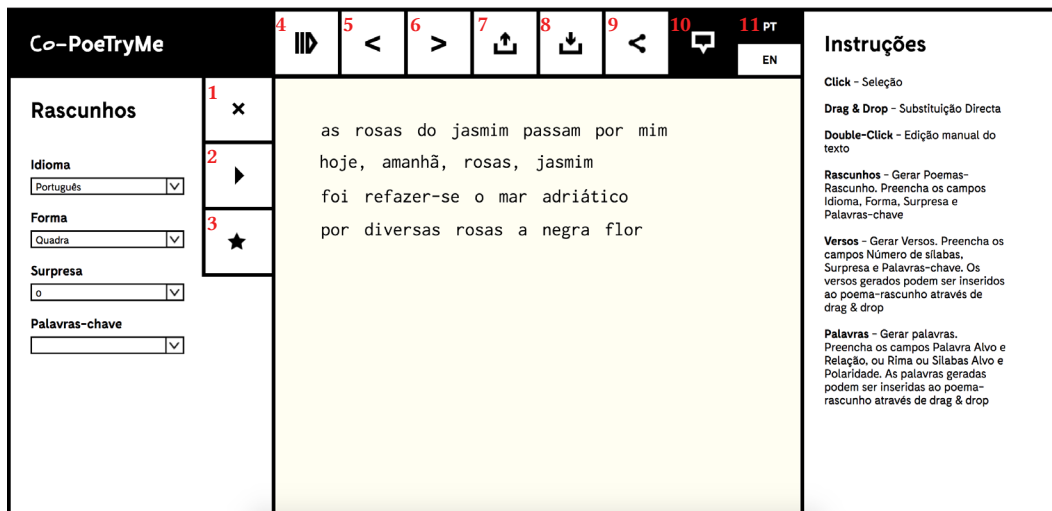
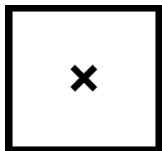


Fig.22- Screenshot da aplicação com a numeração dos botões.



Botão Fechar (1), presente em todos os módulos que tenham uma estado reduzido e expandido. A forma “x” é amplamente utilizada em aplicações para representar o fecho da actual janela, e por isso é a forma mais intuitiva para representar essa ação.



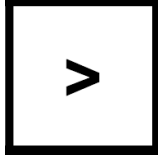
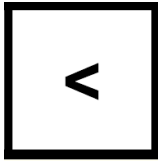
Botão Gerar (2), presente em vários módulos da aplicação e que serve para confirmar acções, na maioria das vezes no âmbito da geração de conteúdos. Tem a forma triangular de um botão *play*, que é uma forma muito utilizada noutros ambientes, mas em situações semelhantes, tais como em jogos de computador, para começar a jogar ou retomar um jogo em pausa, ou para iniciar a reprodução de um vídeo na maioria dos programas de reprodução de vídeo. Neste caso, o processo de escrita é representado sobre a forma de uma espécie de puzzle ou jogo, e o carregar neste botão traz novas possibilidades de interação com o poema (novas jogadas possíveis).



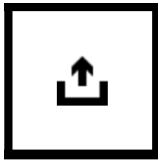
Botão Avaliar (3), presente no módulo Rascunhos, que fica disponível quando um poema-rascunho é gerado. Esta funcionalidade é similar à pontuação que a maioria dos jogos dispõem. Muitas vezes esta pontuação é atribuída numa escala de zero a cinco estrelas. Posto isto, e visto que os valores retornados por esta ferramenta são entre zero e cinco, a estrela é adequada à representação desta funcionalidade.



Botão Vizualizar Processo Co-Criativo (4), também presente na barra superior de ferramentas. Este é um botão rádio e a representação do seu estado é feita de igual modo ao botão de “Ativar/Desativar Janelas”. Esta funcionalidade é exclusiva desta aplicação, e por isso não é possível utilizar uma representação que já tenha vindo a ser usada em outras aplicações, e que por isso os utilizadores já estejam familiarizados. A seta tracejada utilizada pretende representar vários *frames* ou estados, juntamente com um botão *play*. Visto que o resultado de carregar neste botão permite ver as alterações feitas ao poema, numa primeira fase através de um aglomerado de todas as alterações, e numa segunda fase através de uma animação que mostra as alterações feitas individualmente, achei que este símbolo seria o mais indicado para representar esta ferramenta.



Botões Desfazer Ação (5) e Re-fazer Ação (6), presentes na barra de ferramentas, na zona superior da aplicação. Estas ferramentas são vulgarmente representadas por setas orientadas para sentidos contrários: no caso da de Desfazer Ação (5) é representada por uma seta a apontar para a esquerda, e aparece sempre a par com a de Re-fazer Ação (6), representada por uma seta a apontar para a direita. Visto que esta dupla de setas é amplamente usada no design de *interfaces*, julgo que elas são as formas que melhor representam estas ferramentas.



Botões Importar Poema (7) e Exportar Poema (8), também presentes na barra de ferramentas superior. À semelhança das ferramentas Desfazer e Re-fazer Ação, estas representações costumam aparecer juntas, para melhor se perceber o significado. No caso da ferramenta “Importar” a seta aponta para baixo, para a forma que pretende representar uma caixa, e na ferramenta “Exportar” a seta está a apontar para cima, ou seja para fora da caixa. Resumidamente estas setas pretendem representar os prefixos “in” e “ex”, onde “in” indica para dentro e “ex” indica para fora. Mais uma vez, estes símbolos já estão presentes em muitas aplicações, o que contribui para a familiarização dos utilizadores com eles.

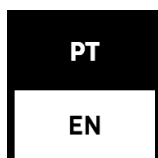


Botão Partilhar (9), também presente na barra superior de ferramentas. Este símbolo já é usado noutras aplicações para representar esta ação, no entanto há algumas representações alternativas, nomeadamente através de uma seta com uma curva para a direita. No entanto, julgo que este símbolo está mais relacionado com a ideia de partilha do que a seta, uma vez que os três nós ligados por linhas transmitem a ideia de comunicação.



Botão Ativar/Desativar Janelas (10), também apresentado na barra de ferramentas superior. Este tipo de botão é vulgarmente designado como “botão rádio”, e é um botão que tem duas posições possíveis. Através de interação com ele, neste caso de *click*, é possível trocar o seu estado. A alteração de estado é representada, como já referido na secção anterior, através da utilização da representação em positivo (fundo branco e símbolo a preto) e em negativo (fundo preto e símbolo a branco). Quanto à simbologia

utilizada, e visto que esta é uma funcionalidade que não está muito presente em outras aplicações e por isso não permite tirar partido de simbologia já amplamente usada, pretendi que fosse bastante literal. O símbolo representa um balão de fala, elemento amplamente utilizado em banda-desenhada, mas que neste caso não tem a forma circular mais comum, mas pretende representar a forma dos balões de ajuda utilizados na aplicação.



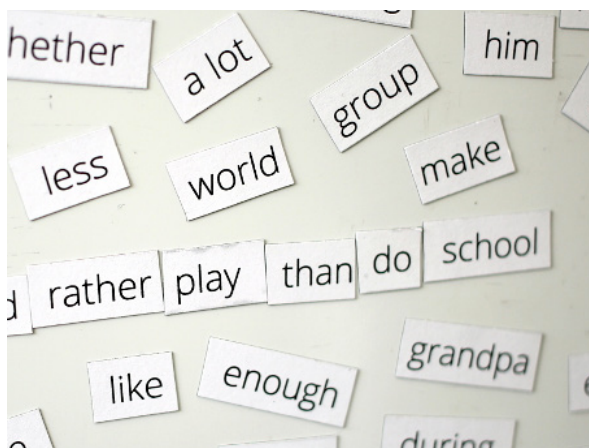
Botão Seleção de Idioma (11), também presente na barra de ferramentas superior. À semelhança do botão anterior, este também é um botão rádio, ou seja, dispõe de duas posições possíveis. No entanto, ao contrário do botão anterior, as duas possibilidades de estado estão visíveis e uma delas está a negativo, para indicar que é a opção que está ativa. Neste caso optei por esta abordagem, para informar o utilizador, *à priori*, dos idiomas disponíveis, em vez de o obrigar a clicar no botão para descobrir as outras possibilidades. Para representar os idiomas possíveis decidi utilizar a sigla de duas letras ao invés de criar símbolos de representação dos idiomas, que iriam dificultar a leitura e aumentar o nível de abstração. Este tipo de representação é amplamente utilizada em outras aplicações, *Websites*, jogos, etc..

## 6.2.6 - Fragmentos Poéticos

Tanto os poemas-rascunho, sejam eles gerados, importados ou escritos manualmente, como os versos e palavras, gerados ou inseridos manualmente, são representados através de fragmentos poéticos. Há dois tipos de fragmentos, versos e palavras. Desta maneira é possível interagir com linhas inteiras ou apenas com determinada palavra de uma linha. Existem também dois graus de importância para estes fragmentos: fragmentos principais, que são os que estão a ser utilizados no poema-rascunho e fragmentos secundários, que são os conteúdos gerados/sugeridos, que podem ou não ser incluídos no poema-rascunho. Tanto os versos como as palavras presentes no poema-rascunho permitem três tipos de interação: substituição de posição, edição do conteúdo e supressão. Os fragmentos que contêm os conteúdos gerados, sejam eles versos ou palavras, permitem quatro tipos de interação: substituição por um fragmento do poema-rascunho; adição ao poema-rascunho (no caso dos versos é possível adicioná-los ao final do

poema-rascunho, no caso das palavras é possível adicioná-las ao final de qualquer uma das linhas do poema-rascunho); podem ser arrastados para o módulo Repositório para que sejam preservados caso haja novas fases de geração de conteúdos; podem ser arrastados para o Lixo, local onde são guardados fragmentos que já pertenceram ao poema-rascunho, mas que foram substituídos por outros.

Posto isto, e visto que as interações disponíveis com os fragmentos dão a sensação de jogo/puzzle, era pretendido que a representação dos fragmentos também conferisse uma certa “jogabilidade” ao processo de escrita do poema. Por isso o design dos fragmentos é inspirado nos vulgares ímans que se colocam nos frigoríficos (Figura 23) e em puzzles. Inicialmente os elementos estão alinhados e sob a forma de texto, sem qualquer decoração. No entanto, quando o rato lhes passa por cima, eles revelam um caixilho, que pretende representar essa palavra ou verso como uma peça do jogo. Quando uma peça é selecionada esta muda de cor, neste caso fica em negativo, para que seja perceptível a alteração do seu estado. Também quando novos conteúdos são gerados, são representados sob a forma de peças, com o contorno bem definido. Esta similaridade na representação dos fragmentos do poema-rascunho e dos fragmentos sugeridos ajuda a que os utilizadores percebam que estas novas peças também podem ser “jogadas”, neste caso incluídas no poema-rascunho (Figura 24).



**Fig.23-** Módulos magnéticos com palavras, colados no frigorífico.

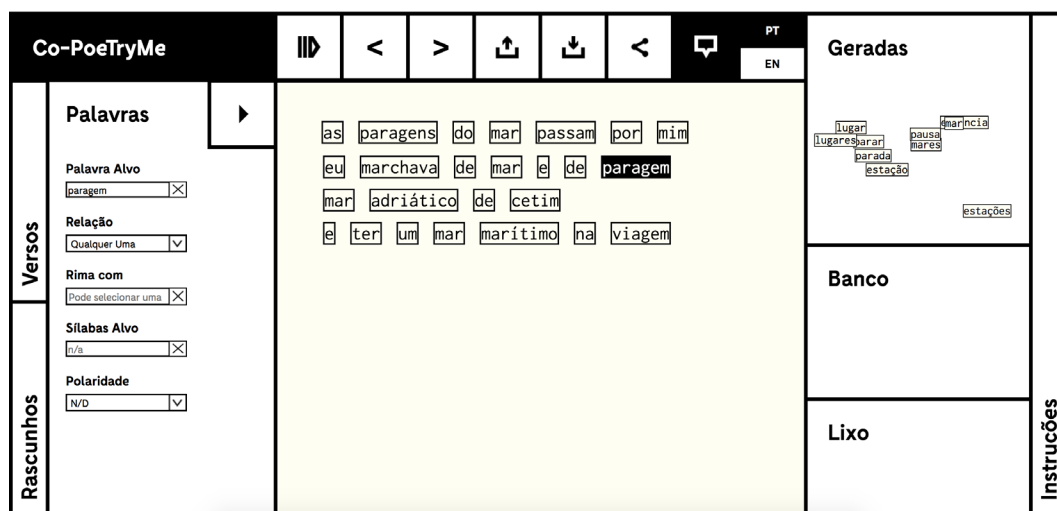


Fig.24- Fragmentos poéticos gerados como peças possíveis de jogar.

## 6.2.7 - Representação do processo co-criativo

Todas as alterações feitas ao poema-rascunho são guardadas para serem apresentadas ao utilizador, tanto através de uma representação estática como de uma dinâmica. Na estática, todas as alterações são mostradas através de um aglomerado, onde as trocas mais antigas têm menor opacidade do que as mais recentes. Na dinâmica é reproduzida uma animação, onde as alterações são mostradas individualmente. A linguagem visual utilizada é inspirada num conjunto de símbolos de revisão de provas, utilizados na fase da revisão da produção editorial, processo que permite que correctores avaliem e identifiquem erros antes da publicação final ser publicada. Embora não exista uma sinalética standardizada, há um conjunto de símbolos amplamente utilizados para identificar determinados erros, como mostram as Tabelas 2, 3, 4 e 5. Estes símbolos são desenhados à mão pelos revisores na versão da publicação que estão a analisar. A criação de uma linguagem visual inspirada nestes símbolos pretende ajudar os utilizadores a identificar e memorizar as acções efectuadas e tem a vantagem de ser uma linguagem universal, e por isso não precisa de ser traduzida para os vários idiomas.



Sinal	Significado	Exemplo	Mudar para
Y ou ---	excluir	errar é <del>h</del> humano errar é <del>não</del> humano	é errar humano
^	inserir	errar <sup>é</sup> humano	
□	palavras mudadas	é[errar] humano	
(sp)	não abreviar	[3° andar] <sup>sp</sup>	[terceiro andar]
(ab)	abreviar	[terceiro andar] <sup>ab</sup>	[3° andar]

**Tab. 2-** Símbolos de revisão de provas, relacionados com a gramática e ortografia.

Sinal	Significado	Exemplo	Mudar para
#	inserir espaço	errar é humano 	errar é humano
⌋	remover espaço	errar é humano 	
—< #	inserir espaço de linha	errar é humano errar é humano 	errar é humano errar é humano
—< √ #	remover espaço de linha	errar é humano errar é humano 	errar é humano errar é humano
[	alinhamento com esquerda justificada		_____
]	alinhamento com direita justificada		_____
]...[	centrado	errar é humano	_____
¶	mudar parágrafo	errar é humano 	errar é humano
√ ¶	não mudar parágrafo	errar é humano 	errar é humano

**Tab. 3-** Símbolos de revisão de provas, relacionados com a formatação do texto.

Sinal	Significado	Exemplo	Mudar para
ital	itálico	[errar é humano] <sup>ital</sup>	<i>errar é humano</i>
bf	negrito	[errar é humano] <sup>bf</sup>	<b>errar é humano</b>
↗	converter para maiúscula	<del>errar</del> é humano	Errar é humano
↘	converter para minúscula	<del>Errar</del> é humano	errar é humano
sc	converter para Small Caps	[ERRAR HUMANO] <sup>sc</sup>	ERRAR HUMANO
rom	converter para românico	errar é [ <i>humano</i> ] <sup>rom</sup>	errar é humano
wf	fonte errada	errar é [humano] <sup>wf</sup>	errar é humano
v	subscrito	H <del>2</del> O	H <sub>2</sub> O
^	sobrescrito	H <del>2</del> O	H <sup>2</sup> O

**Tab. 4-** Símbolos de revisão de provas, relacionados com a tipografia.

Sinal	Significado	Exemplo	Mudar para
⊙	ponto	errar é humano⊙	
⊙,	vírgula	errar é humano⊙,	é errar humano,
:	dois pontos	errar é humano <sup>:</sup> ^	errar é: humano
;	ponto e vírgula	errar é humano <sup>;</sup> ^	errar é; humano
?	ponto de interrogação	errar é humano? <sup>?</sup> ^	errar é humano?
!	ponto de exclamação	errar é humano! <sup>!</sup> ^	errar é humano!
⊖	hífen	errar é humano <sup>⊖</sup> ^	errar é-humano
'	apóstrofe	errar é humano' <sup>'</sup> ^	errar é humano'
“...”	citação	“errar é humano” <sup>“”</sup>	“errar é humano”
$\frac{1}{N}$	traço largura n “n-dash”	errar é humano <sup><math>\frac{1}{N}</math></sup>	errar é–humano
$\frac{1}{M}$	traço largura m “m-dash”	errar é humano <sup><math>\frac{1}{M}</math></sup>	errar é—humano
/ barra	barra	errar é/humano	errar é/humano

(...)	parêntesis	(errar é humano) ^ ^	(errar é humano)
[...]	parêntesis rectos	[errar é humano] ^ ^	[errar é—humano]

**Tab. 5-** Símbolos de revisão de provas, relacionados com a pontuação.

Tomando estes símbolos como inspiração, foi desenvolvida uma linguagem visual que pretende representar os diversos tipos de alteração possíveis ao poema-rascunho. Para além disso pretendia-se conferir uma certa “manualidade” à representação das trocas, isto é, de maneira a parecer que as alterações foram feitas com um marcador, à semelhança da fase de revisão do processo editorial. De seguida serão indicadas as interações possíveis com o poema-rascunho e a respectiva representação estática:

#### Troca de Palavras:

sem sair (vinho) deste mar marinho  
até massas pacatas, mar vinho  
a laranjeira tem no mar marinho  
até massas pacatas, mar (nunca)

#### Troca de Linhas:

levados na massa turva do mar  
[ sete da paragem dum mar qualquer ]  
[ um mar marítimo que tentam calar ]  
de um mar marítimo de mulher

### Apagar Palavras:

de paragem e mar a mesma roupa  
digno de ser no mar marinho  
~~Teito~~  
até massas pacatas, mar vinho  
as paragens do mar passam por mim

### Apagar Linhas:

as baías do mar passam por mim  
a massa move, mas o mar arrasta  
até massa do mar me anoitece  
[ ~~amiga amante, mar adriático~~ ]

### Editar Palavras:

que não tem mar dentro do oceano  
o mar ~~marítimo~~ e humano  
nalguma burka o teu mar marinho  
até águas pacatas, mar vinho

### Editar Linhas:

toda a báltica, todo o mar  
coisas, rosas, lenços, fatos por passar  
que a razão dum mar adriático  
nova linha escrita à mão

### Palavra Gerada:

se um mar é igual a atlântico  
as baías do mar passam por mim  
de baía e mar a mesma roupa  
toda a paragem, todo o mar

### Linha Gerada:

digo bom mar à paragem e canto  
de uma súbita água e mar santo  
um odor a tensão do mar marítimo  
quer de mar quer de paragem

### Palavra Adicionada:

as baías do mar passam por mim  
levados na água turva do mar  
um mar marítimo que tentam calar  
amor é mar, amor é oceano **serpentear** <sup>+</sup>

### Linha Adicionada:

porque mar é igual a oceano  
chamam-se baía ou gritam mar  
o mar marinho por te encontrar  
toda a paragem, todo o mar  
**isto disse o mar marinho** <sup>+</sup>

A representação dinâmica pretende representar, o mais literal possível, as representações estáticas, embora, em alguns casos, não haja uma associação tão literal e imediata entre representação estática e dinâmica. De seguida vão ser descritas as animações referentes a cada tipo de interação ao poema-rascunho:

### Troca de Palavras:

As palavras começam na posição inicial, antes da troca, e ambas descrevem o movimento que as leva à sua actual posição, depois da troca. Esta animação pretende representar literalmente a alteração da posição das palavras em questão.

### Troca de Linhas:

À semelhança da troca de palavras, as linhas são colocadas na posição inicial, antes de troca, e deslocam-se até às suas novas posições, depois da troca. Mais uma vez, esta animação pretende mostrar, literalmente, a troca de posições.



**Apagar Palavra:**

A palavra que foi apagada aparece na sua posição inicial, e surge uma linha vermelha que a risca de uma ponta à outra. De seguida, tanto a palavra como a linha desaparecem, eliminando também o espaço reservado para ambas. O acto de riscar por cima é amplamente utilizado para simbolizar que a palavra ou expressão por baixo deve ser eliminada, inclusivamente está presente na tabela de símbolos para representar erros ortográficos ou gramaticais (Tabela 2).

**Apagar Linhas:**

À semelhança da animação de apagar palavras, a linha que foi apagada aparece na sua posição original, e surge uma linha vermelha que a risca de uma ponta à outra. Depois do risco sobrepor toda a linha, ambos desaparecem bem como o espaço reservado para eles. Mais uma vez optei por esta animação porque a ação de riscar está fortemente ligada à supressão de elementos.

**Editar Palavras:**

A palavra anterior à edição é colocada na respectiva posição, e vai sendo apagada, caracter a caracter. Quando a palavra anterior desaparece completamente, surge a nova palavra, também caracter a caracter. Esta animação pretende simular o processo de apagar uma palavra e escrever outra, em qualquer editor de texto.

**Editar Linhas:**

À semelhança da edição de palavras, a linha original é mostrada na sua posição e vai sendo apagada, um caracter de cada vez. Quando a linha original desaparece na totalidade, começa a aparecer a nova linha, também um caracter de cada vez. Mais uma vez, esta animação simula o processo de apagar uma linha e escrever outra em qualquer editor de texto.

**Palavra Gerada:**

Esta animação pretende simular o processo de leitura do livro “Cent mille milliards de poèmes” (Queneau, R, 1961), que é um livro cujas páginas estão cortadas de maneira a permitir folhear linhas individualmente. Assim sendo, quando uma linha é folheada, surge outra por trás com um conteúdo diferente, e que, por consequência, altera o sentido do poema como um todo. Neste caso, este conceito é aplicado a palavras, ou seja, é como se tivéssemos um *post-it* por cima do texto que é retirado, revelando assim o que está por trás.

**Linha Gerada:**

À semelhança da animação de palavras geradas, pretende-se simular o processo de leitura do livro de Queneau (1961), onde as linhas podem ser folheadas individualmente, revelando outras linhas. Neste caso, a animação corresponde ao folhear de uma linha, que por consequência revela outra linha, que inicialmente estava por trás.

**Palavra Adicionada:**

A animação deste tipo de interação também é inspirada no livro de Queneau (1961), mas neste caso, é como se tivéssemos a recolocar no poema uma linha que já foi folheada. Na prática, como não existe nenhum texto a ser tapado, porque não havia antes nenhuma palavra no sítio desta nova palavra, é como se tivéssemos a colar um *post-it* com uma nova palavra no final de uma linha.

**Linha Adicionada:**

Esta animação é muito semelhante à de Adição de Palavras, explicada acima, no entanto é aplicada a linhas.

## 6.3 - Design de Interação

Nesta secção são definidas as escolhas relativas ao design de interação, isto é, as escolhas feitas quanto ao comportamento dos elementos da *interface* quando o utilizador interage com eles. Visto que há dois tipos de elementos, ferramentas e fragmentos poéticos, são descritas em separado, as interações possíveis com ambos os tipos. Para ambas as situações deu-se prioridade à utilização de interações que já são usadas em outras aplicações, de maneira a que o utilizador já esteja familiarizado com os tipos de interação. Para além disso, pretende-se que a interação seja o mais intuitiva possível.

### 6.3.1 - Interação com as Ferramentas

Ao longo da aplicação é disponibilizado um conjunto de ferramentas para que o utilizador crie o seu poema. No entanto, graficamente, estes elementos são semelhantes a outros elementos estáticos da aplicação, isto é, que não oferecem interação. Por isso é importante que os elementos dinâmicos reajam a vários tipos de interação, para que se torne perceptível que estes suportam interatividade.

Se olharmos para as ferramentas existentes, é possível ver que há dois tipos de botões: botões rádio, que têm dois estados possíveis, passíveis de serem escolhidos através da interação; botões mais comuns, que desencadeiam uma ação sempre que são clicados. No caso dos botões mais vulgares, que apenas desencadeiam uma ação quando há interação com eles, é crucial que reajam quando o rato lhes passe por cima, para que seja perceptível que há uma interação disponível. Esta técnica é recorrente em aplicações, *Websites*, etc, e esta mudança de aspeto pode acontecer a vários níveis: mudança de cor do elemento, alteração no tamanho do elemento, etc. Neste caso, os botões invertem a cor de fundo e do pictograma associado, e após serem clicados, estes assumem a sua aparência inicial. Nos botões rádio, para além de ter que ser perceptível que são botões, e que por isso dispõem de interação, é necessário que o respectivo estado seja visível. Na aplicação existem dois tipos de botão rádio (Figura 25): botões em que os estados disponíveis são do tipo selecionado/não selecionado (botão de “Visualizar Processo Co-Criativo” e “Ativar/Desativar Janelas de Ajuda”); botões que dispõem de mais que uma possibilidade, onde uma não é o oposto da outra (botão “Seleção de Idioma”). No primeiro caso, em que as possibilidades são opostas, a representação do estado é feita através da cor de fundo do botão e da cor do símbolo que representa a ação. Caso o botão esteja em positivo, que é a aparência por defeito – fundo branco e pictograma a preto – a respectiva ferramenta está desativada. Caso haja um *click* no botão há alteração na sua aparência, que acompanha a mudança de estado para a negativo – fundo preto e pictograma a branco. No segundo caso, em que as possibilidades não são opostas, é importante informar o utilizador dos estados disponíveis, em vez de o obrigar a carregar no botão consecutivamente para descobrir os estados disponíveis. Neste segundo caso, onde apenas se enquadra o botão “Seleção de Idiomas”, as duas possibilidades de idioma estão visíveis, no entanto, a posição selecionada está a negativo, enquanto que a não selecionada está a positivo. Quando o rato passa por cima da opção ativada nada acontece, no entanto, caso passe por cima da opção não ativa, esta muda de aparência, e caso haja um *click*, é efectuada a alteração do estado, passando a nova opção para negativo e a antiga para positivo.



Fig.25- Tipos de botões rádio disponíveis.

Tanto no módulo “Rascunhos” como no “Versos” e “Linhas”, são disponibilizados um conjunto de opções que ajudam o utilizador a direccionar a geração de conteúdos para a os resultados pretendidos. Essas opções são disponibilizadas sobre a forma de *inputs*, que podem ser de três tipos: de texto, de números e *inputs* que permitem seleccionar uma de vários opções disponíveis. Neste último caso, quando o rato passa por cima do *input*, a seta que representa a possibilidade de abrir um menu *drop-down* muda de cor e, à semelhança dos módulos, passa a negativo – fundo preto e seta a branco – para que esta possibilidade de interação seja perceptível. No caso dos *inputs* de números, que só existem no campo “Número de Sílabas”, opção do módulo de geração de versos, e nos campos de inserção de formas personalizadas, quando o rato lhe passa por cima aparecem duas setas, que permitem aumentar ou reduzir o algarismo representado no *input*. Quando um *input* de texto é clicado o seu fundo muda para preto, para se perceber que este campo está em foco. Quando um *input* está em foco, caso uma palavra do poema-rascunho seja clicada, essa palavra é colocada como conteúdo do *input*. Para além disso, os *inputs* de texto contêm um botão “X”, que permite limpar o conteúdo do *input* de texto, que à semelhança dos outros botões muda de aparência quando o rato lhe passa por cima.

O facto da alteração de estados, tanto de botões como de *inputs*, ser representado sempre através da mesma alteração na aparência, isto é, fundo branco e símbolo a preto, ou fundo preto e símbolo a branco, reforça a consistência visual mantida ao longo da aplicação, e, como é referido na secção 3.2, faz com que o utilizador consiga padronizar a execução de tarefas semelhantes.

Caso a opção das janelas de ajuda esteja ativada, todos os *inputs* de texto disponibilizam informação sobre o seu preenchimento, quando são clicados (Figura 26). Essa informação aparece como conteúdo de um balão de fala e ajuda o utili-

zador a preencher os respectivos campos. Para além disso informam o utilizador acerca da funcionalidade do *input* “em foco”, que se refere à funcionalidade descrita no parágrafo anterior – possibilidade de carregar numa palavra do poema-rascunho para preencher determinado campo, caso este esteja em foco.

Para além dos *inputs* de texto, todos os botões dispõem de um balão de fala explicativo da ferramenta que representam, que aparece quando o rato lhes passa por cima (Figura 27), caso a opção das janelas de ajuda esteja ativada. Este factor ajuda os utilizadores menos familiarizados com aplicações *Web* e aplicações móveis a perceber o significado dos ícones apresentados, e a sua utilização no âmbito da aplicação. A existência de uma funcionalidade que permite ativar/desativar janelas de ajuda é muito importante visto que possibilita que um utilizador recente consulte a informação necessária para aprender a utilizar o sistema, mas por outro lado, permite que utilizadores experientes desativem esta funcionalidade para optimizarem a sua utilização, porque a determinada altura pode tornar-se aborrecido para o utilizador que estas janelas estejam sempre a aparecer, visto que este já está familiarizado com as ferramentas disponíveis. Este factor vai de encontro à directriz de design de interação (secção 3.2) que nos diz que a aplicação deve ser flexível na sua utilização.

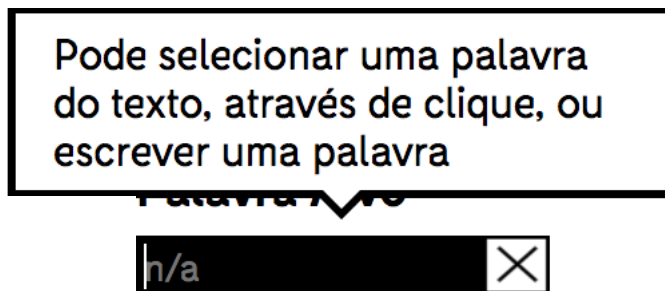


Fig.26 -Janelas de ajuda nos *inputs* de texto.

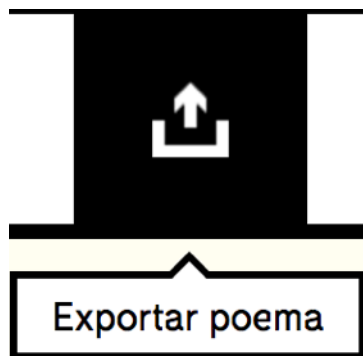


Fig.27 - Janelas de ajuda nos botões.

Sempre que as condições reunidas não sejam suficientes para a execução de determinada tarefa, seja porque determinado campo duma ferramenta não está preenchido, ou porque ainda não há conteúdos para uma dada ação ser executada, aparece uma janela que informa o utilizador sobre que componentes estão em falta (Figura 28). Isto vai de encontro à regra da “Visibilidade do estado do sistema”, apresentada na secção 3.2. Também vai de encontro à regra do “reconhecimento, análise e recuperação de erros” que nos diz que o *feedback* fornecido pelo sistema, em caso de erro, deve ser explicativo e identificar a causa do mesmo com clareza, tanto apresentada na secção 3.2.

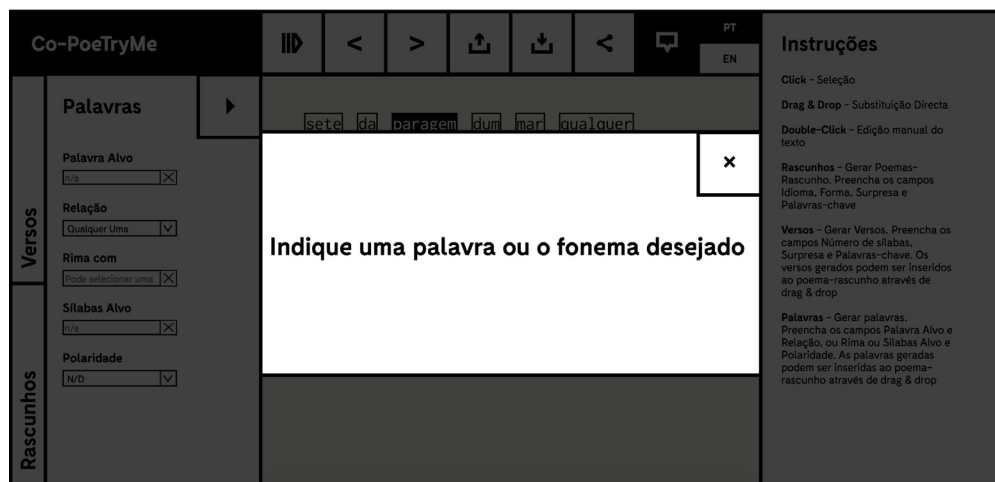


Fig.28- Janelas de *feedback* a situações de erro.

## 6.3.2 - Interação com os Fragmentos Poéticos

Como já foi introduzido na secção 5.2.5, a representação dos fragmentos poéticos sobre a forma de pequenos módulos pretende “transformar” o processo de escrita de um poema numa espécie de jogo/puzzle. Essa mesma lógica é seguida no que toca à interação com os módulos, e pretende-se que a interação com os fragmentos poéticos seja semelhante à interação com as peças de um puzzle. No entanto, nem sempre é fácil transportar de uma maneira literal as acções humanas sobre um conjunto de objectos para o meio digital. De seguida são explicadas as interações possíveis com os fragmentos poéticos e o seu paralelismo, caso exista, com acções do mundo real:

### **Rato em cima de um fragmento poético:**

Quando o rato passa por cima de um fragmento poético, este revela o seu contorno para demonstrar a possibilidade de interação. No caso dos fragmentos poéticos que integram o poema-rascunho, caso o rato passe por cima de uma linha, esta revela um contorno que engloba todas as palavras desse verso, e tem alguma margem em relação ao contorno das palavras de maneira a facilitar a selecção de versos. Caso o rato passe por cima de uma palavra, tanto o contorno da palavra como do verso da qual ela faz parte são revelados (Figura 29). No caso dos fragmentos sugeridos, visto que fragmentos do tipo verso nunca estão juntos de fragmentos do tipo palavra, não é necessário fazer esta distinção ao nível da margem do contorno. No entanto, como estes fragmentos são dispostos como um aglomerado/núvem, quando o rato lhes passa por cima, o respectivo contorno fica mais grosso para identificar qual será o fragmento seleccionado caso haja um *click* (Figura 30) e esta peça passa para a frente das outras, para o seu conteúdo poder ser lido.

digo bom mar à baía e canto  
 mar, e água certo de alto espanto  
 vibrará contra mim seu mar marítimo  
 esqueces-me da baía pró mar

**digo** bom mar à baía e canto  
 mar, e água certo de alto espanto  
 vibrará contra mim seu mar marítimo  
 esqueces-me da baía pró mar

Fig.29- Hover verso e hover palavra do poema-rascunho

### Gerados

quer de mar quer de paragem

marítimas mar de mão

à massa do mar

no mar passado neste golfo

adriáticas mar de mão

brilhas à massa do mar

o gosto dum mar adriático

e a paragem passa a mar

toda a canção é um mar marinho

adriatico em mar pe em cima

### Geradas

calda oxigénio

misturas

nutrições

coisa beheragens substâncias

gelo

farelada ensaboadura

mistura casal cachoeira

alimentos gelos

líquidas prato salmouras

hidromancia pratos

Fig.30 - Hover linha gerada e hover palavra gerada.



### Click num fragmento poético:

O *click* num fragmento poético faz com que este fique selecionado, e esta alteração de estado é representada através de mudança no aspeto do fragmento poético clicado, que à semelhança dos botões e ferramentas passa para negativo (fundo preto e tipografia a branco). A este nível há diferenças entre carregar num verso ou numa palavra, e esse *feedback* é dado através do módulo à esquerda do da Edição do Poema: caso se *click* numa palavra, o módulo imediatamente à esquerda revela as opções de geração de palavras e preenche as opções “Palavra-Alvo” e “Sílabas Alvo” com o conteúdo do fragmento – palavra clicada; caso se *click* num verso, o módulo imediatamente à esquerda mostra as opções de geração de versos e preenche o campo “Linha Seleccionada” com o conteúdo do verso clicado (Figura 31). Isto acontece tanto para fragmentos integrantes do poema-rascunho, como para os fragmentos poéticos sugeridos.

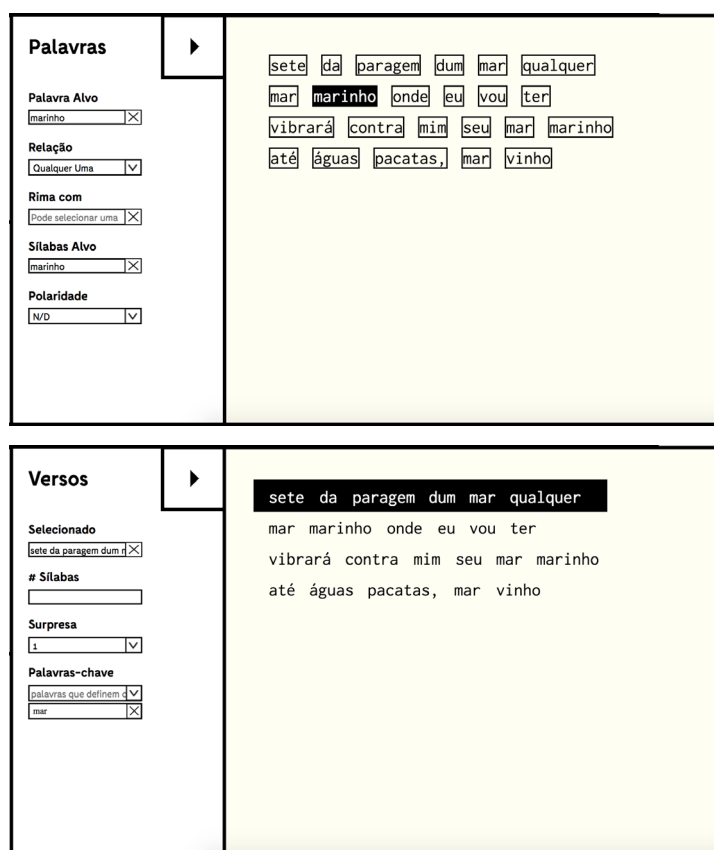


Fig.31- Verso clicado vs Palavra clicada

### ***Double-Click num fragmento poético:***

Caso haja um *click* duplo num fragmento poético que integre o poema-rascunho, seja ele uma palavra ou um verso, este revela um *input* de texto, cujo conteúdo é o mesmo do fragmento poético carregado. Outra maneira de o fazer é, selecionar um fragmento através de *click* e carregar na tecla *enter*. Este tipo de interação permite que o utilizador edite ou apague o conteúdo do respectivo fragmento. Embora apareçam imediatamente à direita do *input*, os botões de confirmar a alteração ao conteúdo e apagar o fragmento poético, existem mais maneiras de o fazer. Isto é, são utilizadas interações que já estão presentes noutras aplicações, e com as quais o utilizador já está familiarizado. Por exemplo, caso um fragmento esteja em modo de edição e a tecla *enter* seja carregada ou caso haja um *click* fora do *input*, estas acções são consideradas como validação, isto é, são equivalentes ao *click* no botão de confirmação. De maneira similar, caso um fragmento esteja selecionado e a tecla *backspace* seja carregada, este fragmento é imediatamente eliminado. No caso dos fragmentos poéticos serem palavras, e visto que a introdução de várias palavras num campo que apenas está à espera de uma poderia levar ao erro, o *click* na tecla “barra de espaços” serve também como confirmação. Este tipo de interação só existe para os fragmentos poéticos que são parte integrante do poema-rascunho. Para além disto, é possível editar o poema-rascunho como um todo, através de *double-click* no módulo de edição do poema-rascunho. Neste caso surge uma área de texto cujo conteúdo é o poema-rascunho, na integra. À semelhança da edição de palavras e versos, é disponibilizado um botão Confirmar, mas a ação pode ser concluída através de um *click* fora da zona de texto – neste caso o botão *enter* não está disponível para confirmar a ação porque a tecla é necessária para adicionar quebras de linha ao poema (Figura 32).

as baías do mar passam por mim	Ok
	X

mar adriático de cetim

mais do que tentar pode mar marinho

até massas pacatas, mar vinho

as	baías	do	mar	passam	Ok	mim
					X	

mar	adriático	de	cetim
-----	-----------	----	-------

mais	do	que	tentar	pode	mar	marinho
------	----	-----	--------	------	-----	---------

até	massas	pacatas,	mar	vinho
-----	--------	----------	-----	-------

as	baías	do	mar	passam	por	mim
mar	adriático	de	cetim			
mais	do	que	tentar	pode	mar	marinho
até	massas	pacatas,	mar	vinho		
Ok						

Fig.32- Editar Linha/ Editar Palavra / Editar Texto.

### ***Drag & Drop num fragmento poético:***

Em primeiro lugar, este tipo de interação é diferente para fragmentos poéticos que integram e que não integram o poema-rascunho. Caso haja um *click* e arrastamento de um verso do poema-rascunho, este fica selecionado e os contornos dos outros versos ficam visíveis para dar a entender aos utilizadores que esses são os espaços destinados para “largar” o verso arrastado. No caso das palavras, caso haja um arrastamento, a palavra, à semelhança do verso, fica selecionada e assume a aparência de um fragmento poético selecionado (negativo). No entanto, visto que no modo palavras todos os fragmentos revelam os seus contornos, não existe essa alteração na aparência. Contudo, caso a palavra arrastada esteja por cima de outra do poema-rascunho, esta segunda altera o seu aspeto para mostrar ao utilizador qual a troca que será efectuada caso este “largue” o fragmento nesse momento (Figura 33). Para os fragmentos poéticos sugeridos, caso o utilizador arraste um dos versos gerados, todos os versos do poema-rascunho revelam os seus contornos como forma de revelarem que são as zonas para “largar” este fragmento. Quando um verso sugerido é arrastado, para além dos versos do poema-rascunho revelarem o seu contorno, surge também um botão “+” que permite a adição do verso ao poema. Esta adição é feita arrastando o verso sugerido para cima do botão “+” que altera o seu aspeto quando o verso sugerido está “por cima” dele, de maneira a representar essa possibilidade. No caso de o arrastamento ser feito numa palavra sugerida, aparece na terminação de todos os versos um botão “+” que permite a adição da palavra sugerida a essa linha. Mais uma vez, quando a palavra é arrastada para cima de um desses botões, este altera a sua aparência para representar a disponibilidade desta interação (Figura 34).

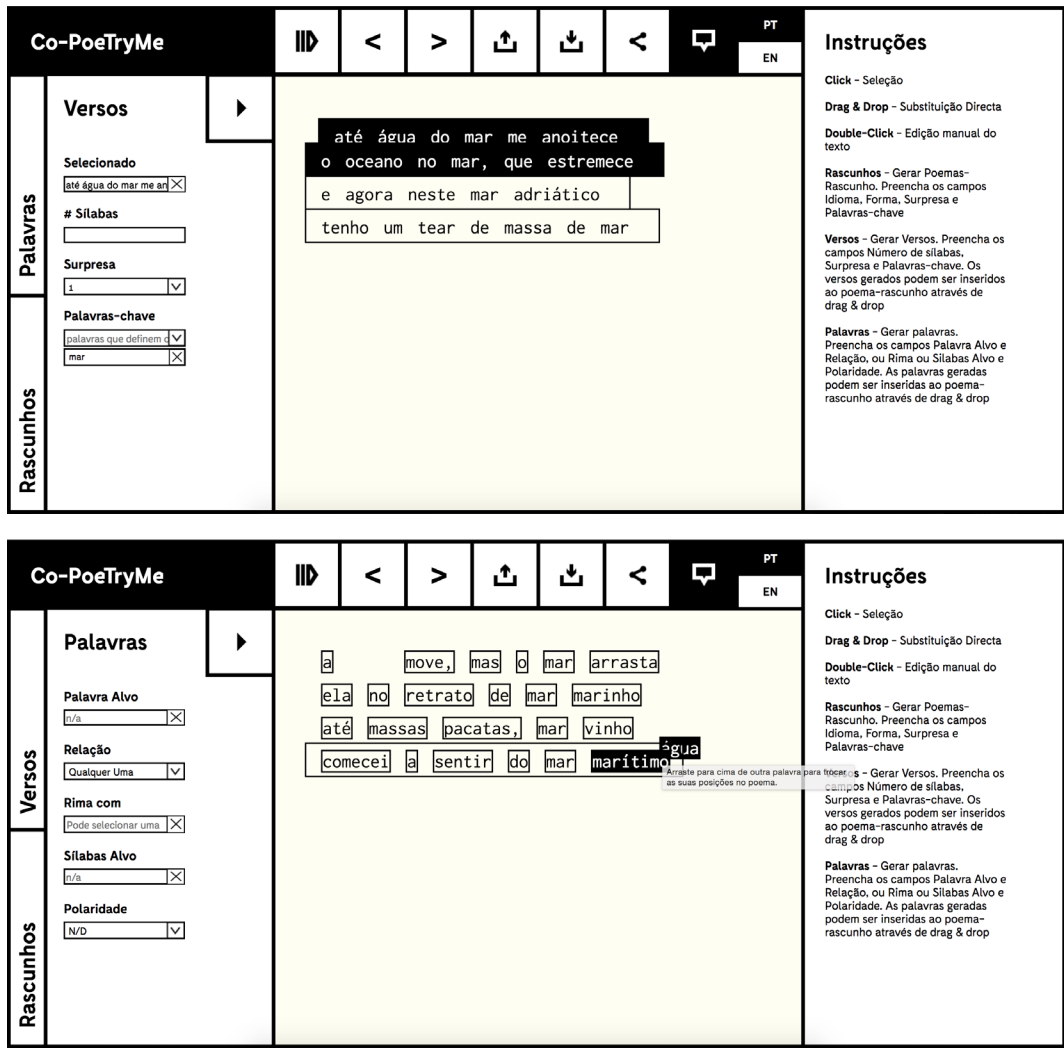


Fig.33- Drag linha / drag palavra

**Co-PoeTryMe** PT EN

**Versos**

**Palavras**

Selecionado

# Silabas

Surpresa

Palavras-chave

**Rascunhos**

as baías do mar passam por mim  
 leite em pó, leite mar adriático  
 adriático, tlim, tlim, tlim, tlim  
 eu marchava de mar e de paragem  
 +  
 onde uns vêem mar e massas

**Gerados**

maritimas mar de mão  
 marinhas mar de mão paradas  
 marinhas mar de mão coo  
 que o mar vai levando pelo oceano  
 as águas não têm mar  
 tendo mais o fim do mar maritimo  
 mas por mar maritimo

**Banco**

**Lixo**

**Instruções**

**Co-PoeTryMe** PT EN

**Palavras**

Palavra Alvo

Relação

Rima com

Silabas Alvo

Polaridade

**Versos**

tenho um tear de massa de mar + bolos  
 o asilo o marinho mar lar +  
 nem mar maritimo em virgem pura +  
 vem com mar marinho e traz consigo +

**Gerada**

nuvem  
 tifão Cero feg  
 grude tropas  
 bolos  
 polmes  
 pizime  
 pastel  
 pastel runde  
 gentes  
 turbas tropa  
 bolo grudes gentes  
 grupos

**Banco**

**Lixo**

**Instruções**

Fig.34- Adicionar linha / Adicionar palavra

## 6.4 -Protótipos / Ecrãs

A prototipagem é um método de representação usado na fase inicial do processo de desenvolvimento de uma aplicação e ajuda a testar várias possibilidades de *interface* e interação. Isto permite que o designer experimente várias possibilidades de design, de forma fácil e barata, e possibilita o reconhecimento de erros de design antes da fase de implementação.

Nesta secção são ser mostrados os ecrãs das várias versões desenvolvidas, e a sua evolução até aos ecrãs da aplicação final. Por um lado, é possível verificar a evolução desde a primeira abordagem, mas por outro permite identificar semelhanças. Serão ainda visíveis as alterações ao nível do estilização e disposição dos módulos, e será possível perceber que módulos passaram a existir, e que módulos deixaram de existir.

A primeira versão da aplicação, representada nas Figuras 35, 36 e 37, já tinha uma estrutura dividida em módulos que se organizavam em torno do módulo principal da aplicação – o da Edição do Poema. No ecrã inicial, era apresentado um módulo chamado “Resumo” que continha uma pequena descrição da origem e objectivos do projecto, que após a geração de um poema “encolhia” para o módulo “+” apresentado na barra superior de navegação. Este módulo poderia ser novamente expandido através de *click*. Do lado esquerdo, numa fase inicial, tínhamos o módulo de Geração de Poemas, onde as opções de geração eram distribuídas pelo espaço disponível, e abaixo estava disponível o botão de geração de poemas. Após a geração de um poema-rascunho, o módulo de Geração de Poemas dava lugar ao módulo de geração de palavras e a um outro módulo que permitia escolher o tipo de fragmentação do poema, em versos ou em palavras. Do lado direito estavam presentes três módulos: o módulo das Instruções; o módulo “twitter” que continha uma hiperligação para a página do twitter do PoeTryMe, onde periodicamente são publicados poemas gerados automaticamente pelo sistema, acerca dos assuntos mais atuais e mais presentes nos Tweets dos utilizadores; o módulo “Mais” que continha mais um conjunto de ferramentas: numa fase inicial, enquanto não houvesse poema-rascunho, apenas estava disponível a possibilidade de “Importar” um poema, mas após um poema-rascunho ser gerado ou importado, era possível “Exportar” e “Partilhar”. Podemos ainda verificar que a forma de representar a seleção de um fragmento era feita através de sublinhado, que era também a maneira como os fragmentos reagiam caso o rato lhes passasse por cima.

<b>Geração de Poemas</b>  Idioma <input type="text" value="Português"/>  Forma <input type="text" value="Banco de 4"/>  Sementes <input type="text" value="Palavras sobre o tema escolhido"/>  Surpresa <input type="text" value="1"/>  Gerar Poema	<b>Co-PoeTryMe</b>	<b>Instruções</b>  Click - Seleção  Drag & Drop - Substituição direta  Double-Click - Edição manual do texto  <b>twitter</b>  Página do Twitter do PoeTryMe onde são postados, de duas em duas horas, poemas gerados relacionados com os tweets mais populares e actuais.  <b>Mais</b>  Importar
	<b>Resumo</b>  PoeTryMe é uma aplicação co-criativa de geração de poesia. O objectivo deste projecto é aglizar o processo criativo humano, na área da escrita criativa.  O sistema, num primeiro momento, através da inserção de dados como a forma poética, língua, sementes (palavras de inspiração/tema do poema) e um grau de surpresa (que influencia a escolha de palavras por parte do sistema), permite gerar um poema rascunho. Caso esse poema não seja minimamente satisfatório, é possível gerar outros poemas, com (ou sem) as mesmas características. Num segundo momento permite edição do poema rascunho a vários níveis.  >Permite substituição directa de posições de palavras ou expressões no poema.  >Permite a substituição de palavras ou expressões por outras geradas pelo sistema: No caso da re-geração de palavras, podemos especificar características como a relação semântica com a palavra a substituir, o número de sílabas e a rima; No caso da re-geração de linhas podemos definir novas sementes, um novo grau de surpresa e o número de sílabas.  >Permite a edição directa de texto, isto é, apagar ou escrever palavras.	

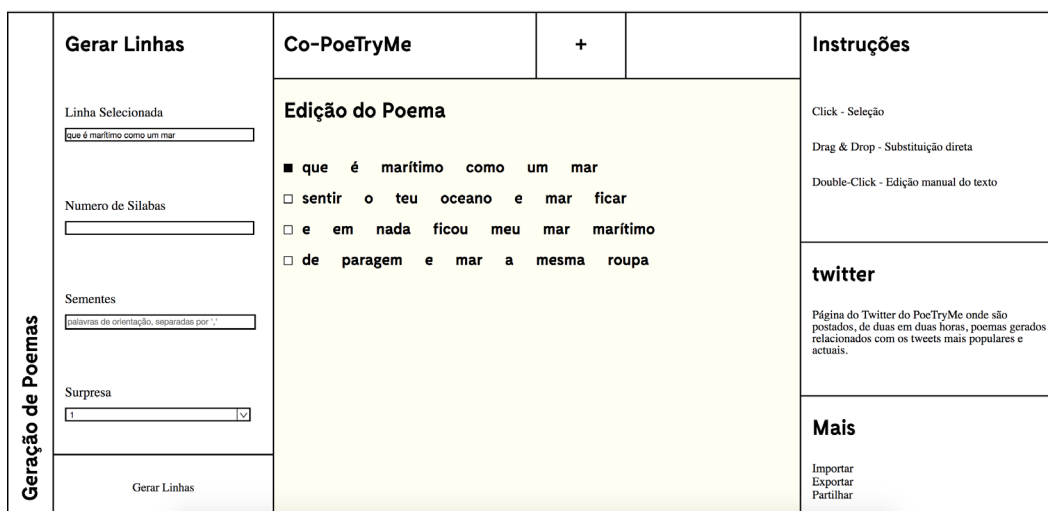
<b>Geração de Poemas</b>  Sustituição por palavras <input checked="" type="checkbox"/> linhas  <b>Gerar Palavras</b>  Palavra Seleccionada <input type="text" value="ba"/> Relação <input type="text" value="Qualquer Uma"/> Rima com <input type="text" value="Palavra que rima com"/> Nr. sílabas <input type="text" value="5"/>  Gerar Palavras	<b>Co-PoeTryMe</b>	+	<b>Instruções</b>  Click - Seleção  Drag & Drop - Substituição direta  Double-Click - Edição manual do texto  <b>twitter</b>  Página do Twitter do PoeTryMe onde são postados, de duas em duas horas, poemas gerados relacionados com os tweets mais populares e actuais.  <b>Mais</b>  Importar Exportar Partilhar
	<b>Edição do Poema</b>  toda a baía, todo o mar  mar marinho, engraçado no falar  louco e ébrio num mar marinho  até massas pacatas, mar vinho  <b>Pontuação</b>	<b>Forma</b>	

<b>Geração de Poemas</b>  Sustituição por palavras <input checked="" type="checkbox"/> linhas  <b>Gerar Palavras</b>  Palavra Seleccionada <input type="text" value="engraçado"/> Relação <input type="text" value="Sinónimo"/> Rima com <input type="text" value="Palavra que rima com"/> Nr. sílabas <input type="text"/>  Gerar Palavras	<b>Co-PoeTryMe</b>	+	<b>Banco de Palavras</b>  magano      quebra chistoso      jocoso galante  folgazão      jovial divertido      gracioso espirituoso      brincalhão faceto	<b>Instruções</b>  <b>twitter</b>  <b>Mais</b>
	<b>Edição do Poema</b>  toda a baía, todo o mar  mar marinho, engraçado no falar  louco e ébrio num mar marinho  até massas pacatas, mar vinho  <b>Pontuação</b>	<b>Forma</b>	<b>Repositorio Palavras</b>  <b>Lixo</b>	

Figs.35, 36, 37- Screenshots da primeira versão da aplicação.



Na segunda versão representada na Figura 38, a maioria da estrutura manteve-se, sendo que as principais alterações foram feitas ao nível da seleção de fragmentos poéticos. Como se pode verificar, o módulo que permitia alterar o tipo de fragmentação do poema desapareceu, e pretende-se que seja o próprio poema a mostrar os tipos de fragmentação possível. Nesta versão, caso se pretenda seleccionar uma palavra, basta carregar nela, no entanto, para se seleccionar um verso, é preciso ir com o rato para a zona imediatamente antes da linha, e estes quadrados, representados na Figura 38 são revelados. O *click* num desses quadrados correspondia à seleção da linha imediatamente à direita. A substituição da posição de linhas no poema, em vez de ser feita através do arrastamento de uma linha para cima da outra, era feita através do arrastamento destes quadrados, uns para cima dos outros. No entanto, esta solução não era a mais intuitiva, e foi abandonada nas versões seguintes.



**Fig.38-** *Screenshot* da segunda versão da aplicação.

Na terceira versão, mais uma vez o foco foi colocado na zona de Edição de Poemas, e na maneira como se podia seleccionar os diferentes tipos de fragmentos poéticos. Neste caso a seleção por palavras ou linhas era feita através da seleção de ferramentas, representadas através de botões, no módulo de Edição de Poemas, como é possível ver na Figura 39. Neste caso está seleccionada a opção de arrastar palavras, no entanto, caso se carregue no botão, aparece um menu

drop-down (Figura 40) que permitia alterar o modo de fragmentação para linhas. O botão em baixo correspondia a ferramenta de edição, e dispunha das opções edição por palavras, linhas ou poema (Figura 40). Mais uma vez, esta solução não foi considerada a mais adequada e foi abandonada nas versões posteriores.

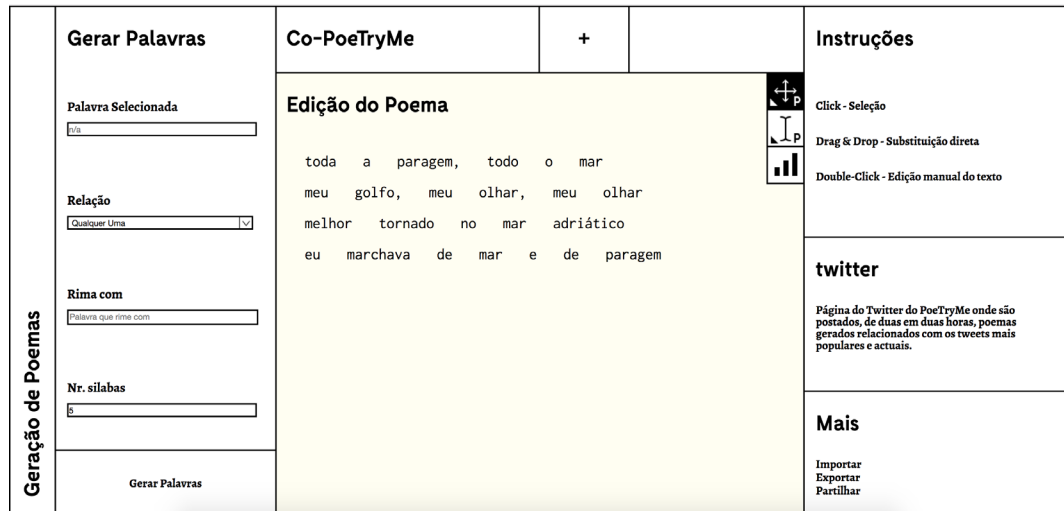


Fig.39- Screenshot da terceira versão da aplicação.

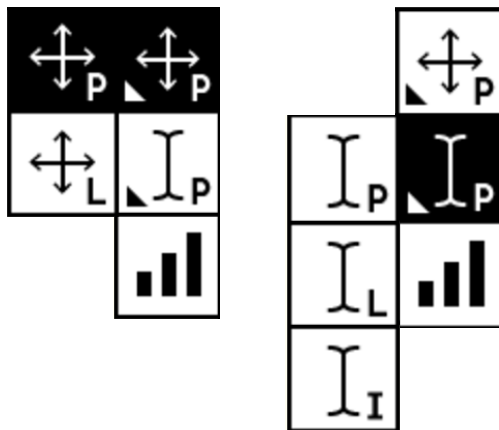


Fig.40- Ferramentas de seleção e edição utilizadas na terceira versão da aplicação.

Na quarta versão (Figuras 41 e 42), uma vez mais se tentou resolver a questão da fragmentação do poema e da seleção dos vários tipos de fragmentos, enquanto que o resto da estrutura da aplicação se manteve. Nesta versão houve uma aproximação ao processo de seleção de fragmentos actualmente implementado. No entanto, neste caso, só após uma linha ser selecionada, através de *click*, é que as palavras que a constituem estão disponíveis para *click*, ou seja, para se seleccionar uma determinada palavra seria preciso, em primeiro lugar, seleccionar a linha que a continha, e só depois seria possível carregar nessa mesma palavra. Embora esta solução ainda não fosse ideal, o facto das linhas terem uma área maior para *click*, e as palavras aparecerem como “peças” mais pequenas dentro das “peças-linha”, foi um avanço no processo de selecção de fragmentos poéticos. É nesta versão que é definido o estado de um fragmento poético quando é seleccionado, que passou do sublinhado para negativo – os fragmentos, inicialmente, têm fundo “amarelo” e tipografia a preto, e quando seleccionados passam a ter fundo preto e tipografia a branco.

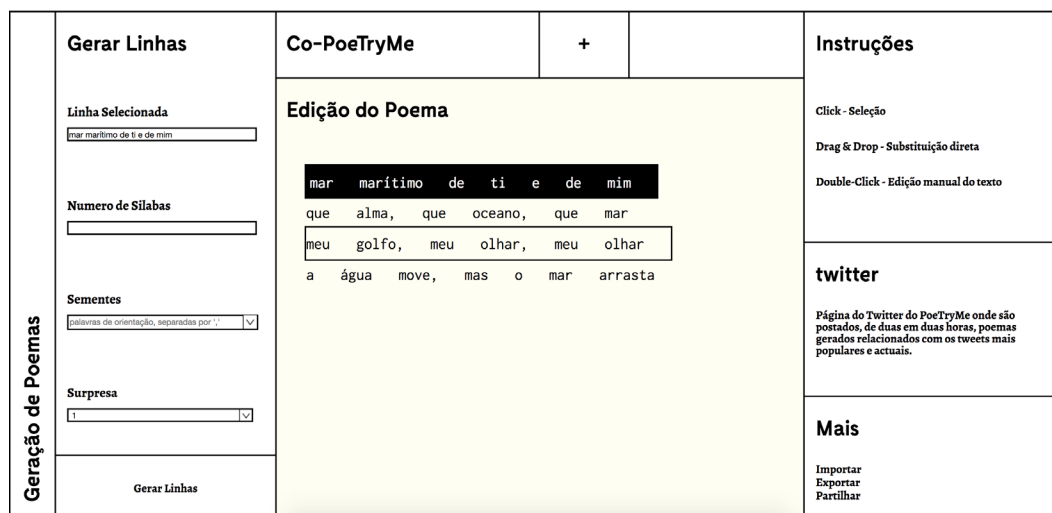


Fig.41- Screenshot da quarta versão da aplicação – linha seleccionada.

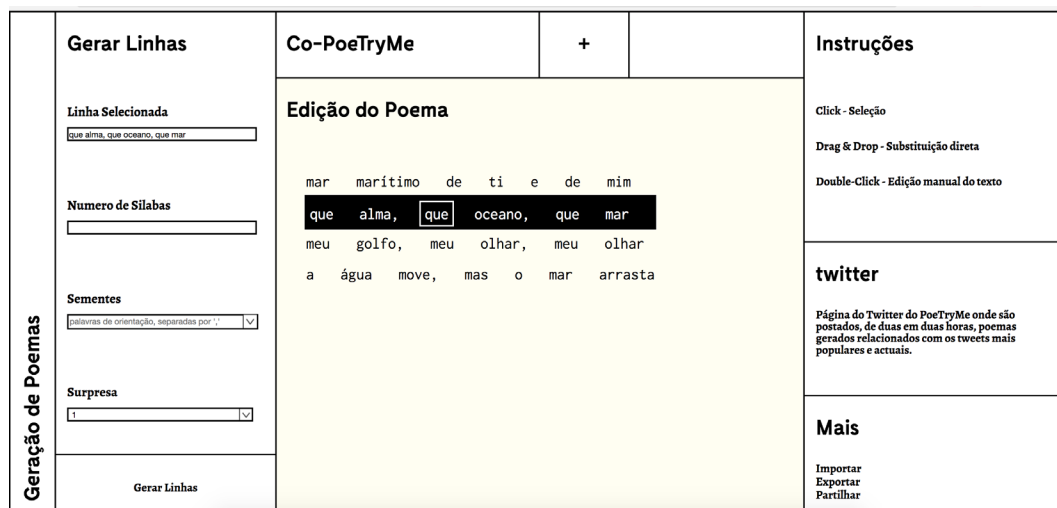


Fig.42- Screenshot da quarta versão da aplicação – palavra selecionada

Na quinta versão, em primeiro lugar, podemos verificar uma tentativa de uniformizar visualmente a *interface*: os contornos ficaram mais rígidos para que “casassem” melhor com a tipografia e as opções deixaram de estar espalhadas pelo espaço disponível (Figura 43). Ficou finalmente definida a forma como o utilizador pode interagir com os fragmentos do poema, neste caso linhas e versos. No caso do utilizador querer selecionar uma linha, basta que carregue na área que a contorna, mas fora das áreas das palavras que a constituem (Figura 44). No entanto, caso o utilizador esteja exatamente por cima duma palavra, esta revela o seu contorno e pode ser diretamente selecionada, o que não acontecia na versão anterior. É nesta versão que surge pela primeira vez o botão que permite alterar o idioma da *interface*. Este é um botão rádio, e neste caso, o facto da sigla “PT” estar a negativo significa que esta é a opção selecionada. Esta é também a primeira versão em que o processo co-criativo não é diretamente representado no módulo da Edição de Poemas. Neste caso, para a representação ser visível é necessário carregar no botão “a”, que ao estar selecionado fica com o fundo branco e tipografia a preto, para identificar que está ativo. Quando esta funcionalidade está ativa, todas as interações com o texto são desativadas, e é mostrado o conjunto de alterações feitas sob a forma de aglomerado. Esta é também a primeira versão que permite visualizar as trocas sob a forma de animação. Neste caso, como demonstra a Figura 45, era possível selecionar um intervalo de trocas a visualizar, sendo que o quadrado a vermelho indicava o número da primeira

troca a mostrar e o azul o número da última troca a mostrar. É possível ainda verificar que foi adicionado um parâmetro que permite restringir ainda mais a geração de palavras, que é a “Polaridade”.

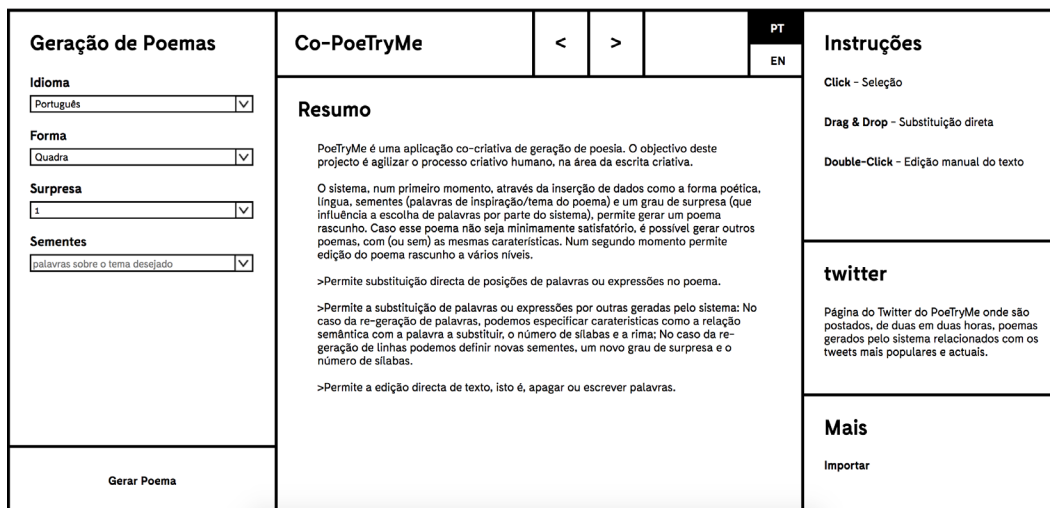


Fig.43- Screenshot da quinta versão da aplicação.

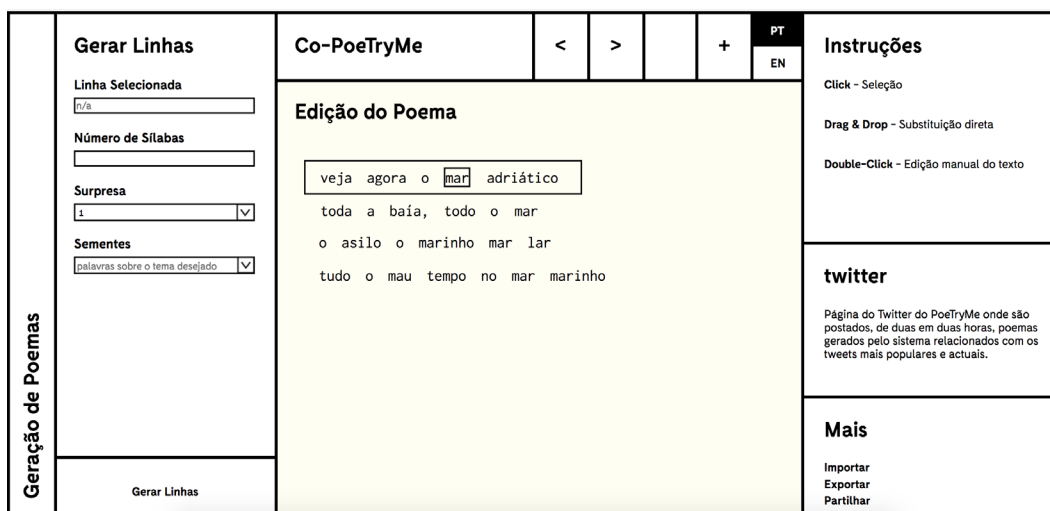
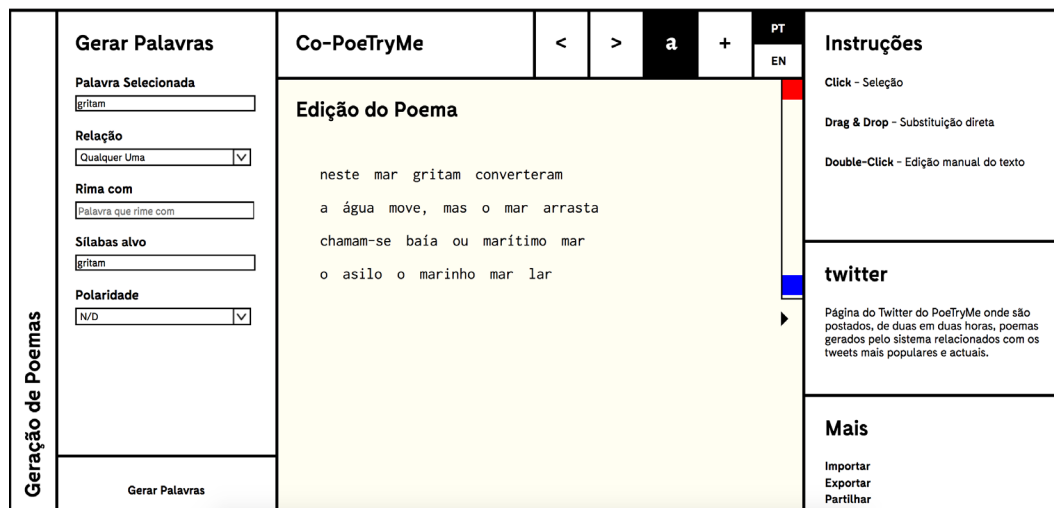


Fig.44- Screenshot da quinta versão da aplicação – método de seleção de fragmentos poéticos.



**Fig.45-** Screenshot da quinta versão da aplicação – método de seleção do intervalo de alterações ao texto a visualizar.

Também na última versão podemos identificar algumas diferenças em relação à versão anterior. Em primeiro lugar, a posição do título foi alterada. Isto porque com todos os títulos de módulo ao mesmo nível, o título da aplicação passava despercebido, e não tinha relevância que devia. Para além disso, e com o objectivo de destacar ainda mais esta secção do resto da *interface*, este é o único módulo que, não estando seleccionado, aparece por defeito em negativo (fundo preto e tipografia branca). Os contornos dos módulos foram ainda mais reforçados, de maneira a terem a mesma espessura das linhas que compõem a tipografia. Tanto o módulo “twitter” como o “Mais” desapareceram, sendo que no primeiro caso o módulo foi totalmente suprimido por não ser considerado relevante no âmbito desta aplicação, e no segundo caso, as ferramentas disponíveis neste módulo passaram a ser apresentadas sob a forma de botões, na barra de navegação superior (Figura 46). Também podemos verificar que os botões de gerar conteúdos mudaram de posição e alteraram a sua forma. Neste caso, como se pretendia fazer uma comparação entre o processo de escrita de um poema com um jogo, os botões Gerar passaram a ter a forma de um botão *play*. As opções e informações dos módulos viram o seu espaço reduzido, ficando assim uma área lateral reservada para botões e barras de scroll. O módulo “Resumo” também foi suprimido, por não apresentar relevância para a utilização da aplicação, e substituído pelo módulo “Começar” que indica as diferentes maneiras possíveis de começar a utilizar a aplicação. Podemos ainda verificar que o ícone que representa a vi-

sualização do processo co-criativo foi alterado, e embora seja, provavelmente, o ícone mais difícil de associar à tarefa que desempenha, penso que tem mais que ver com a tarefa que representa, visto que os traços verticais pretendem representar *frames* e a seta pretende representar um botão *play*. Podemos também ver que foi incluído um novo botão, para além dos adicionados em substituição do módulo Mais, o de “Ativar/Desativar Janelas de Ajuda”. Relativamente à representação dos versos e palavras geradas, que não sofria alteração desde a segunda versão, podemos verificar que foram alvo de remodelação. A sua nova aparência pretende que estes elementos sejam mais idênticos à representação dos fragmentos incluídos no poema-rascunho, de maneira a ser mais fácil de entender que podem ser adicionados ou usados em substituição dos fragmentos presentes no poema. Estas peças surgem sobre a forma de aglomerado, e por isso encobrem-se parcialmente, sendo contudo possível arrastar os fragmentos dentro do módulo para ler o seu conteúdo (Figura 47).

Quanto à visualização do processo co-criativo, podemos verificar que o botão *play* passou para o canto inferior direito, e deixou de ser possível escolher um intervalo de alterações a visualizar, visto que não foi atribuída relevância ao facto de ser possível ver apenas parte das trocas efectuadas. Em vez disso, ao carregar no botão *play* é possível visualizar todas as alterações feitas ao poema, e a barra imediatamente à esquerda é uma barra de progresso que indica em que ponto da animação é que nos situamos (Figura 48). Esta versão foi entregue aos utilizadores/*testers*, e foi sobre esta versão que foram feitos os testes de usabilidade. Posteriormente foram feitas algumas alterações à *interface*, que serão apresentadas na secção dos testes de usabilidade, visto serem resultado deste processo.



Fig.46- Screenshots da última versão da aplicação.

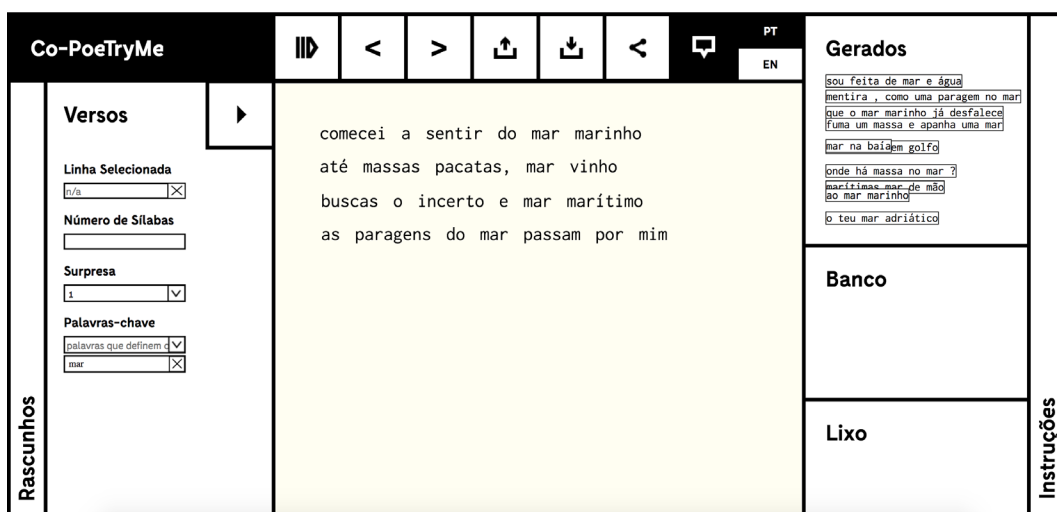
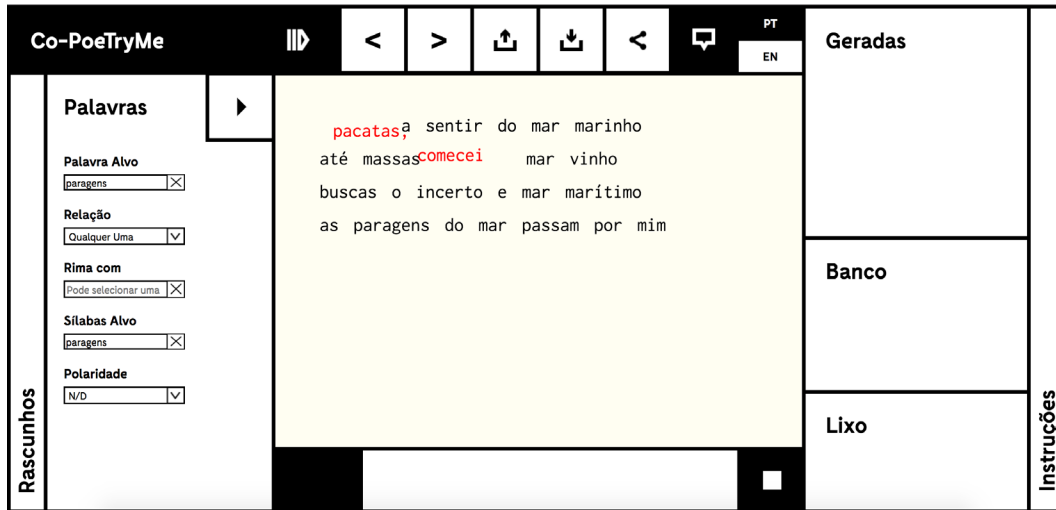


Fig.47- Screenshots da última versão da aplicação – representação dos conteúdos (neste caso versos) sugeridos pelo sistema.





**Fig.48-** Screenshot da última versão da aplicação – visualização da animação, neste caso, da troca da posição de duas palavras.

## 7 - Avaliação

Este capítulo descreve alguns aspetos que contribuem para a validação e avaliação da aplicação desenvolvida. Foram realizados testes de usabilidade, onde a utilização da aplicação por parte de um conjunto de utilizadores foi importante para detectar alguns problemas, posteriormente corrigidos.

Descreve-se ainda a utilização do sistema para gerar letras de música, por um grupo de investigadores na área da criatividade computacional. O capítulo termina com uma descrição do sistema desenvolvido enquanto plataforma criativa, à luz da classificação de Kantosalo (2016)

### 7.1 Testes de Usabilidade

Os testes de usabilidade, para além de permitirem identificar erros de design e implementação, permitem perceber que tarefas os utilizadores sentem mais dificuldade em desempenhar. Uma das maneiras mais comuns de testar a usabilidade de aplicações é através da disponibilização de uma versão beta da aplicação.

Esta versão da aplicação é testada, num ambiente supervisionado, por um conjunto de utilizadores. É-lhes dado um conjunto de tarefas a desempenhar e são apontados alguns valores: o tempo que o utilizador leva a desempenhar as tarefas; número de *clicks* necessários, o número de erros cometidos e ainda se o utilizador repetiu erros anteriores. São ainda recolhidos alguns dados que têm que ver com a satisfação do utilizador, bem como sugestões do mesmo. Estes valores permitem, em primeiro lugar por comparação com a performance do autor, verificar se as funcionalidades são facilmente identificáveis e intuitivas, bem como perceber se há alguma funcionalidade que tenha uma consequência que não é a esperada pelos os utilizadores.

Numa fase inicial, foi elaborado um conjunto de tarefas, listado na Figura 49, que envolve a utilização dos vários módulos da aplicação, bem como os vários tipos de interação disponíveis. O conjunto de tarefas a resolver foi dado aos utilizadores (Tabela 6), para que eles as tentassem resolver, enquanto os tempos de execução de cada tarefa eram cronometrados e os seus erros e sugestões apontadas (Tabela 7), num formulário igual ao da Figura 50.

### Lista de Tarefas:

- 1•Gere um Poema-Rascunho;
- 2•Troque a posição de duas palavras no poema-rascunho;
- 3•Troque a posição de duas linhas no poema-rascunho;
- 4•Altere uma palavra do poema-rascunho por uma escrita por si;
- 5•Desfaça a ação anterior;
- 6•Gere palavras relacionadas com uma palavra do poema-rascunho;
- 7•Substitua uma palavra do poema-rascunho por uma palavra sugerida pelo sistema;
- 8•Gere linhas com palavras de inspiração diferentes das usadas para gerar o poema-rascunho;
- 9•Adicione uma das linhas sugeridas ao poema-rascunho;
- 10•Elimine uma palavra do poema-rascunho;
- 11•Exporte o poema em formato “.txt”.

**Fig.49** - Conjunto de tarefas propostas aos utilizadores/testers.

Idade:  
Escolaridade:

	tempo	nr. clicks/ nr. erros	repetição de erros	satisfação	sugestões
1•					

**Fig.50** - Formulário preenchido aquando dos testes de usabilidade.

Para abranger um conjunto mais alargado de pessoas, alguns testes não puderam ser presenciais, e nesse caso, era pedido às pessoas que realizassem esse mesmo conjunto de tarefas e preenchessem o formulário elas próprias. Era-lhes ainda pedido que descrevessem a sua interação com a aplicação, nomeadamente que identificassem as tarefas onde tiveram mais dificuldades e sugestões para as suprir. Os testes presenciais foram filmados, tanto através da gravação de ecrã que permitem mostrar a utilização da aplicação, como através da gravação do utilizador, com o objectivo de captar expressões que revelassem satisfação, insatisfação ou frustração face à utilização de determinada ferramenta (Figura 51).

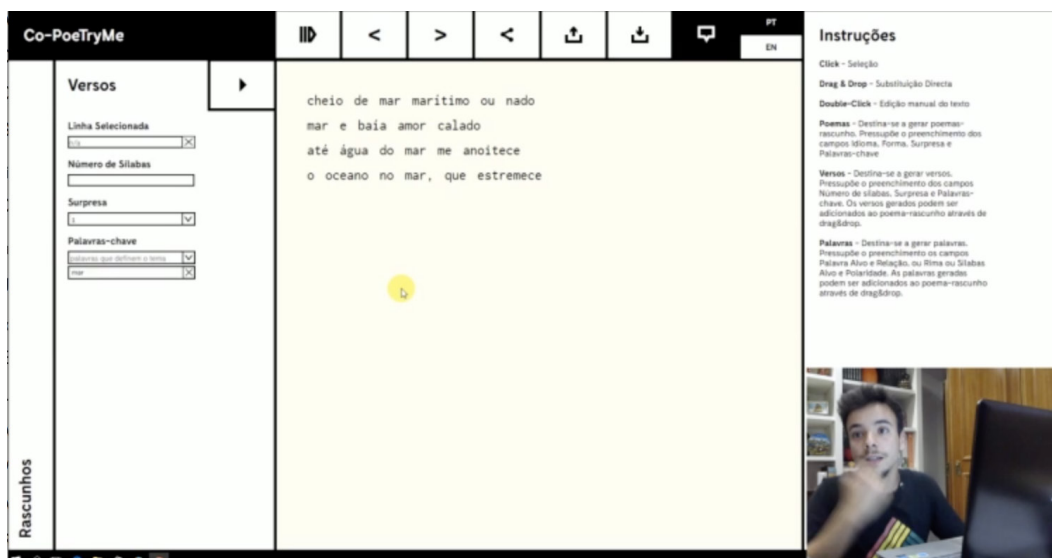


Fig.51- Frame de um dos vídeos dos testes de usabilidade efectuados

Analisando os resultados obtidos foi possível identificar alguns problemas na *interface* e na interação que dificultavam o processo de aprendizagem da aplicação para os utilizadores:

**Dificuldade 1** – A maioria dos utilizadores sentiu dificuldade na resolução da tarefa 6, que pede ao utilizador para gerar palavras relacionadas com determinada palavra do sistema. A principal dificuldade na resolução desta tarefa prendia-se com o facto de o utilizador não conseguir encontrar o módulo de geração de palavras, isto porque é necessário carregar numa palavra do poema-rascunho para este módulo aparecer. Esta foi a tarefa que consumia mais tempo aos utilizadores, nomeadamente para encontrar o módulo para a realização desta tarefa, e também foi nesta tarefa que os utilizadores cometeram mais erros, nomeadamente *click* em módulos que não levavam à resolução da tarefa pretendida.

**Dificuldade 2** – Alguns utilizadores, neste caso os mais jovens, carregaram em botões do teclado na tentativa de utilização de atalhos que não existiam, nomeadamente na tarefa 10, carregaram no *backspace* na tentativa de apagar a palavra seleccionada. Outro caso semelhante ocorreu na tarefa 4, quando alguns utilizadores carregaram na tecla *enter* na tentativa de editar uma palavra seleccionada.

**Dificuldade 3** – Alguns utilizadores, neste caso os com mais de 50 anos, tiveram algumas dificuldades na identificação dos símbolos utilizados, nomeadamente o símbolo de exportar na tarefa 11. No entanto, através da visualização das janelas de ajuda dos diversos botões, conseguiram efectuar a tarefa, embora lhes tivesse consumido mais tempo em comparação com utilizadores mais jovens. Esta maior dificuldade deve-se ao facto de estes utilizadores não estarem tão habituados a utilizar outras aplicações, *Web* ou para dispositivos móveis, e por isso não estão tão familiarizados com os símbolos utilizados.

**Dificuldade 4** – Alguns utilizadores tiveram dificuldade na tarefa 9, visto que o símbolo “+”, que representa a possibilidade de adicionar uma linha ao poema-rascunho, apenas aparece quando um verso é arrastado para a zona imediatamente abaixo da última linha. Foi ainda sugerido por dois utilizadores que o botão “+” aparecesse imediatamente ao arrastar um verso, mas com fundo branco e tipografia a negro. Quando o verso fosse arrastado para a zona imediatamente abaixo da última linha do poema-rascunho, o botão “+” alteraria a sua aparência para fundo negro e tipografia a branco, para representar a possibilidade de interação que está disponível. Esta sugestão poderia ser expandida para a adição de palavras a um verso do poema-rascunho, que embora não fosse uma das tarefas a resolver nos testes de usabilidade, acabou por ser identificada por alguns utilizadores aquando da realização da tarefa 7.

**Dificuldade 5** – Houve ainda a sugestão, por um utilizador, para que quando uma palavra ou verso sugerido fosse clicado, arrastado, ou o rato lhe passasse por cima, este passasse imediatamente para a frente de todas as outras palavras ou versos sugeridos, em vez de ter de se arrastar o fragmento para uma zona do módulo disponível para ler o seu conteúdo. Isto acontece porque os fragmentos sugeridos são adicionados por uma ordem, e visto serem opacos, encobrem-se parcialmente, e caso haja muitos conteúdos dentro do módulo, torna-se difícil “arranjar” um espaço vazio para arrastar o fragmento sugerido e conseguir ler o seu conteúdo.

Nr. Utilizador	Escolaridade	Idade
1	12º ano	62
2	9º ano	18
3	Bacharelato	65
4	Mestrado (MDM)	24
5	Mestrado (MDM)	24
6	Mestrado (MDM)	25
7	Licenciatura (LDM)	25
8	Licenciatura (LDM)	24
9	12º ano	24
10	12º ano	24
11	12º ano	23
12	12º ano	23
13	12º ano	26

**Tab. 6-** Lista de utilizadores cujos testes foram presenciais.

Nr. Tarefa	Média	Problemas Identificados	Nr. utilizadores	Alterações Feitas
1	26s	gerar poemas sem palavras de inspiração	3	nenhuma alteração, janela de erro foi suficiente
2	13s	<i>double-click</i> ao tentar arrastar	1	nenhuma alteração, erro não foi repetido
3	7s			
4	21s	carregar no <i>enter</i> para editar – <b>dificuldade 2</b>	3	acrescentou-se o atalho <i>enter</i> para editar texto
5	5s			
6	52s	difícil encontrar geração de palavras – <b>dificuldade 1</b>	10	novo botão de fragmentação do poema
7	11s	difícil ver palavras sugeridas – <b>dificuldade 5</b>	5	fragmentos sugeridos passam para a frente com <i>hover</i>
8	27s	geraram linhas com as mesmas palavras de inspiração	3	nenhuma alteração, janela de erro foi suficiente
9	26s	botão de adicionar linhas difícil de encontrar – <b>dificuldade 4</b>	8	alteração da ferramenta de adicionar linhas e palavras
10	12s	carregar no <i>backspace</i> para editar – <b>dificuldade 2</b>	5	acrescentou-se o atalho <i>backspace</i> para apagar texto
11	8s	simbologia não intuitiva – <b>dificuldade 3</b>	2	nenhuma alteração

**Tab. 7-** Erros presenciados nos testes de usabilidade.

## 7.2 Alterações Feitas à Aplicação

Visto que foram identificadas algumas lacunas na aplicação (apresentadas na secção acima) que dificultavam que os utilizadores a aprendessem, foram feitas algumas alterações à mesma com o objectivo de suprir essas mesmas lacunas:

· Em resposta à primeira dificuldade sentida pelos utilizadores – **dificuldade 1** – que se prende com o facto de ser necessário carregar num fragmento do tipo palavra para revelar o módulo de geração de palavras, foi incluído um novo botão que permite a alteração do tipo de fragmentação (Figura 52). Esta solução chegou a estar presente na primeira versão da aplicação (como mostra na secção 6.4), mas foi abandonada logo na segunda versão, porque havia o objectivo de que fosse o próprio texto a mostrar os tipos de interação possível. No entanto, através dos testes de usabilidade ficou claro que a interação com o texto não é o suficiente, e por isso foi adicionado um botão que permite alterar o tipo de fragmentação. Contudo, este botão é diferente do que estava disponível anteriormente. Depois de haver um poema-rascunho, e quando o módulo Rascunhos está no seu estado reduzido, surge um novo botão, lateral, que indica o tipo de fragmentação que não está seleccionada. Basta carregar neste novo botão para alterar o tipo de fragmentação.

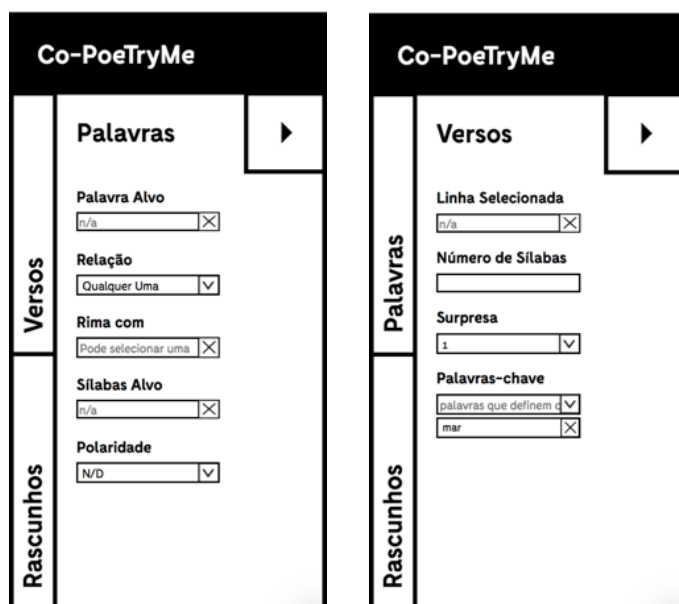


Fig.52 - Novo botão para alteração do tipo de fragmentação do poema.

Relativamente à segunda questão – **dificuldade 2** – que se prendia com o facto de o conjunto de atalhos disponíveis não ser tão extenso quanto o esperado pelos utilizadores, foram adicionados mais alguns atalhos com o objectivo de que utilizadores mais experientes consigam otimizar a utilização da aplicação:

- *Enter*: Quando um fragmento poético do poema-rascunho está seleccionado, seja ele do tipo verso ou linha, e a tecla *enter* é carregada, é aberto o modo de edição desse fragmento. Este tipo de interação já é utilizado em algumas aplicações, *Websites* e sistemas operativos.

- *Backspace* quando um fragmento está seleccionado: Quando um fragmento poético do poema-rascunho está seleccionado, seja ele um verso ou linha, e a tecla *backspace* é carregada, este fragmento é automaticamente eliminado, tornando o processo de eliminação de um fragmento poético mais rápido e célere, pois exige menos *clicks* para efectuar essa tarefa.

Relativamente à questão dos utilizadores mais velhos não estarem tão familiarizados com os símbolos utilizados – **dificuldade 3** – acredito que a possibilidade de os botões mostrarem um balão de fala que descreve a funcionalidade (caso as janelas de ajuda estejam ativadas), é o suficiente para que o utilizador perceba a ferramenta em questão. Isto porque os símbolos utilizados foram escolhidos pela ampla utilização em outras aplicações, e o facto de os alterar podia piorar a percepção dos mesmos por parte dos utilizadores mais jovens.

Visto que vários utilizadores sentiram dificuldade na utilização das ferramentas de adição tanto de versos ao poema-rascunho, como de palavras a determinada linha – **dificuldade 4** – foram feitas algumas alterações para que se tornasse mais perceptível que estas funcionalidades estão disponíveis. Quanto à adição de linhas, mal começamos a arrastar um dos versos sugeridos aparece um botão “+” por baixo do poema-rascunho que sugere a possibilidade de adicionar ali uma linha. Quando o rato passa por cima desse botão este muda a sua aparência, neste caso passa a ter fundo preto e texto a branco, para reforçar essa possibilidade de interação (Figura 53). De igual modo, para a adição de palavras, quando começamos a arrastar uma palavra sugerida, aparece um botão “+” à frente de cada linha. Caso o utilizador arraste a palavra sugerida para cima de um dos botões “+”, de igual modo à adição de linhas, o botão altera a sua aparência, com o mesmo objectivo de reforçar a possibilidade de interação (Figura 54).



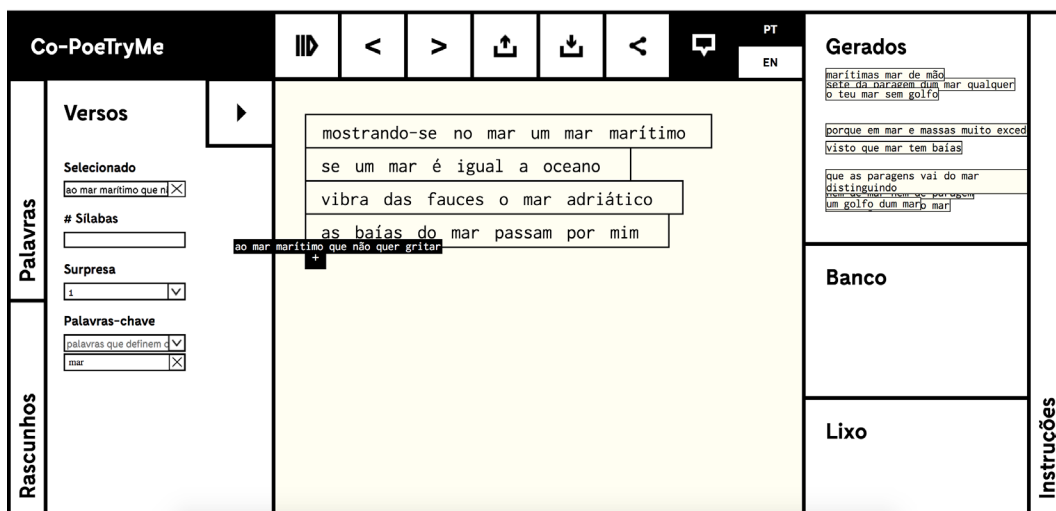


Fig.53- Nova representação da ferramenta de adição de linhas.

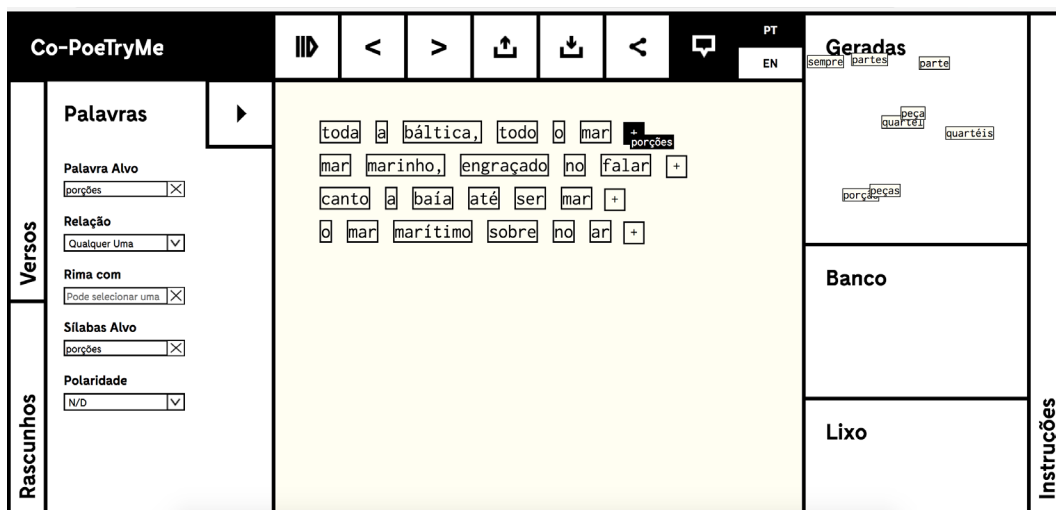


Fig.54- Nova representação da ferramenta de adição de palavras.

Quanto à última questão – **dificuldade 5** – a sugestão dada foi seguida à risca: quando são gerados fragmentos poéticos, sejam eles palavras ou linhas, são apresentados sobre a forma de aglomerado. No entanto, quando um dos fragmentos é arrastado ou quando o rato lhe passa por cima, ele passa imediatamente para a

frente dos outros para que o seu conteúdo seja visível imediatamente (Figura 55). Isto evita que o utilizador tenha que arrastar o fragmento sugerido para uma zona livre do módulo em que está para conseguir ler o seu conteúdo, tarefa essa que seria mais difícil quanto maior fosse o resultado do processo de geração.

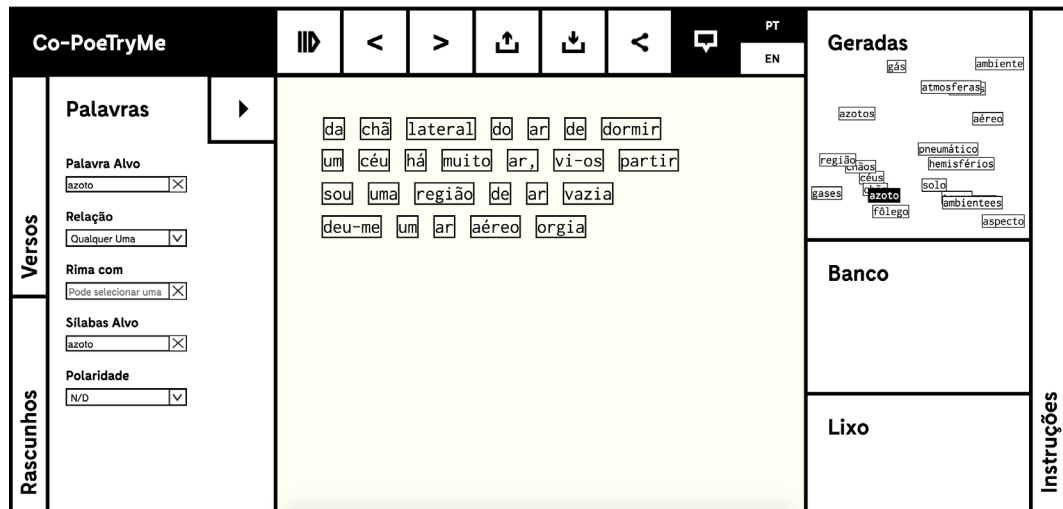


Fig.55 - Nova representação dos fragmentos poéticos sugeridos pelo sistema.

### 7.3 - Utilização por terceiros

O Co-PoeTryMe foi recentemente utilizado com o objectivo de produção de letras de música, por investigadores do Computer Engineering Department, Santa Clara University, na área da criatividade computacional e geração de música. O Co-PoeTryMe foi utilizado para a composição de letras que seriam dadas como *input* ao sistema AYSIA (Ackerman et al, 2017)(Cassion et al, 2017), para que o sistema criasse uma canção. Durante a utilização, o grupo de investigadores retirou algumas notas sumárias da utilização da aplicação, bem como do processo de transformação do poema-rascunho gerado no poema final. A partir de um dos poemas gerados chegou mesmo compor duas canções, que nos enviou sobre a forma de pauta (Figura 56).

Nas notas que nos foram enviadas é descrito o processo de criação de dois poemas. No primeiro foram usadas como palavras de inspiração as palavras-alvo:

*love, femme e kind*, e o resultado do processo de geração foi o seguinte poema:

*wisdom turns emotion, and science and love  
with kind categories every eye  
emotions and love and moon and sky  
and whether haters in love above*

untitled

Co-PoeTryMe Alysia, arr. Ai Nakamura  
♩ = 80

I am a tho - usand o bjects that love

through my pain and ple - asure

that e - ver love did strike

u - plift the tough and te - nder a - like

**Fig.56** -Pauta da canção criada, cuja letra foi criada no Co-PoeTryMe.

De seguida, a descrição de todos os passos efectuados até ao poema final:

“For creating this poem, I fed into Co-PoeTryMe and got a raw output of which I only manipulated two awkward places. The first was the word ‘categories’ in the middle of the second line. When I attempted to regenerate a replacement word, I had little return because it kept trying to return the same syllable length. I was able to manipulate the

line generation function into accepting a single word input and use a short phrase generated that way to replace it. The other modification came from the word “haters” in the final line. I was able to use the word generating function and luckily got “hatred” on the first try, allowing me to keep the contrasting feel of the last line.”

Embora a descrição fornecida não vá muito de encontro à experiência de utilização, no sentido de tentar descrever as dificuldades sentidas na utilização da *interface*, tempo de aprendizagem da *interface*, etc, o grupo de investigadores revela o domínio das principais ferramentas da aplicação, que são a geração de poemas-rascunho, versos e linhas. Revela ainda um conhecimento para além do básico quando diz que recebeu um *feedback* pequeno, neste caso da ferramenta de geração de palavras, por ter a opção o número de sílabas preenchida. Na descrição do processo do segundo poema, diz ainda que reordenou as linhas, neste caso para [3,2,1,4], o que revela o domínio de mais uma ferramenta. Neste caso, em que a aplicação foi utilizada por um grupo de investigadores de outra nacionalidade que não a portuguesa, foi importante o facto de a alteração do idioma de toda a *interface*, quer os elementos estáticos, quer os dinâmicos, ser feita através de um simples *click* num botão.

## 7.4 Co-PoeTryMe e a co-criatividade

Avaliando o Co-PoeTryMe enquanto plataforma co-criativa segundo as categorizações de Lubart (2005) e de Mary Lou Maher (2012), esta aplicação funciona como um colega que pode ser criativo ou potenciar a criatividade do autor e sua expressão. As ideias geradas pelo sistema são sujeitas ao critério de avaliação do utilizador e podem ou não ser integradas no artefacto final. Isto faz com que o Co-PoeTryMe encaixe nas últimas categorias de ambas as categorizações acima mencionadas: “O sistema funciona como um colega que pode ser criativo, contribuindo com ideias no diálogo com os utilizadores” (Lubart, 2005) e “O sistema gera ideias que são avaliadas e podem ser integradas, ou não, no artefacto final” (Maher, 2012).

Segundo a categorização de Kantosalo e Toivonen (2016), Co-PoeTryMe é um agente que se se enquadra na co-criatividade dividida por tarefas. O sistema é capaz, através da definição de um conjunto de dados de entrada acerca domínio, gerar um poema sobre esse mesmo domínio. O sistema neste caso toma o papel

de gerador de conceitos enquanto que o utilizador assume o papel de definidor do domínio. Na alteração de palavras/expressões, há novamente um pedido para que o utilizador defina o espaço de palavras a operar. No entanto, após gerado o poema-rascunho, o utilizador pode editá-lo. Esta edição pode ser feita de duas maneiras: através de substituição directa de palavras ou linhas, ou por pedido ao sistema que gere uma nova palavra ou linha que cumpra determinados requisitos. Ou seja, tanto o computador como o humano estabelecem regras de iteração ao espaço de artefactos possíveis. O sistema é completo porque é capaz de gerar e avaliar os seus artefactos através de uma função de *fitness*, embora a avaliação seja incompleta – apenas pontua a métrica e a rima. Esta é uma das principais razões que motivou a criação da plataforma co-criativa assente no PoeTryMe – a função *fitness* não consegue pontuar os conteúdos propriamente ditos. O utilizador serve como avaliador do conteúdo/significado e conduz a geração de acordo com as suas intenções. A avaliação dos conceitos gerados é feita pelos dois agentes (a níveis diferentes), ainda que a validação final esteja ao cargo do utilizador, que decide quando parar o processo de iteração do poema.

Para além disso, os problemas que Kantosalo e Toivonen (2016) identificaram no âmbito da criação de aplicações de co-criatividade alternada também se podem aplicar ao Co-PoeTryMe. Para evitar os problemas de incompatibilidade tanto universal como conceptual, há a preocupação ao nível do design da aplicação para os evitar. Neste caso, a *interface* do Co-PoeTryme limita o *input* do utilizador a um conjunto de opções, com o intuito de não deixar que o utilizador introduza conceitos que o agente não domina. Por exemplo, não permite introduzir uma métrica diferente das disponíveis, apenas permite que utilizador escolha uma das várias dominadas pelo sistema. No que toca à discordância artística, a aplicação Co-PoeTryMe dá mais importância à avaliação do utilizador, visto que o sistema é incapaz de avaliar o significado e a intenção do texto produzido. Então a avaliação do utilizador sobrepõe-se à do sistema, visto ser uma avaliação mais completa. Quanto à impotência generativa, o sistema não conhecendo um conceito introduzido, faz uma procura aleatória a todo o universo. Assim nunca entra num estado em que deixa de produzir artefactos, embora, segundo Kantosalo e Toivonen (2016), esta solução não seja a ideal para cenários co-criativos.

## 8. Conclusão e Prespectivas Futuras

Este documento descreveu o estudo efectuado acerca dos temas cruciais desta tese, entre eles a poesia, a co-criatividade e o design de aplicações. Pretende-se também descrever a aplicação criada, desde a ideia inicial até a aplicação final, passando tanto por questões de design como de implementação.

A principal contribuição desta tese é a disponibilização de uma ferramenta co-criativa de apoio à composição de poesia e letras de música. Esta aplicação pode ser acedida por qualquer computador com acesso à Internet, sem haver a necessidade de fazer o *download* de nenhum ficheiro e suprimindo todos os problemas de compatibilidade que poderiam existir. A aplicação foi desenvolvida através da utilização das tecnologias HTML, CSS e Javascript. Para além disso, a aplicação apresenta um design intuitivo e fácil de aprender e já foi testada junto de alguns utilizadores, o que significa que alguns erros já foram corrigidos, nomeadamente a reformulação de algumas ferramentas cuja representação e utilização não era tão fácil de aprender/usar.

Para além disso, esta tese traz algumas contribuições complementares, nomeadamente a transmissão do estudo efectuado sobre os temas abordados: , e ainda uma análise de plataformas existentes na área da co-criatividade e da criatividade computacional. Para além disso, as necessidades que foram surgindo contribuíram para o desenvolvimento de novas funcionalidades na API do PoeTryMe, neste caso implementadas por Gonçalo Oliveira. Durante este trabalho foi ainda escrito um artigo científico (Gonçalo Oliveira et al., 2017) sobre as limitações do PoeTryMe, as novas funcionalidades da API e o Co-PoeTryMe, aceite para publicação no 2nd Symposium on Problem-solving, Creativity and Spatial Reasoning in Cognitive Systems.

Posto isto, julgo que o resultado deste projecto foi para além da proposta inicial, visto que foi desenvolvido um estudo mais aprofundado da co-criatividade computacional, foi desenvolvida uma identidade visual para a aplicação e ainda foram adicionadas várias funcionalidades não estavam previstas nem na proposta de dissertação, nem na proposta intermédia, tais como: importação e exportação de poemas, tanto para um formato “.txt” que permite voltar a ser importado, como para uma imagem que preserva todas as representações das alterações feitas; as próprias representações do processo co-criativo, tanto a estática como a dinâmica; a partilha dos poemas gerados nas redes sociais. Contudo, o desenvol-

vimento destas funcionalidades extra consumiu o tempo necessário para desenvolver um dos objectivos previstos na proposta intermédia, relacionado com o desenvolvimento de um *layout* responsivo para a aplicação.

Visto que cada vez mais os acessos à Internet são feitos através de dispositivos móveis, é crucial o desenvolvimento de um design responsivo, visto que atualmente a aplicação é apenas fluída, isto é, adapta-se aos diversos tamanhos dos ecrãs de computador. Esta tarefa envolverá repensar a estrutura da aplicação, nomeadamente na representação das diversas ferramentas e tipos de interação.

No âmbito da representação do processo co-criativo, e de maneira a aumentar a fidelidade de representação, seria vantajosa a utilização de um algoritmo capaz de identificar as diferenças entre dois textos. Isto é, caso o utilizador, no âmbito da edição de um verso, apenas altere uma palavra, na representação do processo co-criativo, toda a linha aparece como tendo sido editada. No entanto, através da utilização de um algoritmo capaz de verificar onde houve alterações, neste caso entre a linha inicial e a editada pelo utilizador, seria possível uma maior precisão na representação desta ação.

Outra funcionalidade que poderia ser melhorada no futuro seria a de definição de formas para o poema. Neste momento limita-se a um conjunto pré-definido ou uma sequência de linhas com o mesmo número de sílabas. No futuro poder haver mais liberdade, no entanto isso também implicará alterações à API.

Também seria importante repetir os testes de usabilidade, de forma a verificar se as alterações melhoraram a experiência de utilização e se não foram introduzidos novos problemas.

Para além disso há a intenção de, num futuro próximo, escrever um artigo completamente dedicado ao Co-PoeTryMe, e de colaborar com o grupo da Maya Ackerman, na Computer Engineering Department, Santa Clara University.

## 9. Anexos

Nesta secção é apresentado o manual da API do sistema PoeTryMe, desenvolvida por Gonçalo Oliveira, durante a realização deste trabalho. Vão ser ainda indicadas as funcionalidades existentes e disponíveis na aplicação actualmente implementada, o TryMe, bem como as funcionalidades que foram desenvolvidas, também por Gonçalo Oliveira, no âmbito da realização deste projecto.

### Geração de Poemas:

Está funcionalidade já existia e é a que se encontra na aplicação actualmente disponível do PoeTryMe, que serve para gerar poemas através de definição de um conjunto inicial de parâmetros.

**URL:** <http://poetryme.dei.uc.pt/PoeTryMe/rest/poetry>

### Parâmetros:

·Lingua - lang:

- “en” para Inglês;
- “pt” para Português;
- “es” para Espanhol.

·Forma - form:

- “10-4” para quadra;
- “10-2” para dístico;
- “5-7-5” para haiku;
- “sonnet” para soneto;
- “alecrim” para a métrica da música “Alecrim”;
- “barquinho” para a métrica da música “O Barquinho”;
- outras músicas;
- Personalizada - na form nr. de versos”x”nr. de sílabas (ex: 4x8).

·Sementes - seeds:

- conjunto de palavras separadas por “,”.

·Factor Surpresa - surp:

- valor entre 0.1 e 0.



**Exemplo:****Pedido:**

<http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry?lang=pt&form=10-4&seeds=mar+rio&surp=0.0>

**Resposta:**

```
{form: "quadra_10.est", language: "pt", score: "-1.0",  
seeds: ["águas", "rio", "mar"], surprise: "0.0",  
text: "tenho um tear de massa de mar\n  
a água aquosa de brigar\n  
água serve para hidromancias\n  
hidrotérmico à margem da água\n\n"}}
```

**Geração de Palavras:**

Está foi uma das funcionalidades desenvolvidas por Gonçalo Oliveira no contexto da realização deste projecto, e gera uma lista de palavras que cumpram determinados requisitos definidos pelo utilizador.

**URL:** <http://poetryme.dei.uc.pt/PoeTryMe/rest/poetry/words>

**Parâmetros:****·Lingua – lang:**

- “en” para Inglês;
- “pt” para Português;
- “es” para Espanhol.

**·Palavra – word:**

- palavra a substituir.

**·Relação – rel:**

- “any” para qualquer relação;
- “anton” para antónimo;
- “hyper” para hiperónimo;
- “hypon” para hipónimo;
- “cohyp” para co-hipónimo;
- “synon” para sinónimo;
- “none” para nenhuma.

- Rima** – rhyme:
  - fonema ou palavra que termine com o fonema desejado.
  
- Número de Sílabas** – nsyl:
  - número de sílabas da palavra a gerar.
  
- Polaridade** – pol:
  - “-1” para negativa;
  - “1” para positiva.

### **Exemplo:**

#### **Pedido:**

<http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetrywords/?lang=pt&word=apertar&rel=any&syl=apertar>

#### **Resposta:**

```
{language:"pt", polarity:"N/A",
related::["consumir", "devorar", "acochar", "encurtar", "imprensar", "limitar", "aper-
tos", "abraçar", "abrochar", "resumir", "estringir", "retrair", "aperto", "apoucar", "am-
putar", "confranger", "apressar", "desgostar", estritar", "tarraxar"],
relation:"any", syllablesMatch:"apertar", word:"apertar"}
```

#### **Geração de Linhas:**

Está foi outra das funcionalidades desenvolvidas por Gonçalo Oliveira no contexto da realização deste projecto, e serve para gerar uma linha, de acordo com um conjunto de parâmetros pré-definidos pelo utilizador.

**URL:** <http://poetryme.dei.uc.pt/PoeTryMe/rest/poetry/line>

#### **Parâmetros:**

- Lingua** – lang:
  - “en” para Inglês;
  - “pt” para Português;
  - “es” para Espanhol.
  
- Sementes** – seeds:
  - conjunto de palavras separadas por “,”.

- Número de Sílabas – nsyl:
  - número de sílabas da palavra a gerar.
  - Factor Surpresa – surp:
  - valor entre 0.1 e 0.
  
- Número de fases de geração – bestof:
  - valor inteiro maior ou igual a 1.

**Exemplo:**

**Pedido:**

<http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry/line?lang=pt&nsyl=5&seeds=fogo&bestof=10>

**Resposta:**

{form: “1-liner”, language: “pt”, score: “0.0”, seeds: “fogo”, surprise: “0.0”, text: “meu fogo , meu lar”}

**Avaliação da métrica de um poema:**

Está foi outra das funcionalidades desenvolvidas por Gonçalo Oliveira no contexto da realização deste projecto, e serve para avaliar poemas-rascunho em relação a determinada métrica, definida pelo utilizador, bem como para identificar a presença de rima.

**URL:** <http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry/score>

**Parâmetros:**

- Lingua – lang:
  - “en” para Inglês;
  - “pt” para Português;
  - “es” para Espanhol.
  
- Forma – form:
  - “10-4” para quadra;
  - “10-2” para dístico;
  - “5-7-5” para haiku;
  - “sonnet” para soneto;
  - “alecrim” para a métrica da música “Alecrim”;

- “barquinho” para a métrica da música “O Barquinho”;
- outras músicas:
- Personalizada – na form nr. de versos”x”nr. de sílabas (ex: 4x8).

- **Poema** – text:
- O poema a avaliar

### **Exemplo:**

#### **Pedido:**

[http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry/score?lang=pt&form=10-4&text=as baías do mar passam por mim \n de paragem e mar a mesma roupa \n toda a paragem, todo o mar \n meu golfo, meu olhar, meu olhar](http://poetryme.dei.uc.pt:8080/PoetrymeWeb/rest/poetry/score?lang=pt&form=10-4&text=as%20ba%C3%ADas%20do%20mar%20passam%20por%20mim%20%5Cn%20de%20paragem%20e%20mar%20a%20mesma%20roupa%20%5Cn%20toda%20a%20paragem,%20todo%20o%20mar%20%5Cn%20meu%20golfo,%20meu%20olhar,%20meu%20olhar)

#### **Resposta:**

{form:“10-4”, language:“pt”, score:“-1.0”,  
text:“as baías do mar passam por mim \n de paragem e mar a mesma roupa \n toda a paragem, todo o mar \n meu golfo, meu olhar, meu olhar \n”

## 10 Bibliografía/Webgrafía

### Textos:

Ackerman, M., Locker, D. (2017). Algorithmic Songwriting with ALYSIA. International Conference on Computational Intelligence in Music, Sound, Art and Design (EvoMUSART 2017). Amsterdam, The Netherlands.

Albert, W., & Tullis, T. (2013). Measuring the user experience: collecting, analyzing, and presenting usability metrics. Newnes.

Boden, M. (1992). The creative mind. London: Abacus.

Boden, M. A. (2004). The creative mind: Myths and mechanisms. Psychology Press.

Cardoso, A., Veale, T., & Wiggins, G. A. (2009). Converging on the divergent: The history (and future) of the international joint workshops in computational creativity. *AI Magazine*, 30(3), 15.

Cassion, C., Ackerman, M., Locker, D., Palkki, J. (2017). Songwriting with ALYSIA & Emily Dickinson. In Proceedings of 5th International Workshop on Musical Metacreation (MUME 2017). Atlanta, USA.

Charnley, J., Colton, S., Llano, M. T., & Corneli, J. (2016). The FloWr Online Platform: Automated Programming and Computational Creativity as a Service. In Proceedings of the Seventh International Conference on Computational Creativity, ICC3.

Charnley, J., Colton, S., & Llano, M. T. (2014). The FloWr framework: Automated flowchart construction, optimisation and alteration for creative systems. In Proceedings of the 5th international conference on computational creativity, 9.

Colton, S., & Wiggins, G. A. (2012). Computational creativity: The final frontier?. In Proceedings of the 20th European conference on artificial intelligence (pp. 21-26). IOS Press.

Colton, S., de Mántaras, R. L., & Stock, O. (2009). Computational Creativity: Coming of Age. *AI Magazine*, 30(3), 11-14.

El Bolock, A. (2014). Automatic Poetry Generation Using CHR.

Gero, J. S. (2000). Computational models of innovative and creative design processes. *Technological forecasting and social change*, 64(2), 183-196.

Gervás, P. (2001). An expert system for the composition of formal spanish poetry. *Knowledge-Based Systems*, 14(3), 181-188.

Gervás, P. (2002). Exploring quantitative evaluations of the creativity of automatic poets. In *Second workshop on creative systems, approaches to creativity in Artificial Intelligence and Cognitive Science (ECAI)*.

Gervás, P. (2013). Computational modelling of poetry generation. In *Artificial Intelligence and Poetry Symposium, AISB Convention*.

Gonçalo Oliveira, H. (2012). PoeTryMe: a versatile platform for poetry generation. In *Proceedings of the ECAI 2012 Workshop on Computational Creativity, Concept Invention, and General Intelligence, C3GI 2012*.

Gonçalo Oliveira, H. (2016). Poe, now you can TryMe: Interacting with a poetry generation system. In *Proceedings of the Demonstration Session of PRO-POR'2016, 12th International Conference on Computational Processing of the Portuguese Language*.

Gonçalo Oliveira, H., Mendes, T., Boavida, A. (2017) Towards finer-grained interaction with a Poetry Generator. In *Proceedings of 2nd Symposium on Problem-solving, Creativity and Spatial Reasoning in Cognitive Systems (ProSocrates 2017)*. Delmenhorst, Germany. CEUR.

Hervás, R., Alves, A. O., Oliveira, H. G., Xiao, P., Linkola, S., Toivonen, H., ... & Lavrac, N. (2016). Computational creativity infrastructure for online software composition: A conceptual blending use case. In *Proceedings of the Seventh International Conference on Computational Creativity*.

[https://pt.wikipedia.org/wiki/Poesia#cite\\_note-3](https://pt.wikipedia.org/wiki/Poesia#cite_note-3)

Kantosalo, A., Toivanen, J. M., Xiao, P., & Toivonen, H. (2014). From isolation to involvement: Adapting machine creativity software to support human-computer co-creation. In *Proceedings of the Fifth International Conference on Computational Creativity* (pp. 1-8).

- Kantosalo, A. A., Toivanen, J. M. & Toivonen, H. T. T. (2015). Interaction Evaluation for Human-Computer Co-Creativity : A Case Study. In Proceedings of the Sixth International Conference on Computational Creativity.
- Kantosalo, A., & Toivonen, H. (2016). Modes for Creative Human-Computer Collaboration: Alternating and Task-Divided Co-Creativity. In Proceedings of the Seventh International Conference on Computational Creativity.
- Krug, S. (2005). Don't Make Me Think! A Common Sense Approach to Web Usability. Indianapolis, IN: New Riders, 3.
- Liapis, A., & Yannakakis, G. N. (2016). Boosting Computational Creativity with Human Interaction in Mixed-Initiative Co-Creation Tasks.
- Lubart, T. (2005). How can computers be partners in the creative process: classification and commentary on the special issue. *International Journal of Human-Computer Studies*, 63(4), 365-369.
- Lupton, E. (2014). *Type on Screen: A Critical Guide for Designers, Writers, Developers, and Students*. Chronicle Books.
- Maher, M. L. (2012). Computational and collective creativity: Who's being creative. In *International Conference on Computational Creativity*.
- Manurung, H. (2004). An evolutionary algorithm approach to poetry generation. Ph.D. dissertation, University of Edinburgh.
- Misztal-Radecka, J., & Indurkha, B. (2016). A blackboard system for generating poetry. *Computer Science*, 17(2), 265.
- Newell, Shaw, & Simon, H. A. (1962). The processes of creative thinking. In H. E. Gruber & M. Wertheimer (Eds.), *Contemporary approaches to creative thinking*. Atherton Press, New York
- Nielsen, J. (1994). Usability inspection methods. In *Conference companion on Human factors in computing systems* (pp. 413-414). ACM.

Queneau, R. (1961). 100.000.000.000.000 de pomes. Gallimard Series. Schoenhof's Foreign Books, Incorporated.

Ritchie, G. (2007). Some empirical criteria for attributing creativity to a computer program. *Minds and Machines*, 17(1), 67-99.

Rhodes, M. (1961). An analysis of creativity. *Phi Delta Kappan*, 42

Runco, M.A. (2007). *Creativity theories and themes. Research, development and practices.* Elvise Academic Press, San Diego

Shneiderman, B., & Plaisant, C. (1998). Designing the user interface: Strategies for effective human-computer interaction. *ACM SIGBIO Newsletter*, 9(3), 6.

Tobing, B. C. L., & Manurung, R. (2015). A chart generation system for topical metrical poetry. In *Proceedings of the Sixth International Conference on Computational Creativity June* (p. 308).

Toivanen, J., Toivonen, H., Valitutti, A., & Gross, O. (2012). Corpus-based generation of content and form in poetry. In *Proceedings of the Third International Conference on Computational Creativity*.

Ventura, D. (2016). Mere Generation: Essential Barometer or Dated Concept. In *Proceedings of the Seventh International Conference on Computational Creativity, ICC3*.

Wiggins, G. A. (2006). A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems*, 19(7).

Yannakakis, G. N.; Liapis, A.; and Alexopoulos, C. (2014). Mixed-initiative co-creativity. In *Proceedings of the ACM Conference on Foundations of Digital Games*.

**Imagens:**

<http://akadesign.ca/wp-content/uploads/2015/01/FRIDGE-MAGNET-WORD-TUTORIAL.jpg>

<https://s-media-cache-ak0.pinimg.com/originals/c3/1b/3c/c31b3c0f455a-104153b11bd02ae4c1d3.png>



**Tabelas:**

<http://www.enago.com.br/s%C3%ADmbolos-de-edi%C3%A7%C3%A3o-%C3%A0-m%C3%A3o.htm>