Universidade de Évora

Escola de Ciências e Tecnologia

Mestrado em Engenharia Informática

Editor de Layout para Clientes Multimédia

Mário Silva Gusmão

Orientador

Teresa Cristina de Freitas Gonçalves

Évora, Abril 2012

Mestrado em Engenharia Informática

Editor de Layout para Clientes Multimédia

Mário Silva Gusmão

Orientador
Teresa Cristina de Freitas Gonçalves

Sumário

A existência de mecanismos de comunicação ágeis com o público de massas, adequados e configuráveis de acordo com necessidades específicas de negócio/sector de atividade, são fundamentais no processo de promoção de uma entidade, serviço ou produto.

De forma a minimizar o tempo de resposta a estímulos/necessidades de negócio é fundamental que o processo produtivo requeira a intervenção do mínimo de utilizadores/valências, e se possível, limitado ao próprio gestor do negócio.

As soluções de *digital signage* apresentam-se como mecanismos de comunicação apelativos e visuais para os quais é necessária a criação de uma *framework* que possibilite a sua gestão e dinamização de forma simples, pela composição de ecrãs de informação, compostos por diferentes tipos de controlos multimédia.

A presente dissertação apresenta uma solução, tanto na sua vertente de composição como de interpretação e apresentação dos conteúdos, para a composição de ecrãs de informação aplicados a ambientes *digital signage*.

A solução apresentada consiste num Editor para a composição de conteúdos, num Visualizador para a sua interpretação, apresentação e validação, assim como num Serviço Web para gerir e armazenar os conteúdos e os seus recursos.

Layout Editor for Multimedia Clients

Abstract

Agile communication mechanisms applied to general public, suitable and shaped according to specific business needs, are essential in the process of promoting an entity, service or product.

In order to minimize reaction time to business stimuli/needs, the productive process needs a fast intervention requiring a minimum of users/resources and, if possible, limited to the business manager himself.

Digital signage solutions are presented as appealing visual communication channels, for which it is necessary the creation of a framework that may allow its management and its use in a simple way, through the composition of information screens, made up of different types of multimedia controls.

This dissertation presents a solution for the composition of information screens applied to digital signage environments, not only for their composition but also in their interpretation and content presentation.

The solution consists of an Editor for content composition, a Visualizer for content interpretation, presentation and validation, as well as a Web Service for content management and related resource storage.

A todos os que me apoiaram

Agradecimentos

Para Ana Pissarro pela sua paciência, apoio e compreensão.

Para a minha mãe, Graça Gusmão, e o meu pai, Francisco Sátiro, que fizeram os possíveis para eu seguir nesta jornada, apoiando-me sempre em tudo o que precisei.

Para a minha orientadora, Professora Teresa Gonçalves, pois sem o seu apoio e disponibilidade total para esclarecer dúvidas, orientar e guiar, este trabalho não teria sido possível de realizar.

Para Ricardo Raminhos, pelo seu empenho, visão e motivação, pois foi sem dúvida um dos fatores mais importantes no decurso do projeto.

Para Pedro Tempera, Sérgio Agostinho, Jorge Salitre, Nuno Miranda, Vasco Carvalho, David Caeiro e Gustavo Arranhado, pela paciência e disponibilidade no esclarecimento de dúvidas.

Para Rita Valente, pelo *layout* e ajuda nas regras de usabilidade das aplicações desenvolvidas.

Para a Viatecla e a sua equipa, que me acolheram e me deram a oportunidade de realizar este projeto.

A todos vós, o meu muito obrigado!

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Devido à enorme quantidade de empresas existentes, é fundamental um reconhecimento da marca para que o negócio se possa expandir. Uma das formas mais eficazes para se conseguir este reconhecimento, é fazer-se a publicidade da marca e dos produtos associados.

O vídeo é a ferramenta que as empresas podem utilizar quando preparam a sua estratégia de marketing. A sua eficácia deve-se ao facto do cérebro humano empregar cerca de 40% do córtex para processar vídeo, o que significa que os estímulos visuais são os mais bem recebidos [8]. O vídeo não só atrai um público maior do que o tradicional conteúdo estático, mas também é um meio de publicidade atrativo, já que os anúncios de vídeo são considerados muito mais envolventes.

O *digital signage* e a IPTV[1] são formas de transmitir conteúdos de vídeo com a possibilidade de facilmente alternar entre os conteúdos, obtendo assim a sua dinamização. As vantagens da publicidade por via de *digital signage* são enormes, permitindo que os conteúdos publicitados estejam em constante rotatividade, contribuindo assim para o fator novidade e conservando o fator apelativo da publicidade.

A utilização de *digital signage* e IPTV dentro das empresas apresenta também outras vantagens, tais como: difundir mensagens de formação técnica e de treino para os funcionários, permitir que os visitantes da empresa assistam a informação sobre a empresa, ser uma via de boas vindas na entrada da empresa, tanto para funcionários como para visitantes, difundir diferentes informações conforme o sector da empresa, de acordo com os trabalhos que se realizam em cada sector.

Outra grande área de utilização do *digital signage* é a elaboração de menus digitais na

[1] Do Inglês, *Internet Protocol television*

área da restauração uma vez, que os menus digitais possibilitam a existência de conteúdos dinâmicos e atrativos para o cliente.

O tempo de resposta a uma oportunidade é um fator decisivo quando se pretende adquirir vantagem sobre a concorrência. A melhor forma de responder a uma oportunidade é a criação e a publicação de um conteúdo em *digital signage* ou IPTV, sendo fundamental a existência de ferramentas que agilizem a sua criação e distribuição.

1.0.1 Motivação

Cada vez mais existe o requisito de ferramentas que permitam aos gestores de informação (sem conhecimento informático específico) disponibilizar conteúdos dinâmicos de uma forma fácil e ágil. Outro requisito é que essas ferramentas sejam intuitivas e de fácil utilização, com um foco importante na ergonomia e usabilidade.

Com esta ideia em mente, surgiu a necessidade do desenvolvimento de uma ferramenta informática que criasse ecrãs de informação de uma forma fácil e sem o recurso a programadores informáticos. Essa ferramenta deve ser intuitiva para que possa ser utilizada também por utilizadores básicos.

Os ecrãs de informação permitem apresentar visualmente informação de forma genérica, podem ser dirigidos a um grupo pequeno ou a um grupo grande de espetadores e podem ser apresentados em ecrãs pequenos ou em ecrãs de maior dimensão. Um ecrã de informação pode conter mais do que uma fonte de informação, e juntar diversos tipos de informação de forma a aumentar a experiência do utilizador numa lógica de agregação de conteúdo.

Os ecrãs de informação podem conter animações, vídeos, *layouts* apelativos, de forma a otimizarem a comunicação de informação, ou a melhor captarem a atenção do espetador. A divisão de um ecrã em várias secções permite a transmissão de várias informações ao mesmo tempo, mas de uma forma complementar, a obtenção de mais pontos que captem a atenção do espetador e a criação de um ecrã mais rico.

O projeto foi desenvolvido em parceria com a Viatecla [33] que já possuía uma solução para a distribuição e gestão de conteúdos de *digital signage* e estava interessada no desenvolvimento de uma ferramenta que agilizasse a criação dos conteúdos.

1.0.2 Objetivos

O presente projeto pretende apresentar uma solução, tanto na sua vertente de composição como de interpretação e apresentação dos conteúdos, para a composição de ecrãs de informação aplicados a ambientes *digital signage*. O seu desenvolvimento foi orientado pelos principais requisitos que se apresentam de seguida[2]:

[2] A enumeração completa dos requisitos encontra-se na Secção 4.1

- Permitir que o utilizador possa salvar o seu trabalho e possa retorna-lo;
- Ter uma forma de visualizar, agregar e navegar nos elementos que estão associados à área de trabalho ativa;
- Permitir que o utilizador possa interagir com os objetos através do rato para os mover ou redimensionar;
- Possibilitar ao utilizador pré-visualizar o trabalho realizado para identificar possíveis erros ou futuras melhorias;
- Integrar a solução em ambiente FutureboxTv (WEB e *Digital Signage*) com conteúdos especializados;
- Integrar a solução em *BackOffice*.

A solução necessita de ser acedida através de diversas localizações e visualizada através de dispositivos diferentes (Televisões, PCs). Para tal, será necessário desenvolver uma ferramenta que será integrada em *BackOffice* da solução de *digital signage*, disponibilizada em ambiente web, de forma a estar integrada na área de administração da solução.

Os ecrãs criados pela ferramenta desenvolvida serão integrados na plataforma FutureboxTv (solução de *digital signage* e Internet/WEB TV da Viatecla), que neste momento não oferece forma de criar ecrãs sem o recurso a um informático.

1.0.3 Abordagem proposta

O desenvolvimento de aplicações cliente multimédia (vídeo, animação, texto; estáticos e dinâmicos) apresenta uma série de desafios. Um deles é o controlo do *layout* das aplicações com as quais o utilizador final interage. Cada vez mais há uma maior exigência dos clientes poderem personalizar o *layout* das aplicações visualmente, sem a intervenção de um especialista (*designer* e/ou *developer*). Além disso, a possibilidade de poderem rapidamente pré-visualizar o aspeto final das aplicações é extremamente importante.

É possível dividir este trabalho em 3 grandes áreas: a criação de ecrãs de informação, a visualização de ecrãs de informação e a gestão dos recursos dos ecrãs. Cada uma destas áreas deu origem a uma aplicação.

A área de criação de ecrãs de informação deu origem ao Editor, que é uma aplicação que contém um conjunto de objetos que o utilizador pode utilizar para a criação dos conteúdos. Cada objeto contém um conjunto de propriedades que permitem a sua personalização.

A área de visualização de ecrãs de informação deu origem ao Visualizador. Este está integrado no Editor para permitir aos utilizadores executarem o trabalho realizado. Irá também estar integrado na FutureboxTv (nos ambientes web e *digital signage*) onde os conteúdos finais são disponibilizados.

A área de gestão de recursos deu origem a um Serviço Web responsável por armazenar os projetos e gerir os recursos. Sempre que o Editor necessita de recursos tem de pedir esta informação ao Serviço Web.

1.0.4 Principais contribuições

Este trabalho concretiza uma solução para a composição e interpretação de conteúdos *digital signage*. Durante o seu desenvolvimento criou-se uma Linguagem Declarativa e 3 aplicações: um Editor, um Visualizador e um Serviço Web.

A Linguagem Declarativa foi desenvolvida para guardar as definições dos projetos e as definições dos objetos do projeto. Como o Editor e o Visualizador são duas aplicações desagregadas, a Linguagem Declarativa permite a comunicação entre ambos. O facto de a linguagem ser declarativa permite uma maior liberdade quer na sua composição (apenas a lógica de interpretação fica associada à programação) quer na sua evolução futura.

O Editor é uma aplicação WYSIWYG[3] onde o utilizador tem sempre uma noção do resultado final do seu trabalho. Foi desenvolvido um conjunto de objetos para representar imagens, vídeo, áudio, texto e data e hora. Para criar os conteúdos, o utilizador só tem de acrescentar um objeto ao projeto, de forma visual e interativa. Foi desenvolvida uma barra de propriedades para exibir as propriedades dos *plugins*.

O Visualizador é utilizado para executar os conteúdos criados, está integrado no Editor e foi desenvolvido de forma a poder ser integrado noutras aplicações.

Para tornar possível a inserção dos recursos numa área de projeto remota e limitar o projeto a recursos só existentes nessa área, foi desenvolvido um Serviço Web que também torna possível a publicação dos projetos para o *BackOffice* da FutureboxTv. Este serviço é ainda responsável por gerir a área de trabalho e a área de publicação.

1.0.5 Organização da dissertação

Esta dissertação está organizada da seguinte forma:

Capítulo 1 Introdução, faz uma primeira abordagem ao trabalho referenciando a motivação, os objetivos, a abordagem proposta e as principais contribuições.

Capítulo 2 Conceitos, Tecnologias e Ferramentas, expõe vários conceitos, tecnologias e ferramentas, ligados diretamente com a execução do trabalho.

Capítulo 3 Trabalho Relacionado, referenciados alguns editores bastante conhecidos, concorrentes diretos deste trabalho, e os aspetos a ter em conta quando se pretende desenvolver um editor.

[3]Do Inglês, *What You See Is What You Get*

Capítulo 4 Editor de *Layout* para Clientes Multimédia, faz a descrição do trabalho realizado, detalhando os vários componentes desenvolvidos.

Capítulo 5 Testes de usabilidade, faz a descrição dos testes de usabilidade necessários para avaliar a aplicação.

Capítulo 6 Conclusões e trabalho futuro, apresenta as conclusões retiradas face ao desenvolvimento deste projeto, assim como ideias que podem ser implementadas no futuro, para a sua evolução.

Capítulo 2

Conceitos, Tecnologias e Ferramentas

Este capítulo expõe e clarifica os diversos conceitos, ferramentas e tecnologias ligados diretamente com a execução do trabalho.

2.1 Conceitos

A apresentação dos conceitos relevantes do projeto permite uma melhor compreensão do problema. De seguida apresentam-se os conceitos de *Digital Signage*, Painel de controlo (*Dashboard*), WYSIWYG, Web Services, Aplicações de Internet Ricas, Templates e *Data Binding*

2.1.1 *Digital Signage*

Digital signage é um painel de informação digital que permite promover ou fazer publicidade de uma forma dinâmica e apelativa a um produto ou serviço, pode ser implementada em vários espaços e com diversos conteúdos para atingir tanto o público-alvo interno como o externo. Pode ser aplicado em várias áreas como por exemplo a área da saúde, a área da educação, a área do entretenimento, a área da banca, a área das empresas, a área da restauração, a área das vendas e a área do transporte.

Quando usado com o objetivo de comunicação externa, permite fazer publicidade à empresa, passar informação de produtos, serviços e promoções, para expandir as vendas dos mesmos. Permite ter um ecrã para a gestão de filas de espera e ao mesmo tempo passa

informação sobre a empresa, proporcionando um certo entretenimento para o público-alvo e aumentando a qualidade do atendimento da empresa.

Quando usado com um objetivo de comunicação interna permite reforçar a cultura empresarial, difundir as comunicações internas como notícias, sucessos, estratégias e lançamento de novos produtos ou serviços.

Os seus conteúdos podem ser transmitidos vinte e quatro horas por dia, sete dias por semana. Os ecrãs usados para passar a informação podem ser colocados de forma a otimizar um espaço que de outra forma teria pouca utilidade.

2.1.2 Painel de controlo (*Dashboard*)

O painel de controlo[1] é um painel onde existem vários indicadores. Por exemplo, no caso de um automóvel, o painel de bordo tem vários indicadores sobre o estado da viatura (velocidade, temperatura do motor, rotações do motor) que, no seu conjunto, fornecem ao condutor informação muito detalhada da viatura para que a condução seja segura.

O conceito de painel de controlo pode ser usado com vários fins na área da informática:

Sistemas de monotorização A supervisão de um sistema muitas vezes exige que sejam observados dados com origem em diferentes fontes. Na maioria dos casos, esta informação está dispersa por vários locais e a soma da sua utilidade dispersa é inferior à utilidade ganha ao ser agregada num único local. Os painéis de controlo permitem agregar diversas informações num único local e se necessário alterar a forma como são representadas para facilitar a análise dos dados. A forma mais usual de representar a informação em sistemas de monotorização é através de gráficos;

Negócios Normalmente o painel de controlo contém um conjunto de informações atualizadas em tempo real o que permite a quem as observa e analisa retirar conhecimento crucial do estado do seu negócio;

Software Uma aplicação painel de controlo é responsável por mostrar e gerir várias mini aplicações que normalmente se chamam *widgets*. O local e a forma como cada *widget* é mostrado depende das propriedades do painel de controlo, mas muitas vezes pretende-se que essas propriedades possam ser customizadas pelo utilizador.

Um painel de controlo permite adicionar vários elementos diferentes, contém as regras de como os elementos interagem entre si, de como são agrupados e de como são dispostos. Como cada conteúdo do painel de controlo é um elemento individual, os elementos podem ser desenvolvidos e testados separados uns dos outros e do painel de controlo.

[1] Do Inglês, *dashboard*

2.1.3 WYSIWYG

WYSIWYG[2] pode ser traduzido como "o que se observa é o que se vai obter". O termo é usado em informática para descrever um sistema que permite editar os conteúdos onde o resultado final é muito semelhante ao aspeto aquando da edição.

Utilizar as técnicas WYSIWYG é uma ótima forma de se poder desenvolver algo cujo aspeto tenha uma grande importância, pois estas permitem que o utilizador crie e esteja a visualizar o aspeto final em tempo real.

2.1.4 Web Services

Um *Web Service* é definido pelo W3C [3] como uma aplicação que permite interação entre máquinas através de uma rede. Normalmente os *Web Services* são aplicações que podem ser acedidas através de uma rede e executam serviços num sistema remoto.

Ao receber dados em formato XML, os *Web Services* permitem que as aplicações com quem interagem possam ser construídas com diferentes arquiteturas.

2.1.5 Aplicações de Internet Ricas

As aplicações de internet ricas, RIA[3] são aplicações Web com funcionalidades e características de aplicações desenvolvidas para PCs. Por norma, estas aplicações usam os *browsers* para efetuar a maior parte do processamento, mas os dados são mantidos no servidor.

Umas das características das RIA é que se tornam uma camada intermédia entre o utilizador e o servidor. Normalmente, estas aplicações são carregadas no início da sessão, sendo depois possível o carregamento de partes adicionais. A camada carregada no início da sessão é considerada o motor da aplicação e é responsável por comunicar com o servidor e mostrar os dados corretos aos utilizadores.

A criação de uma camada entre o cliente e o servidor permite:

- Agarrar e arrastar elementos, tornando a interface mais reativa às ações do utilizador;
- Transmitir ao utilizador uma sensação de estar a usar uma aplicação *desktop*;
- Que uma grande parte do processamento dos dados seja efetuada do lado do cliente libertando, dessa forma, o servidor.

A possibilidade de existir uma comunicação assíncrona entre a aplicação e o servidor também é uma grande vantagem destes sistemas, pois permite ao cliente continuar a

[2] Do Inglês, *What You See Is What You Get*
[3] Do Inglês, *Rich Internet Application*

interagir com a aplicação sem ter de aguardar a resposta do servidor. Por outro lado, torna possível ao cliente prever algumas das necessidades do utilizador e requisitar esses dados de uma forma antecipada.

2.1.6 Templates

Os *templates* são documentos que permitem separar o conteúdo da forma como este é apresentado. Normalmente, um *template* contém um conjunto de regras e posições que definem o aspeto de uma aplicação. Quando se realiza uma alteração no *template*, consegue-se alterar o aspeto da aplicação sem alterar a parte da lógica da programação.

Outra utilização para *templates* é a possibilidade de criar um aspeto padrão para componentes utilizados em várias aplicações. Caso seja necessário alterar o aspeto, basta alterar o *template* respetivo, sendo esta alteração visível em todas as aplicações.

2.1.7 *Data Binding*

O *Data Binding* [17] (Ligação de dados) é uma das formas de se conseguir criar um fluxo de dados entre um modelo de dados e uma interface, é a criação de uma ligação entre ambos. Após estabelecer uma ligação entre o modelo de dados e a interface, sempre que os dados são alterados, a interface reflete essas alterações.

O *Data Binding* permite que as alterações ocorram nos dois sentidos, isto é, se o modelo de dados for alterado, a interface capta essa alteração e atualiza-se. Por outro lado, quando se verifica uma alteração na informação na interface, esta pode atualizar a informação contida no modelo de dados.

Quando vários componentes estão ligados ao mesmo modelo de dados, o uso de *Data Binding* permite que, quando se altera um valor nos dados do modelo, esta alteração seja automaticamente refletida em todos os componentes ligados ao modelo. A Figura 2.1 mostra os conceitos do *Data Binding*.

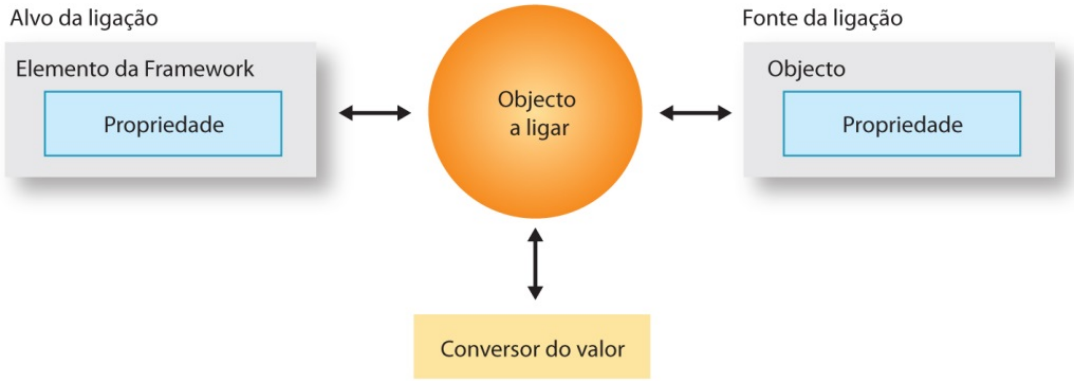

Figura 2.1: *Data Binding*.

2.2 Tecnologias

Para a escolha da tecnologia principal a utilizar, foi preciso considerar que o trabalho deve ser executado numa vertente *web*, e que necessita de utilizar uma tecnologia capaz de modelar elementos gráficos e animações. As tecnologias que cumpriam estes requisitos eram o HTML5 [34] , o Flash [1] e o Silverlight [19]. Foi escolhido utilizar o Silverlight pois o trabalho teria de ser integrado numa plataforma já desenvolvida nesta tecnologia, facilitando a integração do mesmo.

Esta secção descreve as tecnologias utilizadas no desenvolvimento do trabalho, nomeadamente: .NET *Framework*, Silverlight, XML, XML Schema, XAML e C#;

2.2.1 .NET *Framework*

A *Framework* .NET [20] é uma plataforma da Microsoft que permite executar, criar e desenvolver aplicações e serviços. Surgiu no início de 2002 e tem uma arquitetura semelhante à plataforma Java [13], permitindo que as aplicações criadas possam ser executadas em qualquer dispositivo com a plataforma .NET instalada.

A plataforma .NET, ao contrário da plataforma JAVA, suporta mais de 20 linguagens de programação distintas, tais como Haskell, Java, Java Script, LUA, Pascal, C#, C++, Fortran, Ruby, SmallTalk, entre outras.

A aplicação é compilada em duas fases. A primeira compilação é efetuada sobre a linguagem em que a aplicação é desenvolvida, para gerar uma linguagem intermédia comum à plataforma. A segunda compilação é efetuada sobre a linguagem intermédia, para gerar código máquina. Na Figura 2.2 podemos ver um esquema da compilação da plataforma .NET.

A plataforma .NET contém uma grande biblioteca de classes e utilitários, disponibilizando aos seus utilizadores um elevado número de funções, componentes e objetos.

2.2.2 Silverlight

O Silverlight [19] é uma ferramenta para criar e executar aplicações de internet ricas. A primeira versão do Silverlight foi lançada em 2007 e a mais recente, a versão 5, foi lançada a 9 de Dezembro de 2011. Encontra-se disponível para quase todos os *browsers* em forma de *plugin*. O Silverlight usa um subconjunto da Framework .NET.

O Silverlight integra multimédia, gráficos, animações e interatividade num único ambiente. A interface é criada usando uma linguagem declarativa baseada em XML ou XAML. O código da interface gráfica encontra-se separado da lógica de programação, tornando mais fácil a interação entre o *designer* e o programador. Isto também torna possível a alteração de uma parte sem a necessidade de alterar a outra.

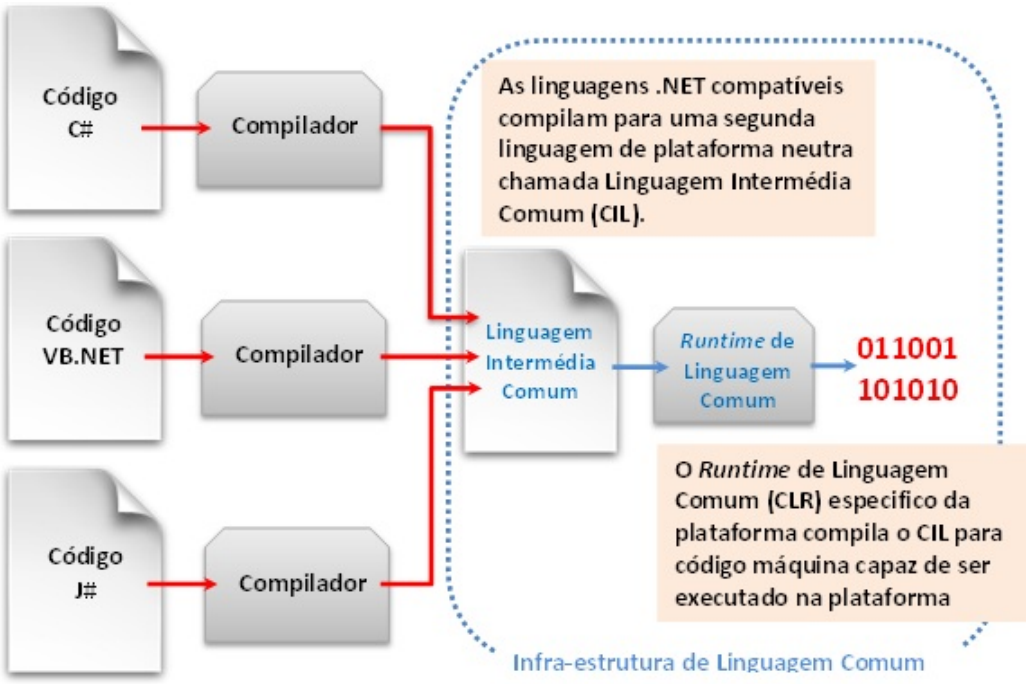

Figura 2.2: Esquema da compilação da plataforma .NET.

Para além da criação de RIA, o Silverlight permite o desenvolvimento de aplicações executadas fora do *browser*.

2.2.3 XML

O XML[4] [4] é uma linguagem de marcação recomendada pelo W3C[5] [3] desde 1998. Foi desenvolvida para facilitar a partilha de informação através da internet de uma forma simples e generalista.

Os documentos criados em XML podem conter uma linguagem ou uma estrutura de dados personalizada, o que permite que quem desenha a estrutura de dados do documento a possa otimizar para melhor resolver o seu problema.

2.2.4 XML Schema

É importante criar um conjunto de regras que um documento XML deva cumprir para validar a sua estrutura. Para tal, é usado um documento de XML Schema.

O XML Schema [5] é um sucessor do DTD[6] que não tem limitadores sobre o tipo de dados suportados. O XML Schema foi proposto pelo W3C como alternativa ao DTD,

[4]Do Inglês, *Extensible Markup Language*
[5]Do Inglês, *World Wide Web Consortium (W3C)*
[6]Do Inglês, *Document Type Definition*

apresentando melhorias a nível da especificação da estrutura, do tipo de dados suportados e permite uma descrição muito mais rica dos documentos XML e a validação da sua estrutura de dados.

2.2.5 XAML

XAML[7] [23] é uma linguagem declarativa baseada em XML. Foi criada pela Microsoft para definir as interfaces, fornecendo uma forma de separar a lógica da programação da interface.

Como o XAML é baseado em XML, a complexidade das ferramentas de processamento é reduzida, permitindo que o mesmo ficheiro possa ser editado por múltiplos utilizadores e até múltiplas aplicações.

2.2.6 C#

O C# ou C Sharp [21] é uma linguagem de programação orientada a objetos, fortemente tipada e desenvolvida pela Microsoft. O C# foi criado como parte da *Framework* .NET sem ter a preocupação de compatibilidade com código já existente. O C# foi baseado nas linguagens C++ e Java mas também contém influências de outras linguagens de programação.

2.3 Ferramentas

Para o desenvolvimento deste trabalho foram utilizadas as seguintes ferramentas: FutureboxTv, Expression Blend e o Visual Studio 2010.

2.3.1 FutureboxTv

A FutureboxTv [9] é uma plataforma especializada para a distribuição de conteúdos de vídeo e animação de uma forma fluida, transmitindo uma maior dinâmica quando comparada com conteúdos de natureza mais estática como texto ou imagem. Permite ainda a obtenção de dados a partir de fontes externas tais como RSS e *Web Services*.

A plataforma fornece aos clientes um sistema de *backoffice* com controlo de acessos e utilizadores. A plataforma pode ser adaptada às especificações únicas de cada cliente e ainda engloba um conjunto de serviços que possibilitam os processos de gestão de vídeo e se necessário os processos de codificação e mudança de formato. A plataforma pode ser utilizada como: *Digital signage*, associada a pontos geográficos específicos, permite a transmissão de múltiplas emissões (potencialmente diferentes) e com controlo total do

[7] Do Inglês, *Extensible Application Markup Language*

conteúdo apresentado. Tal permite a integração de experiências mais ricas que valorizam os espaços físicos em que se encontram integrados; Internet/ *WEB TV*, pode ser acedida de qualquer lugar, utiliza um *browser web* para executar a aplicação, tem um ambiente interativo onde o utilizador escolhe o que que quer ver e permite múltiplas categorias de informação.

A FutureboxTv contém um serviço para geração automática de conteúdos. Este serviço permite a criação autónoma de novos conteúdos de vídeo a partir de recursos externos, a partir de *templates* previamente criados que detalham o modo como os recursos externos devem ser articulados e animados para proceder à geração de vídeo.

2.3.2 Expression Blend

O Expression Blend [18] é uma ferramenta de desenho da Microsoft para interfaces de aplicações. O desenho é feito através do arrastamento de formas e desenho de controlos com a possibilidade de adição de interatividade dos seus componentes. O código gerado é XAML e o utilizador pode visualizar e editar o seu programa diretamente no código ou através da interface do Expression Blend.

O uso do Expression Blend permite que os *designers* estejam a trabalhar na interface enquanto os programadores estão a trabalhar na lógica de programação, usando outra ferramenta, como por exemplo o Visual Studio.

2.3.3 Visual Studio 2010

O Visual Studio [22] é um ambiente de desenvolvimento integrado (ADI)[8] da Microsoft, bastante completo, que permite usar várias linguagens de programação para desenvolver aplicações para diversas plataformas. Foi lançado a 12 de Abril de 2010 em conjunto com a versão 4 *Framework* .NET.

Permite desenvolver desde aplicações para a linha de comando até às mais complexas e interativas interfaces gráficas. Também possibilita a criação de aplicações Web, páginas Web e Web *Services*. Contém suporte nativo para um grande número de linguagens de programação tais como C#, C, C++, F#, Visual Basic, M, Python, Ruby, XML, XSLT, HTML, XHTML, Java Script e CSS. É possível adicionar novas linguagens de programação através da instalação de módulos extras.

O editor de código realça a sintaxe e permite auto completar o código, dando dicas de variáveis, funções e métodos, construtores da linguagem e consultas. Também torna possível colapsar blocos de código e inserir marcadores.

O *debugger* é uma ferramenta indispensável em qualquer *IDE*. O Visual Studio contém um *debugger* que funciona não só ao nível do código fonte mas também ao nível do código

[8] Do Inglês, *Integrated development environment (IDE)*

máquina e permite inserir *breakpoints* para se poder analisar certas partes específicas do código.

Tem um modo de desenho para ajudar o desenvolvimento das interfaces gráficas, disponibiliza uma barra de ferramentas com todos os elementos por omissão e permite ao utilizador adicioná-los com um simples arrastar para a posição desejada. Depois dos elementos estarem criados, ainda fornece ferramentas para ajudar o seu posicionamento.

Capítulo 3

Trabalho Relacionado

Neste capítulo são referenciados os aspetos a ter em conta quando se pretende desenvolver um editor, alguns editores bastante conhecidos e concorrentes diretos deste trabalho.

3.1 Aspetos a ter em conta no desenho de um editor

Quando se desenha uma aplicação, deve ter-se em conta diversos aspetos gráficos para tornar a sua interação com o utilizador simples e fluida. Outras considerações importantes são as formas de interação e as funcionalidades a disponibilizar.

Esta secção apresenta e discute: barra de objetos & tela branca, editor de definições, parte principal da aplicação, agrupar e esconder grupos de conteúdo, painéis colapsáveis, anular ações e interação com os objetos.

3.1.1 Barra de objetos & tela branca

Quando se coloca uma tela branca ao lado de uma barra de objetos (Figura 3.1), o utilizador carrega nos botões da barra de objetos para adicionar objetos à tela branca [32]. Este é o modelo mais comum para desenvolver um editor gráfico que envolva criar novos objetos e arranja-los num espaço virtual.

O conjunto barra de objetos e tela branca é tão comum que quase todos os utilizadores de programas de *desktop* já utilizaram um programa que os utilize. A palete costuma conter ícones comuns (mão, lupa, seta, entre outros) que são reutilizados por diversas aplicações, conservando o seu significado.

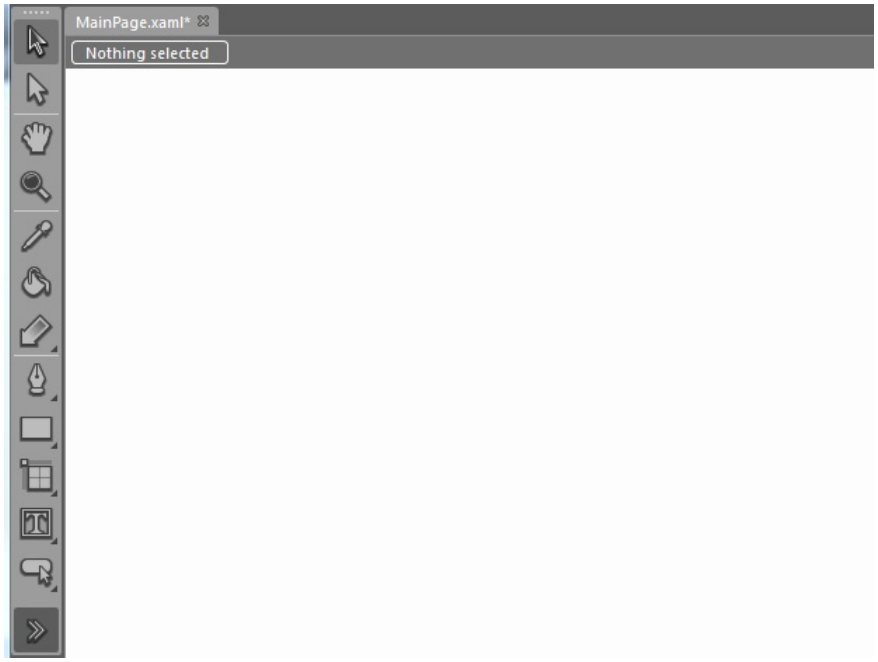

Figura 3.1: Barra de ferramentas e tela do Expression Blend 4.

A tela pode encontrar-se na mesma janela da palete de objetos ou em janelas diferentes, pode estar sozinha ou em conjunto com outras paletes de objetos. Independentemente da maneira como são apresentadas, o importante é estarem lado a lado para que o utilizador perceba que pode arrastar objetos da palete para a janela.

A palete de objetos em si deve ser uma grelha com ícones, podendo conter texto, no caso de os ícones não serem muito expressivos. Algumas aplicações mostram, para além dos ícones, os nomes dos objetos; contendo texto ou não, o mais importante é conter sempre um ícone para permitir ao utilizador reconhecer a palete de objetos pelo que ela é.

3.1.2 Editor de definições

O editor de definições fornece aos utilizadores uma janela onde podem encontrar e alterar as definições e propriedades da aplicação (Figura 3.2). No caso de existir um número elevado de definições, o conteúdo deve ser divido em várias páginas. Os utilizadores podem aceder ao editor não com a intenção de alterar as definições, mas para observar as várias definições existentes.

As propriedades devem ser agrupadas corretamente nas várias categorias. A etiqueta que identifica as propriedades deve permitir ao utilizador perceber de que propriedade se trata de uma forma imediata. O utilizador deve conseguir perceber qual o valor que a propriedade contém, de uma forma quase imediata, ao passar os olhos por ela.

Quase todas as plataformas contêm um local padrão onde se podem encontrar as pro-

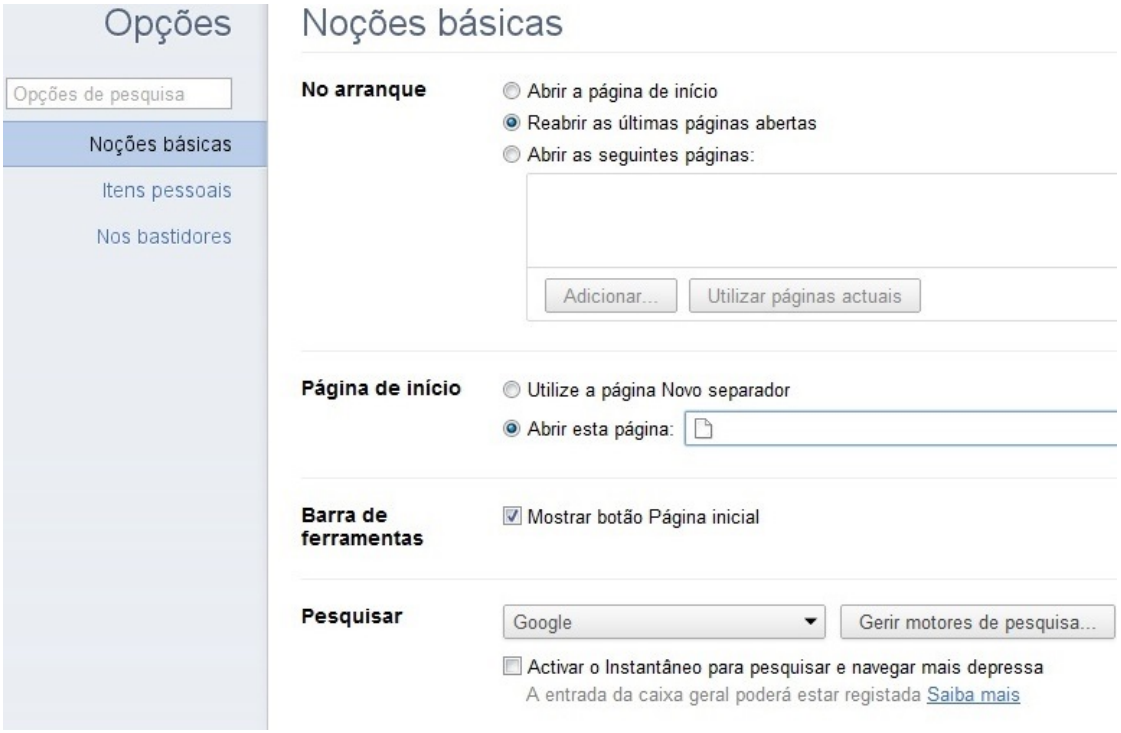

Figura 3.2: Menu opções do Chrome

priedades das aplicações. É conveniente seguir esse padrão pois já é familiar para os utilizadores. As propriedades mais comuns devem poder ser acedidas sem que o utilizador tenha de andar a navegar por múltiplas partes.

Outro ponto importante a considerar é se as propriedades são automaticamente aplicadas ou se deve existir um botão para salvar e outro para cancelar. A escolha entre estas opções depende do tipo das propriedades.

3.1.3 Parte principal da aplicação

A parte mais importante da aplicação deve ser a secção mais larga na página, à volta da qual devem estar os outros conteúdos em painéis mais pequenos. Por exemplo, num editor de objetos, a parte principal é a tela onde os objetos estão dispostos (Figura 3.3).

O desenho da interface deve conseguir guiar os olhos do utilizador imediatamente para a parte mais importante da aplicação. Assim que o utilizador identifica esta parte, ele começa a ver as outras partes e a identificar como estas estão relacionadas com a parte principal.

Ao desenhar a parte principal, deve-se ter em atenção os seguintes pontos:

Tamanho Deve ter o dobro da largura [32] das outras partes que estejam ao seu lado e

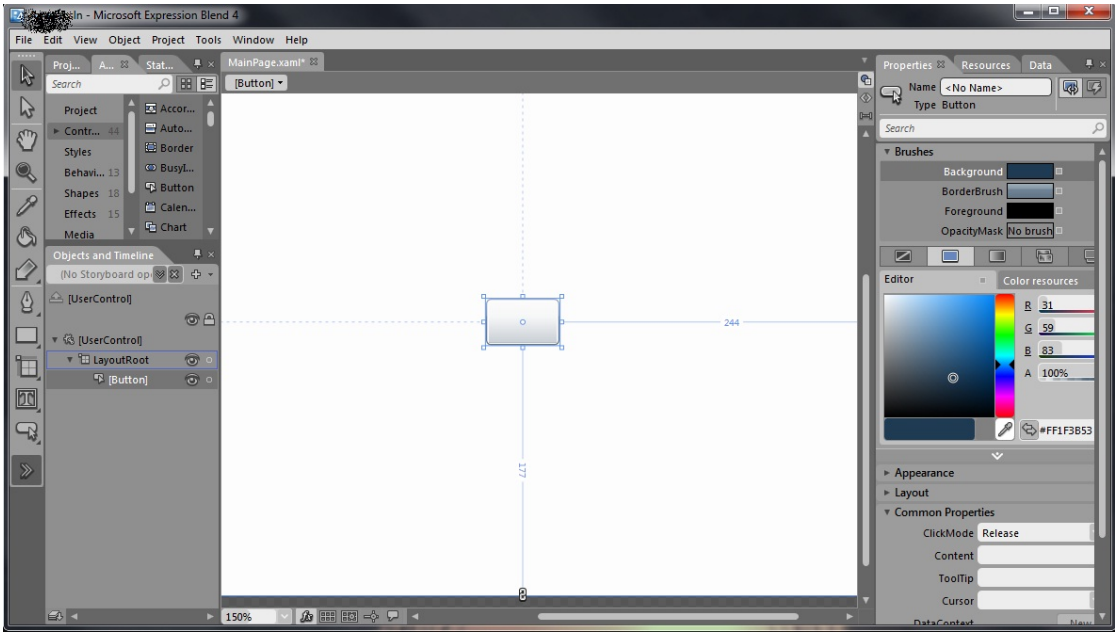

Figura 3.3: Expression Blend 4

pelo menos o dobro da altura das partes que estejam em cima ou em baixo. Deve também ter uma área maior que as restantes partes;

Cor Deve usar cores que contrastem com os elementos nas margens [32];

Destaques Os destaques conseguem atrair o olhar do utilizador de um ponto para outro [32];

Contexto Quando o utilizador abre uma aplicação, espera encontrar algo e vai à sua procura. Deve-se colocar essa área como a parte principal da aplicação e facilitar o seu reconhecimento [32].

A posição da parte principal não é muito importante, e se for suficientemente grande, vai acabar perto do centro, de qualquer forma.

3.1.4 Agrupar e esconder grupos de conteúdo

Muitas vezes surge a necessidade de agrupar diferentes itens de conteúdo em módulos que possam ser fechados e abertos independentemente dos outros. Para conseguir esta funcionalidade, o acordeão é, normalmente, a melhor solução.

O acordeão (Figura 3.4) é um componente que consegue conter grandes quantidades de conteúdos que podem ser agrupados e não estão visíveis todos ao mesmo tempo. Também permite que o utilizador visualize certos grupos e esconda outros a seu gosto e que consiga

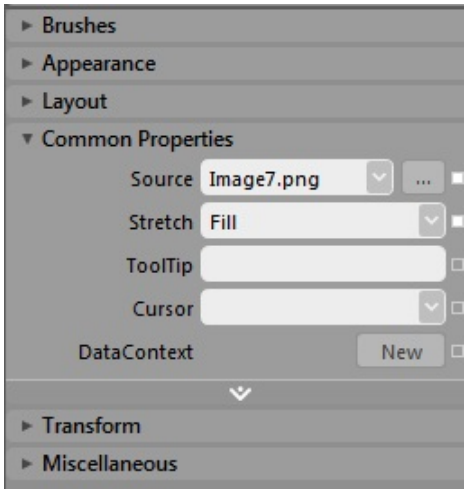

Figura 3.4: Painel das preferências do Microsoft Expression Blend 4

ter mais que um módulo aberto ao mesmo tempo. Permite igualmente agrupar vários módulos com alturas diferentes uns dos outros.

Os acordeões funcionam muito bem em paletes de ferramentas e painéis movíveis, pois permitem aos utilizadores abrirem qualquer número de módulos e deixarem-nos abertos, tendo um controlo total da sua área de trabalho.

3.1.5 Painéis colapsáveis

Uma forma de simplificar uma interface é a possibilidade de esconder partes críticas da mesma. Os painéis colapsáveis servem para colocar conteúdo opcional ou secundário e entregar ao utilizador a opção de os expor ou esconder. Na Figura 3.5 podemos observar o painel da barra de ferramentas aberto. Assim que se deixa de utilizar, ele colapsa.

Quando se contém uma grande quantidade de conteúdo heterogéneo para mostrar e não se tem espaço para tudo, ou se tem de dar prioridade visual ao conteúdo principal, utilizam-se painéis colapsáveis para colocar conteúdos que o utilizador quer esconder.

O conteúdo dos painéis colapsáveis deve poder ser acessível através de um único clique. O botão usado para representar o painel fechado deve conter o nome do painel e um símbolo para indicar que contém conteúdo escondido. Podem-se usar animações para abrir e fechar os painéis.

3.1.6 Anular ações

Permitir que os utilizadores possam desfazer as suas ações transmite a sensação de segurança em relação à interface. Os utilizadores, quando sabem que podem desfazer os seus passos, sentem-se mais à vontade para explorar a interface ou realizar várias ações, e

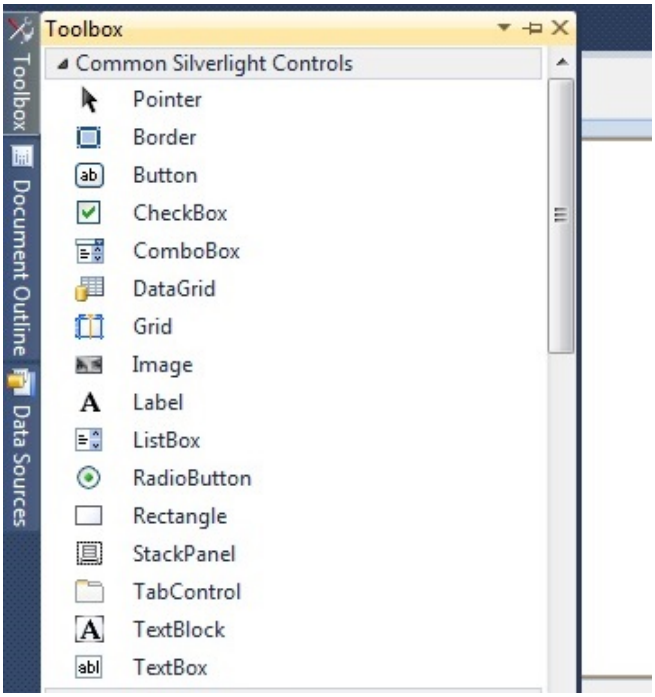

Figura 3.5: Visual Studio barra de ferramentas expandida.

sentem-se confiantes de que não estão a fazer alterações que não podem ser mudadas.

Normalmente, as ações que podem ser anuladas são guardadas numa pilha. Cada operação bem realizada é guardada no topo da pilha; quando o utilizador desfaz uma ação, ela está no topo da pilha, sendo depois removida. Se o utilizador quiser refazer a ação que anulou, esta é retomada para o topo da pilha.

3.1.7 Interação com os objetos

Existem diversas formas de interação com os objetos que são usuais em editores gráficos. As linhas de referência e o redimensionamento de objetos são introduzidos de seguida.

Linhas de referência

As linhas de referência [31] são usadas para ajudar os utilizadores a alinharem objetos. Podem ser verticais ou horizontais e permitem aos utilizadores posicionarem os objetos com precisão em relação às margens da página ou em relação a outros objetos. Na Figura 3.6 pode-se observar um botão alinhado de acordo com as linhas de referência.

Algumas das versões de linhas de referência permitem que, quando o utilizador move a linha de referência, os objetos que estão ancorados a ela se movam juntamente. Isto é bastante útil pois permite ao utilizador mover todos os objetos de uma só vez em vez de

Figura 3.6: Botão alinhado de acordo com linhas de referência.

os arrastar um a um.

As linhas de referência são bastante úteis quando se trata de manipular os objetos diretamente mas não deve substituir as ferramentas tradicionais de alinhamento.

As linhas de referência são linhas que flutuam sobre a área de desenho, não fazem parte do trabalho e não devem ser confundidas como se fizessem. Elas devem ser desenhadas sempre com um pixel de tamanho e esta medida não deve ser alterada mesmo com *zoom*. Normalmente, estas linhas são de cores brilhantes.

Redimensionar objetos

A propriedade de redimensionar os objetos diretamente [31] é uma forma de os utilizadores não terem de alterar o fluxo de trabalho. A utilização de formulários para efetuar a alteração de tamanho não é tão agradável como a manipulação direta e faz com que o utilizador seja obrigado a pensar em números em vez de pensar visualmente.

Uma das formas mais usuais para a funcionalidade de redimensionar os objetos é o arrastamento dos cantos do objeto para redimensionar a altura e a largura em simultâneo, ou por pontos intermédios aos cantos, permitindo só alterar uma propriedade: a altura ou a largura. Na Figura 3.7 temos um botão selecionado onde são destacados os pontos em que o utilizador pode redimensionar o objeto; estes são representados por pequenos quadrados nos cantos e entre os cantos.

Muitas vezes os utilizadores pretendem preservar o rácio do objeto. Esta funcionalidade pode ser realizada fornecendo vários modos de redimensionar os objetos. Um desses modos é o que mantém o rácio. Normalmente, este modo está ativo nos cantos dos objetos, enquanto que nos pontos entre estes só é permitido redimensionar uma das propriedades.

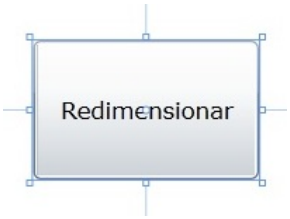

Figura 3.7: Botão selecionada

3.2 Editores

Os editores apresentados nesta secção não estão diretamente relacionados com o trabalho realizado, mas foram estudados durante o seu desenvolvimento, pois destacam-se no número de utilizadores que os utilizam e nas suas funcionalidades. De seguida, descreve-se a interface do Photoshop, a interface do Visual Studio 2010, a Interface do Expression Blend 4 e o GIMP

3.2.1 Interface do Photoshop

O Photoshop [25] é um programa de edição e manipulação de imagens. A interação entre o utilizador e o Photoshop é realizada através de cliques e arrastamento[1] dentro do programa. Na Figura 3.8 podemos observar os elementos chave do Photoshop e a sua localização por defeito [15]. Todas as funcionalidades do Photoshop podem ser encontradas numa única janela, que pode ser redimensionada e movida.

Pode-se dividir a janela do Photoshop nas seguintes secções:

Barra de aplicação(Application bar) Permite realizar funções de navegação e comandos da forma como a janela é vista. No Windows está localizada à direita da barra de menus;

Painel âncora(Docking pane) Contém vários grupos de painéis, empilhados uns em cima dos outros, e é possível adicionar ou remover painéis a este painel com um simples arrastar dos mesmos;

Barra de opções(Options bar) Mostra as várias opções de que cada ferramenta dispõe. Quando se altera a ferramenta que se está a usar, esta barra é automaticamente atualizada;

Barra de estado(Status bar) Contém informação do *zoom* e do tamanho do ficheiro;

Barra de título(Title bar) Exibe o nome da última gravação em disco da imagem. Quando se tem mais do que uma imagem, esta permite fazer a distinção entre elas e seleciona-las;

[1] Do Inglês, *dragging*

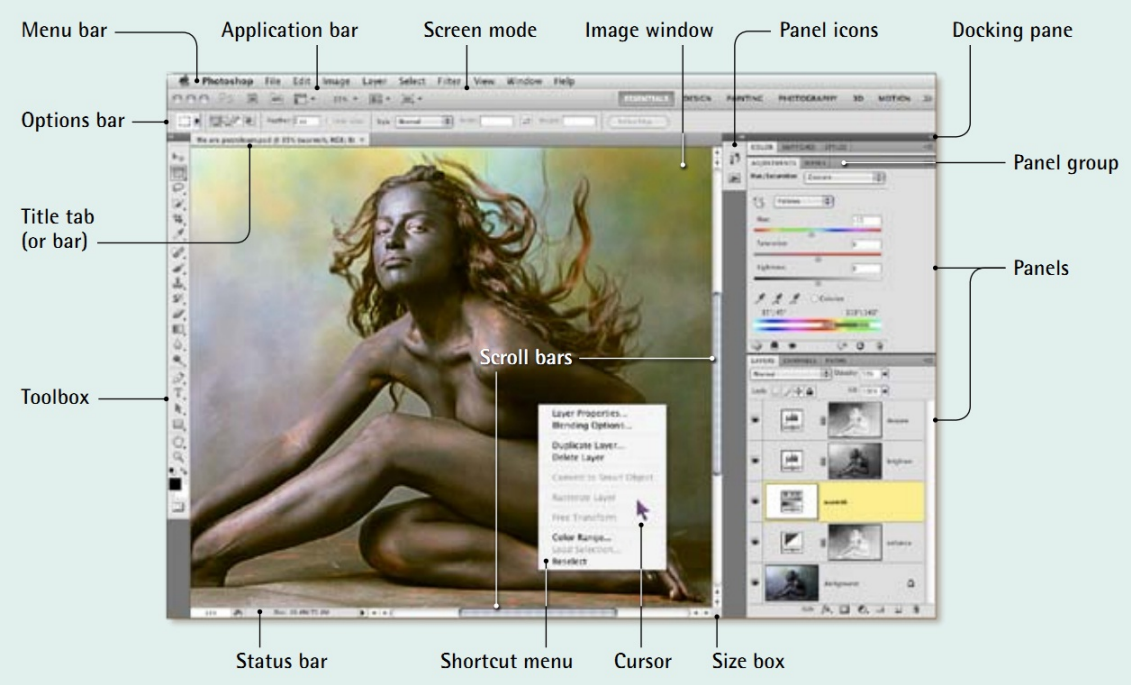

Figura 3.8: Janela do Photoshop

Caixa de ferramentas(Toolbox) Permite o acesso às ferramentas de edição do Photoshop. Ao clicar num ícone, seleciona-se a ferramenta que este representa;

Janela da imagem(Image window) Contém a imagem aberta. No Photoshop a janela das imagens é a zona onde se realiza o trabalho e é permitido ter múltiplas janelas abertas ao mesmo tempo;

Barra de menus(Menu bar) Contém a lista de comandos e opções.

O Photoshop permite que os utilizadores possam customizar a aplicação para melhor satisfazer as suas necessidades e aumentar o seu desempenho. Permite ainda guardar a forma como a sua área de trabalho se encontra, para mais tarde a poder retomar de uma forma fácil.

3.2.2 Interface do Visual Studio 2010

A gama de produtos Visual Studio 2010 partilha uma IDE[2] composta por vários elementos. Na Figura 3.9 podemos observar o Visual Studio 2010.

As várias partes que compõem a interface do Visual Studio 2010 são:

[2]Do Inglês, *Integrated Development Environment*

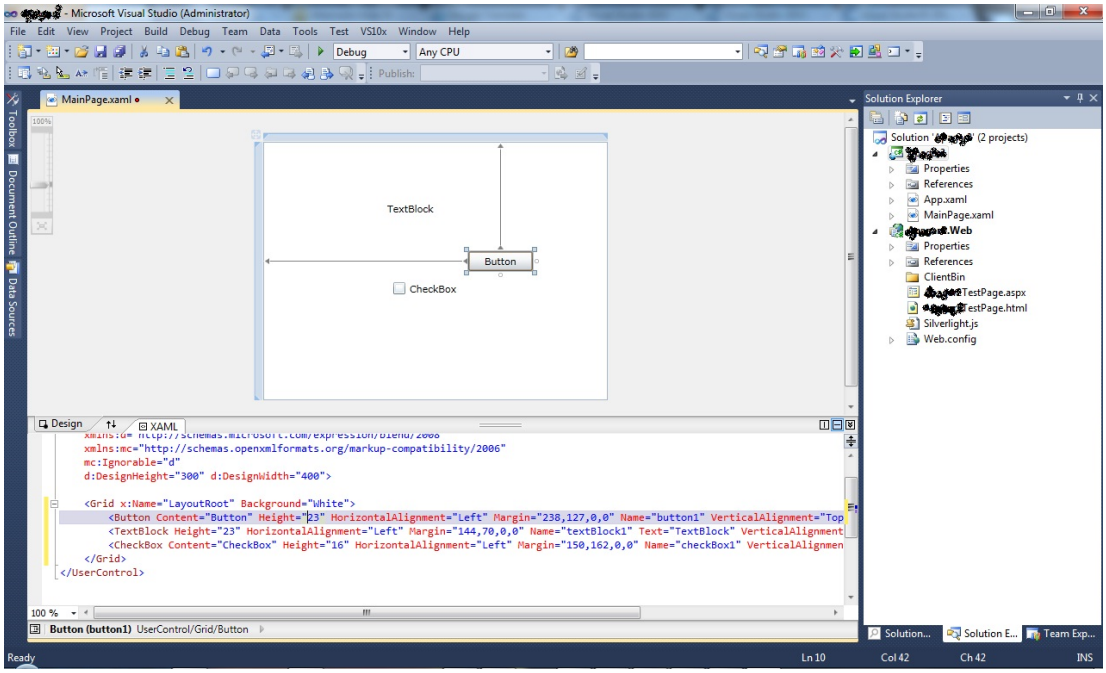

Figura 3.9: Visual Studio 2010 IDE

Menu Encontra-se no topo da aplicação e é semelhante à barra de menus padrão de diversas aplicações. Contém várias funções que podem ser executadas clicando nelas. É possível ver quais as teclas de atalho padrão para as várias funções;

Barra de ferramentas Encontra-se por baixo da barra de menus. Contém um subconjunto das funcionalidades existente nos menus, e que normalmente são mais utilizadas. As barras de ferramentas podem ser customizadas pelos seus utilizadores;

Área de trabalho Encontra-se no centro da aplicação e é onde se escreve o código ou se manipulam os objetos visuais;

Caixa de ferramentas Encontra-se no lado esquerdo, contém uma lista de controlos sensível ao contexto do trabalho. Os controlos podem ser agarrados e arrastados para a superfície de desenho;

Explorador da Solução [3] Encontra-se situada à direita da aplicação e é onde se pode visualizar todos os elementos da solução, dos projetos e dos elementos de cada projeto. É nesta zona que se podem encontrar e organizar todos os ficheiros e propriedades dos projetos;

Barra de estado Encontra-se no fundo da aplicação e transmite *feedback* ao utilizador sobre o que está a acontecer com o Visual Studio. A barra de estado também é sensível ao contexto, mostrando informação relevante para o trabalho que o utilizador está a efetuar.

[3] Do Inglês, *Solution Explorer*

O Visual Studio permite que o utilizador tenha vários documentos abertos e permite que este posicione as várias janelas em posições customizadas. As barras de ferramentas e os vários painéis que se encontram ao dispor do utilizador também podem ser customizados, tanto na posição como no tamanho. As propriedades dos elementos do IDE são definidas no início de acordo com as definições que o utilizador escolhe e existe a possibilidade de fazer um *reset* às definições para voltar às definições padrão.

3.2.3 Interface do Expression Blend 4

A interface do Expression Blend 4 é parecida com a interface dos outros produtos existentes no Expression Studio [24]. O Expression Blend fornece as ferramentas necessárias para se desenvolver e animar as interfaces gráficas em Silverlight ou em WPF. Na Figura 3.10 [14] pode-se observar a interface do Expression Blend.

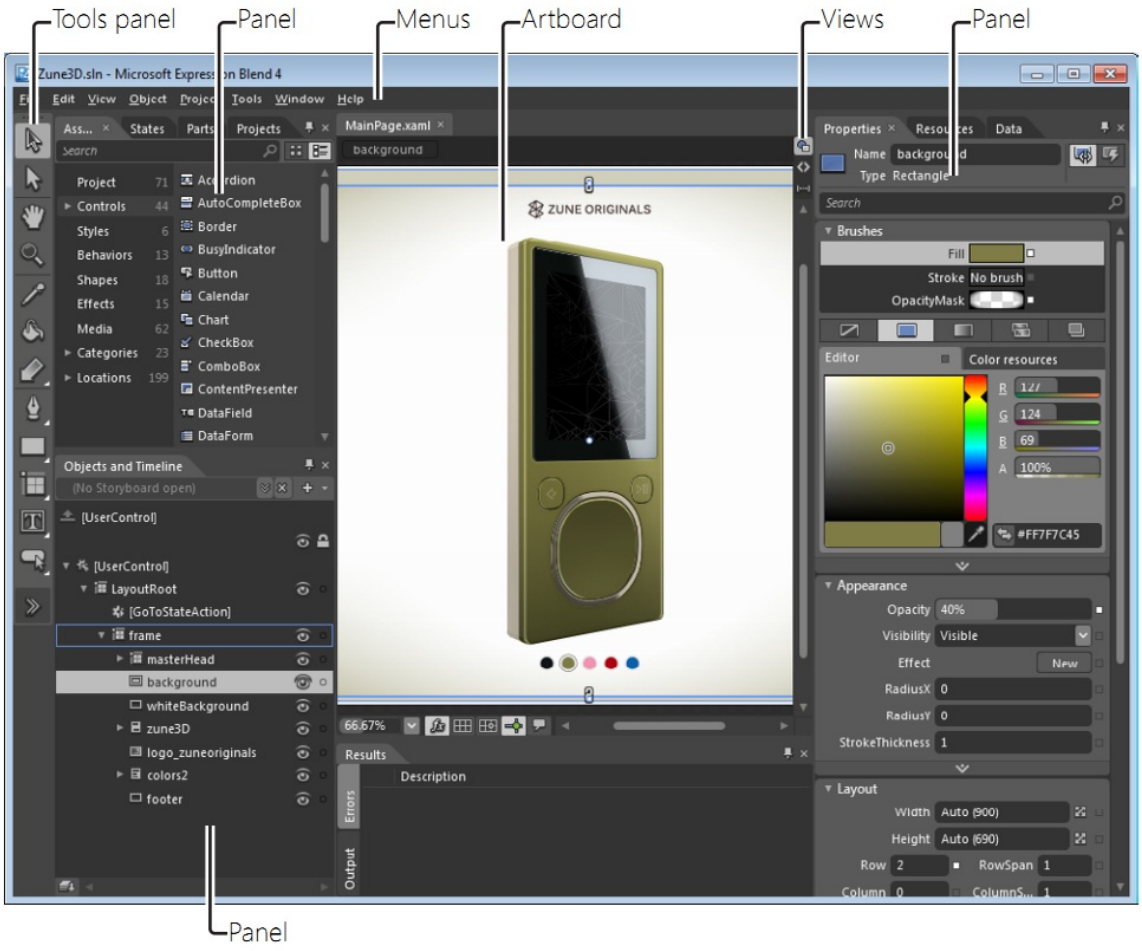

Figura 3.10: Expression Blend 4

Podem-se destacar os seguintes elementos da interface do Expression Blend:

Área de Desenho Área principal do Expression Blend, onde o utilizador constrói a sua aplicação desenhando e modificando os objetos. No topo desta área existe uma barra de separadores onde se pode ver e selecionar os vários ficheiros abertos;

Menus Área onde se encontram todas as funcionalidades do Expression Blend. É uma área sensível ao trabalho que o utilizador está a realizar, se o utilizador não conseguir executar uma ação, é porque essa ação não se encontra disponível para esse trabalho;

Painel do projeto Contém todos os ficheiros e projetos da solução, permitindo organizar e aceder aos vários ficheiros existentes;

Painel de objetos e linha do tempo [4] Contém todos os objetos existentes no documento que se encontra aberto e a sua hierarquia. Neste painel também é possível trancar um objeto, para impedir que as suas propriedades sejam alteradas, e esconder um objeto da área de trabalho;

Painel das propriedades Permite visualizar e alterar todas as propriedades contidas num objeto selecionado. Todas as propriedades estão organizadas por categorias que se podem contrair ou expandir;

Painel dos recursos Contém todos os recursos existentes no projeto;

Painel de ferramentas Encontra-se situado no lado esquerdo da aplicação e permite que o utilizador escolha uma ferramenta para desenhar ou selecionar um objeto na área de trabalho.

É importante que o utilizador possa ver o seu trabalho no modo de desenho ou no modo de código. O Expresssion Blend permite que o utilizador tenha a sua área de trabalho dividida com estas duas vistas ou possa optar por uma delas.

O Expression Blend também permite ao utilizador alterar a forma como os painéis estão arranjados. Contém duas vistas padrão: a vista de Desenho e a vista de Animação. Também é possível esconder os painéis para que a área de trabalho fique maior.

3.2.4 GIMP

O GIMP [10] (Figura 3.11) é um programa *open source* de manipulação e edição de imagens. Quando se abre a aplicação, todas as suas componentes aparecem em janelas desagregadas umas das outras. Este tipo de interface permite uma grande customização da área de trabalho, pois não está limitada somente ao reposicionamento de painéis dentro da janela da aplicação, tornando possível o posicionamento de painéis por toda a área de trabalho.

[4] Do Inglês, *Objects And Timeline*

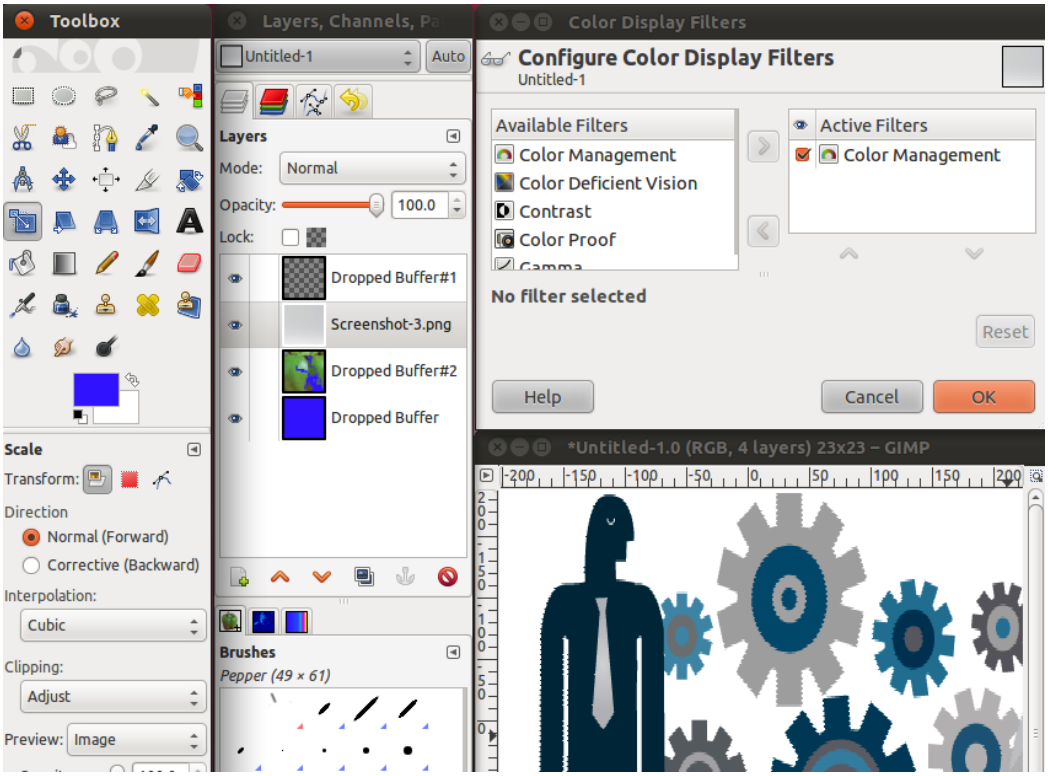

Figura 3.11: GIMP

3.3 Concorrentes diretos

Esta secção apresenta alguns produtos com funções, objetivos e funcionalidades muito semelhantes ao trabalho desenvolvido.

3.3.1 C-nario Messenger

O sistema C-nario Messenger [16] (Figura 3.12) é um *software* desenvolvido e distribuído por C-nario Ltd [2] que permite criar, agendar, gerir, transmitir e traduzir conteúdos de *digital signage*. É possível aumentar as funcionalidades do sistema com a adição de módulos extras, como por exemplo o módulo de relatórios, o que permite controlar a reprodução do conteúdo e, em certas condições gerar relatórios em Excel ou enviar um SMS ou um *email*. O *plugin* de criação de *scripts* permite usar dados de fontes externas.

O C-nario Messenger integra um programa de *design* gráfico que suporta um número elevado de efeitos e contém ferramentas para criar uma sequência de animações a partir de conteúdo estático, criando o efeito de imagens dinâmicas. Também é possível alterar dinamicamente a forma, o grau, o nível de transparência e a posição das janelas de conteúdo.

Suporta os seguintes tipos de conteúdos:

- Imagens (JPG, JPEG, BMP, PNG);
- Vídeo (AVI, MPG, WMV, H.264 (MPEG4));
- *Media streams* (ASF, ASX);
- Adobe Flash.

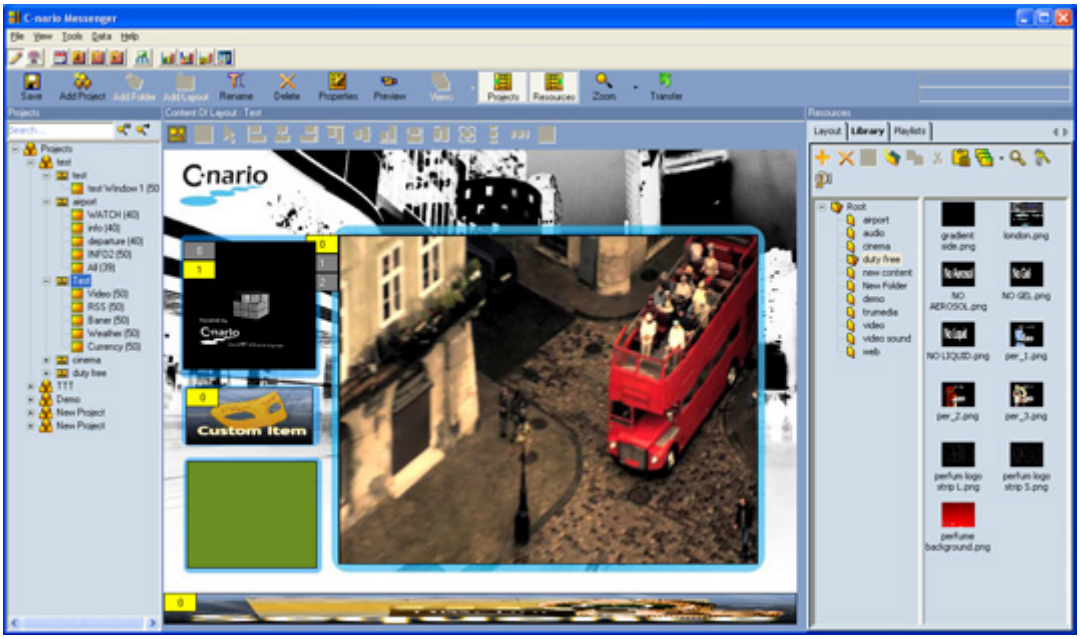

Figura 3.12: C-nario Messenger.

3.3.2 InfoCaster Creator e InfoCaster Player

O InfoCaster Creator [6] e o InfoCaster Player [26] são *softwares* desenvolvidos e distribuídos por Harrys [12].

O InfoCaster Creator (Figura 3.13) contém uma interface semelhante à interface PowerPoint [28]. Tem integrada uma ferramenta de desenho gráfico que permite sobrepor texto, gráficos, vídeos e adicionar todos os tipos de efeitos digitais e de vídeo incluindo complexas animações em 3D. Também permite utilizar uma grande variedade de efeitos tais como rotações em 3D ou deslocamento de texto.

O InfoCaster Creator suporta os seguintes conteúdos:

- Imagens (JPG, JPEG, BMP, PNG, GIF, TIFF, TGA, PCX);
- Vídeo (AVI, MPG, WMV, 3GPP, FLV, H.264 (MPEG4));
- *Media streams* (ASF, ASX, M3U);
- Adobe Flash.

O InfoCaster Creator suporta dados das seguintes fontes externas:

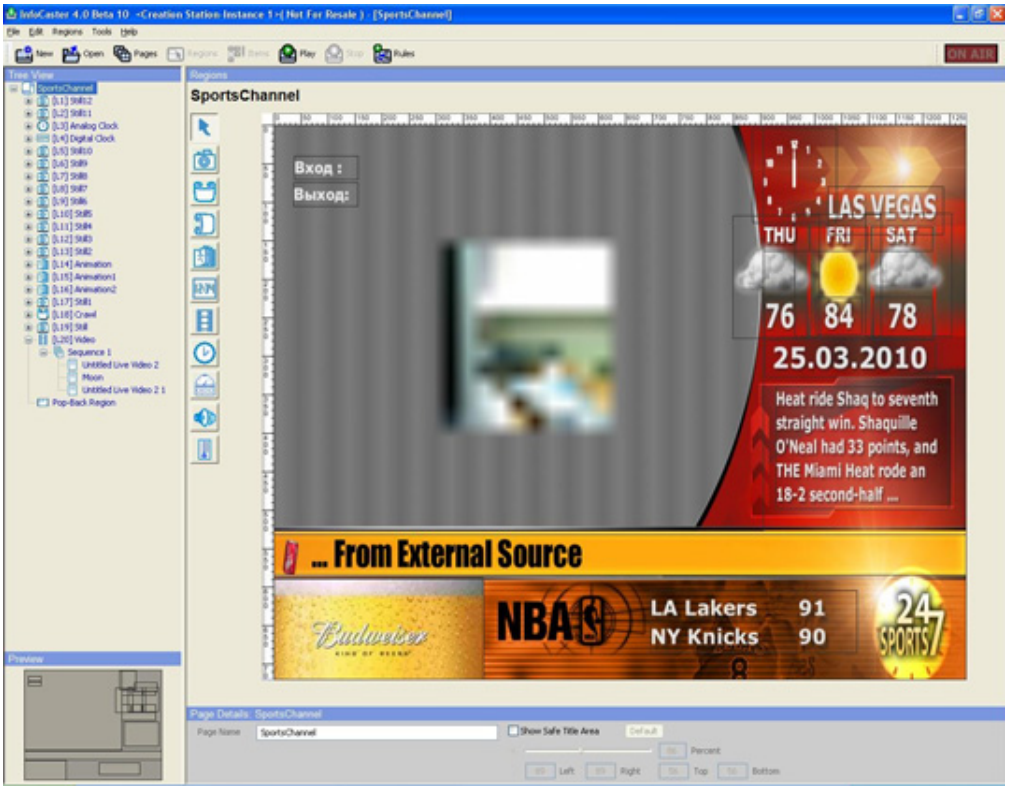

Figura 3.13: InfoCaster Creator.

- TXT / XML;
- RSS;
- ODBC;
- Vídeo e televisão, incluindo *streaming* de vídeo.

O InfoCaster Player é um *software* de reprodução de conteúdos. Permite a transmissão para vários monitores de conteúdo disperso em diferentes fontes. Este software encontra-se disponível em 3 versões que diferem entre si pela resolução máxima, o número de efeitos e formatos suportados.

3.3.3 Scala Designer e Scala Player

O Scala Designer [7] e o Scala Player [27] são *softwares* desenvolvidos e distribuídos por Scala [30]. O Scala Designer (Figura 3.14) é um software para criar conteúdos dinâmicos. Dispõe também de uma série de módulos opcionais que aumentam o leque de possibilidades de criação de modelos. O Scala Player é o *software* para a reprodução do conteúdo multimédia.

Scala Designer tem uma interface com um menu sensível ao contexto, o que facilita a sua utilização e faz com que não seja necessária uma habilidade especial para trabalhar com

Figura 3.14: Scala Designer.

ele. Suporta os seguintes conteúdos:

- Imagens (JPG, BMP, PNG, GIF, TIFF);
- Vídeo (AVI, MPG, WMV);
- *Media streams* (ASF, ASX, M3U).

O Scala Designer suporta dados das seguintes fontes externas:

- Ficheiros de texto;
- Bases de dados;
- Vídeo e emissão televisiva.

Capítulo 4

Editor de Layout para Clientes Multimédia

O desenvolvimento de uma aplicação envolve diversas etapas como o levantamento de requisitos para definir o que a aplicação deve ser e realizar, o desenho da arquitetura que especifica todas as componentes da aplicação e como interagem entre si. Este capítulo apresenta essas etapas e descreve as três aplicações desenvolvidas: o Serviço Web, o Editor e o Visualizador.

4.1 Levantamento de requisitos

O levantamento de requisitos é uma parte fundamental no desenvolvimento de uma aplicação, já que permite definir o que a aplicação deve ser ou deve realizar. Os requisitos definidos no âmbito deste trabalho estão divididos em dois grupos: requisitos não funcionais e requisitos funcionais. Um requisito não funcional é um requisito que especifica um critério que pode ser utilizado para avaliar uma aplicação e não um comportamento específico. Os requisitos funcionais definem funções, comportamentos e funcionalidades que o sistema deve cumprir.

4.1.1 Requisitos não funcionais

Os requisitos não funcionais que a aplicação deve satisfazer são:

Usabilidade A solução pode ser manipulada por utilizadores com poucos conhecimentos

de informática e que nunca tenham interagido com a aplicação. É necessário que os utilizadores consigam interagir com a aplicação de uma forma intuitiva e simples. As várias componentes da solução devem estar bem identificadas para que os utilizadores percebam do que se trata sem necessitarem de uma explicação muito detalhada ou formação. É ainda importante que o utilizador consiga efetuar as várias interações que o trabalho exige de uma forma fluida e dinâmica.

Extensibilidade A solução não pode ser estanque, deve permitir adicionar novas funcionalidades; deve ser extensível permitindo que a adição dos novos elementos com novas funcionalidades seja feita de uma forma simples.

Desempenho Uma aplicação que bloqueia ou que obriga o utilizador a esperar para efetuar uma ação que devia ser rápida causa desmotivação e prejudica a produtividade. É importante que a solução seja fluida e permita executar as ações de manipulação de forma direta sem bloqueios de performance.

Privacidade A solução necessita aceder a conteúdos armazenados numa localização remota. É importante que cada utilizador só consiga aceder aos seus conteúdos e não consiga visualizar os conteúdos dos outros utilizadores. O sistema deve implementar uma solução que restrinja o acessos dos utilizadores aos conteúdos alheios.

Tolerância a erro Uma aplicação que utiliza informação remota necessita de verificar se a informação que recebeu está completa e correta. A solução deve conter uma forma de verificar se um *download* foi efetuado corretamente e repeti-lo se este falhar, garantindo a sua disponibilidade.

4.1.2 Requisitos funcionais

Os requisitos funcionais da aplicação são os seguintes:

- Não apresentar informação técnica, como por exemplo mensagens de aviso, confirmação ou erro;
- Permitir que o utilizador possa guardar o seu trabalho e possa retorna-lo;
- Garantir a transparência do formato dos dados do projeto;
- Conter uma área de apresentação dos objetos que o utilizador pode utilizar;
- Possibilitar a alteração das definições base de um projeto (opções do projeto);
- Ter uma forma de visualizar, agregar e navegar nos elementos que estão associados à área de trabalho ativa;
- Conter uma área de trabalho privada. Esta deve ter dimensões que permitam ao utilizador realizar o seu trabalho com facilidade;

- Permitir que o utilizador possa interagir com os objetos através do rato para os mover ou redimensionar;
- Permitir a personalização dos objetos através de um conjunto de propriedades. É necessário que o utilizador possa ver estas propriedades e as possa alterar;
- Possibilitar ao utilizador pré-visualizar o trabalho realizado para identificar possíveis erros ou futuras melhorias;
- Ser acedida através de diversas localizações e através de dispositivos diferentes (Televisões, PCs);
- Conter um conjunto mínimo de *plugins* base, como o vídeo, a imagem, o som e o texto;
- Integrar da solução em ambiente FutureboxTv (WEB e *Digital Signage*) com conteúdos especializados;
- Integrar a solução em *BackOffice*.

4.2 Arquitetura da solução

Tendo em conta os requisitos funcionais e não funcionais descritos na secção anterior, foi desenhada a arquitetura da solução. Todos os componentes foram desenvolvidos isoladamente e posteriormente integrados no *BackOffice* da plataforma FutureboxTv, que já contém funcionalidades de controlo de utilizadores e de publicação de conteúdos.

A Figura 4.1 representa um esquema da arquitetura da aplicação. Contém todas as componentes desenvolvidas e as componentes externas com as quais existe alguma forma de interação e está dividida em quatro áreas:

- A área de Desenvolvimento de Projeto permite ao utilizador criar e alterar projetos;
- A área de Gestão de Recursos é responsável por criar, atualizar e publicar os projetos, receber, listar e armazenar os seus recursos;
- A área de Gestão de Conteúdos é responsável por administrar todos os conteúdos publicados e gerir em que locais estes vão ser executados;
- A área de Consumo de Conteúdos permite aos utilizadores finais visualizar os conteúdos.

Foram desenvolvidos 3 componentes:

1. Um Editor para criar projetos;
2. Um Visualizador para executar os projetos;
3. Um Serviço Web para armazenar os projetos e os seus recursos.

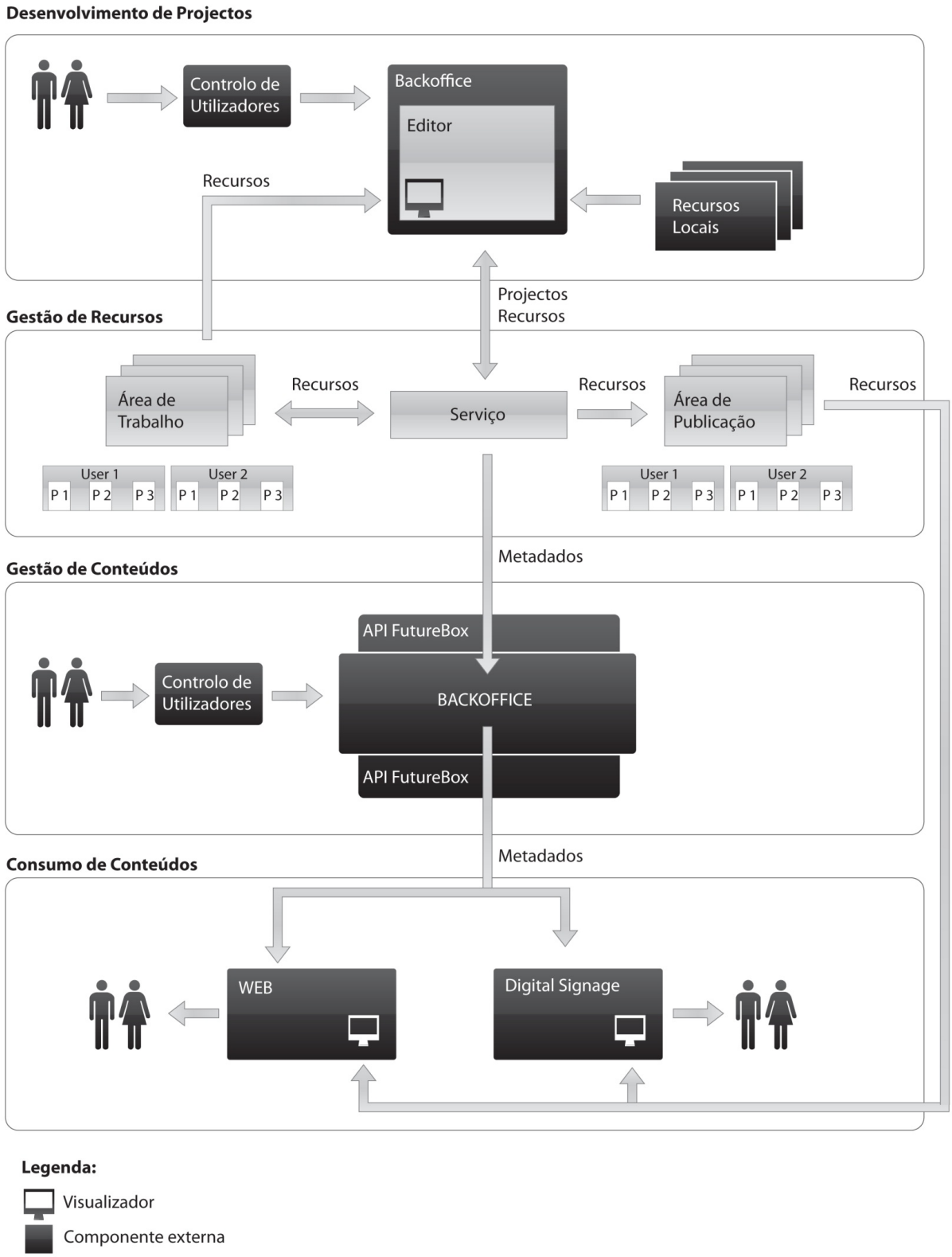

Figura 4.1: Esquema da arquitetura geral do trabalho

O Editor permite aos utilizadores criarem os seus projetos com conteúdos personalizados. É uma aplicação WYSIWYG[1] que permite ao utilizador ter sempre uma noção da aparência final do seu projeto, enquanto o constrói. Pertence à área de desenvolvimento de projetos.

O Visualizador é a componente que permite ao utilizador reproduzir os conteúdos dos seus projetos. Está integrado no Editor e nos clientes WEB e *Digital Signage*. Pertence à área de consumo de conteúdos.

O Serviço Web é a componente utilizada para gerir os projetos, os recursos dos projetos e para efetuar a transição de um projeto da área de trabalho para a área de publicação. Pertence à área de gestão de conteúdos. Descreve-se de seguida detalhadamente cada uma das áreas da arquitetura.

4.2.1 Área de Desenvolvimento

A área de Desenvolvimento de Projetos é utilizada pelos utilizadores que pretendam criar, editar ou publicar um projeto. Para que um utilizador interaja com o Editor, deve ter credenciais de "Criação de conteúdos". Se tiver estas credenciais pode realizar o *login* no *BackOffice* da FutureboxTv e aceder ao Editor. Cada utilizador só consegue aceder aos seus projetos. Para que não seja necessária a existência de um *login* no Editor, as credenciais do utilizador autenticado no *BackOffice* da FutureboxTv são transmitidas ao Editor. Este processo de autenticação é realizado uma vez aquando do acesso ao Editor.

No Editor está inserido o Visualizador que permite ao utilizador executar e visualizar o projeto que está a desenvolver. Para executar, o Visualizador necessita de ter conhecimento do *URL* do projeto para que, com essa informação, possa ir buscar os conteúdos diretamente à pasta do projeto.

O utilizador consegue inserir recursos locais no projeto através de um sistema de *upload* existente no Editor. Ao selecionar um recurso, o Editor submete-o à componente de Serviço Web, juntamente com a informação do projeto e do utilizador.

Sempre que o Editor necessita de saber quais os projetos que um utilizador contém ou quais os recursos existentes num determinado projeto, faz um pedido ao Serviço Web que retorna essa informação.

Para guardar a informação de um projeto ou criar um novo, o Editor envia a informação do projeto e do utilizador ao Serviço Web, que se encarrega de guardar a informação no local adequado.

Como o Editor tem conhecimento da localização do projeto, quando precisa de um recurso desse projeto, vai diretamente ao local onde o mesmo está contido.

[1] Definido na subsecção 2.1.3

4.2.2 Área de Gestão de Recursos

A área de Gestão de Recursos consiste num Serviço Web, numa área de Trabalho e numa área de Publicação. Nenhum utilizador tem acesso direto à área de Gestão de Recursos. A área de Trabalho e a área de Publicação são utilizadas para guardar os projetos e os seus recursos. O Serviço Web é utilizado para gerir os fluxos de dados, transmitir informação sobre os projetos e publicar os projetos.

É necessário ter uma área de Trabalho e uma área de Publicação pois a aplicação deve funcionar diretamente na WEB e, como tal, os recursos têm de estar armazenados remotamente.

A área de Trabalho contém todos os recursos e projetos que podem ser editados pelo Editor. É a esta área que o Editor e o Visualizador vão buscar os recursos dos projetos que estão a editar.

A área de Publicação contém todos os recursos e projetos publicados. É a esta área que os Visualizadores que executam os conteúdos publicados vão buscar os recursos.

A área de Trabalho e a área de Publicação têm um sistema de diretorias idêntico. Em cada uma existe um conjunto de diretorias, cada uma correspondente a um utilizador. Dentro da diretoria de cada utilizador existe uma diretoria correspondente a cada projeto. Este esquema não permite que os utilizadores partilhem recursos entre projetos mas, por outro lado, garante as seguintes características:

- Utilizadores diferentes podem conter projetos com o mesmo nome;
- Maior facilidade no controlo dos recursos a que cada utilizador tem acesso;
- Garantir que os recursos de projetos diferentes não colidam;
- O utilizador só pode ver os recursos do projeto em que está a trabalhar no momento;
- Os utilizadores só podem substituir os recursos do projeto em que estão a trabalhar;
- Para copiar um projeto basta copiar a sua diretoria.

4.2.3 Área de Gestão de Conteúdos

A área de gestão de Conteúdos permite aos utilizadores com credenciais de "Distribuição de Conteúdos" gerir os conteúdos publicados e efetuar a sua distribuição. O processo de gestão de conteúdos não foi desenvolvido no âmbito deste trabalho. Os conteúdos publicados são inseridos no *BackOffice* da FutureboxTv.

Para publicar um conteúdo, o Editor transmite ao Serviço Web qual o projeto que quer publicar e a informação de publicação; o Serviço Web transmite à API da FutureboxTv a informação do conteúdo a ser publicado; a API da FutureboxTv guarda a informação do conteúdo publicado.

Para gerir os conteúdos, o utilizador acede ao *BackOffice* da FutureboxTv que contém ferramentas de gestão de conteúdos e de clientes de conteúdos.

4.2.4 Área de Consumo de Conteúdos

A área de Consumo de Conteúdos disponibiliza os conteúdos aos consumidores finais. Para expor os conteúdos, a FutureboxTv contém dois clientes, WEB e *Digital Signage*, que interagem com a API da FutureboxTv para aceder à informação dos conteúdos que devem reproduzir. O Visualizador foi integrado nestes clientes.

O cliente WEB e o cliente de *Digital Signage* são responsáveis por reproduzir o conteúdo e passar ao Visualizador a informação do conteúdo que deve reproduzir. Tal permite a visualização de vídeos, imagens e animações (já existentes na FutureboxTv) e projetos desenvolvidos com a ferramenta implementada neste trabalho, de forma completamente transparente, ao utilizador final.

4.3 Linguagem declarativa de suporte à aplicação

Um dos requisitos do sistema é a possibilidade dos utilizadores poderem guardar o seu trabalho para que mais tarde o estenderem ou corrigirem. Para tal, é necessário ter uma forma fácil de armazenar e recuperar a informação do projeto.

Outra característica da solução é a existência de duas aplicações desagregadas que precisam de comunicar entre si. O Editor necessita de comunicar ao Visualizador a informação que o utilizador criou, para que este a possa interpretar e executar.

Como foi construído para uma vertente Web, o projeto com a informação do trabalho pode estar armazenado numa localização remota. Ao transmitir a informação de um local remoto, pode acontecer um variado número de falhas, que poderão comprometer a integridade da informação. Assim, é necessário que a estrutura de dados escolhida permita validar a correção da informação recebida.

Como a maioria da informação necessária para armazenar o trabalho realizado é comum ao Visualizador e ao Editor, decidiu-se criar só uma estrutura de dados comum que pudesse armazenar e reutilizar a informação necessária para os dois componentes. A estrutura deve ser extensível para permitir que se possam adicionar novos elementos que não foram considerados no âmbito desta solução, numa lógica de evolução futura.

Com estas restrições a decisão recaiu sobre o XML e XML Schema. Para guardar a informação do trabalho utiliza-se um documento em XML. Esta linguagem permite criar uma estrutura de dados evolutiva e facilmente adaptável que contém toda a informação necessária. Para validar o documento XML usa-se XML Schema, pois permite validar os tipos de dados e a estrutura/coerência da informação. Quando em execução, o XML é carregado e gravado.

O documento XML é composto por dois tipos de informação. O primeiro refere a informação do projeto e contém meta-informação como o nome dos autores, a versão e o nome do projeto. A Tabela 4.1 lista estes campos, bem como o tipo de dados e a sua função. O segundo contém a informação referente aos objetos e às suas propriedades. O Editor necessita de toda a informação, mas o Visualizador só necessita da informação referente aos objetos.

Campos	**Tipo**	**Propósito**
Nome do projeto	String	Guardar o nome do projeto
Descrição	String	Guardar a descrição do projeto
Largura do ecrã	Double	Guardar a largura do ecrã
Altura do ecrã	Double	Guardar a altura do ecrã

Tabela 4.1: Campos de informação base do projeto.

A informação relativa aos objetos é mantida numa lista. Cada objeto tem as suas propriedades e estas diferem consoante o tipo de objeto. A ordem dos objetos é importante, pois é utilizada para representar a ordem pela qual os objetos são inseridos, de modo a controlar as sobreposições dos mesmos. O último objeto é o que será posicionado à frente e os restantes ficam mais atrás. Inicialmente foi considerada a possibilidade de utilizar um campo para representar esta noção de profundidade e agrupar os vários objetos na mesma profundidade. Esta possibilidade foi abandonada, pois aumentava a complexidade na criação e leitura do XML, assim como do processo de criação dos objetos no Editor e no Visualizador.

A lista de objetos contém todos os objetos do projeto. O primeiro objeto é sempre um objeto do tipo *background* e só pode existir um objeto deste tipo. A seguir ao *background*, pode existir um número indefinido de objetos, cada um dos quais a representar um *plugin* no Editor. Quando se pretende criar um novo tipo de objeto, para que a linguagem o reconheça, basta adicioná-lo à lista de objetos com as respetivas propriedades.

A Figura 4.2 representa o documento XML Schema que valida o documento XML. Nela pode-se observar a estrutura principal do projeto. Foram ocultados os vários campos referentes aos objetos pois estes serão explicados de seguida.

Todos os objetos contêm um conjunto de propriedades gerais, que são utilizadas para definir o tamanho, a posição, a opacidade, o nome e o ângulo de rotação do objeto. Na Tabela 4.2, podemos observar todas as propriedades gerais e obrigatórias dos objetos. Para além das propriedades gerais dos objetos, cada objeto pode conter um conjunto de propriedades específicas utilizadas para desempenhar as suas funções únicas.

Outro grupo de propriedades que alguns objetos têm em comum são propriedades de texto, como o tamanho, o tipo e o estilo da fonte e a cor do texto. A Tabela 4.3 lista estas propriedades.

Para além das propriedades gerais dos objetos, cada objeto contém um conjunto de propriedades específicas:

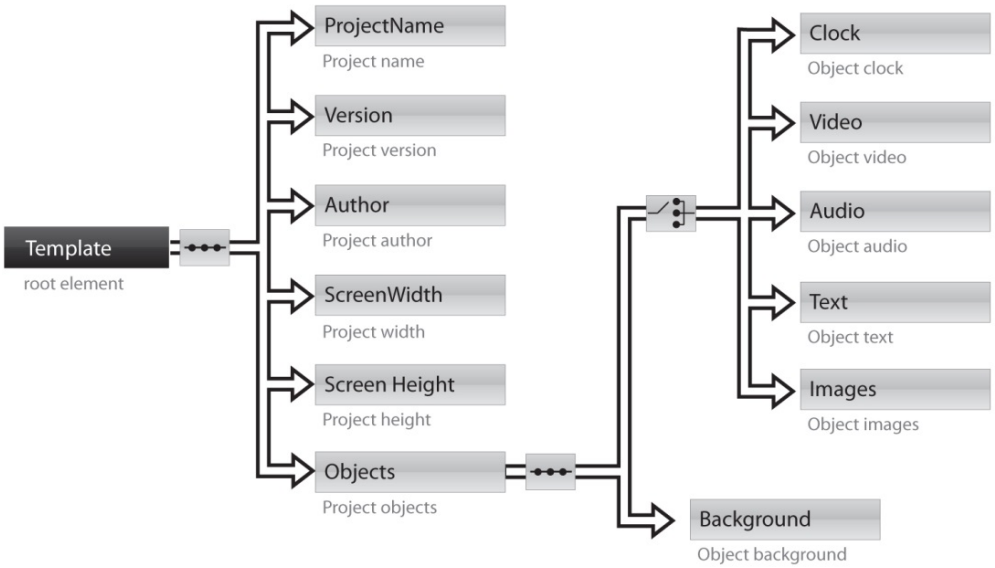

Figura 4.2: Esquema da linguagem declarativa com as propriedades dos objetos ocultas.

Campos	**Tipo**	**Propósito**
Identificação	Int	Identificar o objeto dentro do projeto
Nome	String	Nome do objeto
Largura	Double	Largura do objeto
Altura	Double	Altura do objeto
Coordenada X	Double	Coordenada X do objeto
Coordenada Y	Double	Coordenada Y do objeto
Opacidade	Double	Opacidade do objeto
Ângulo	Int	Ângulo de rotação do objeto

Tabela 4.2: Campos obrigatórios dos objetos.

Texto Contém as propriedades de texto (Tabela 4.3) e ainda informação sobre o *RSS feed* com informação do conteúdo e o texto estático que o utilizador pode querer utilizar em substituição do *feed*. É possível ainda animar o seu conteúdo, sendo necessário guardar a informação para efetuar estas animações. Esta informação está sumarizada na Tabela 4.4;

Imagens Contém as propriedades referentes à animação da imagem e uma lista com os *URLs* das imagens. A Tabela 4.5 sumariza esta informação;

Relógio Tem as propriedades de texto (Tabela 4.3), pois a forma de representar a informação temporal é em formato de texto. O relógio ainda necessita de informação referente à cor do seu fundo e ao formato da data e da hora. A Tabela 4.6 lista os campos necessários para representar esta informação;

Background Tem informação da sua cor (Tabela 4.7);

Vídeo e Áudio Ambos necessitam da mesma informação. A Tabela 4.8 sumariza esta informação.

Campos	Tipo	Propósito
Cor	String	Guarda o código hexadecimal da cor da fonte
Tamanho	Int	Informação do tamanho da fonte
Alinhamento	String	Informação do alinhamento do texto
Estilo da fonte	String	Informação se a fonte está em itálico ou não
Peso da fonte	String	Informação se a fonte está em negrito ou não
Informação se está em negrito e/ou em Itálico	String	Utilizado para guardar o valor da escolha do utilizador sobre se está em negrito e/ou em itálico
Padding de topo	Int	Guarda o valor da distância que o conteúdo deve ter em relação ao topo do objeto
Padding de esquerda	Int	Guarda o valor da distância que o conteúdo deve ter em relação ao lado esquerdo do objeto
Padding de direita	Int	Guarda o valor da distância que o conteúdo deve ter em relação ao lado direito do objeto
Padding de fundo	Int	Guarda o valor da distância que o conteúdo deve ter em relação ao fundo do objeto

Tabela 4.3: Propriedades do texto.

Campos	Tipo	Propósito
Cor de fundo	String	Guarda o código hexadecimal da cor do fundo
Wrap	String	O texto molda-se ao seu contentor
Fonte da informação	String	Se o texto vem de um *feed* ou de texto dado pelo utilizador
Texto do Utilizador	String	Texto inserido pelo utilizador
URL do *feed*	String	*URL* do *feed*
Caminho da informação dentro do *feed*	String	Utilizado para ir buscar um tipo de dados específicos no *feed*
Número de registos que vai buscar ao *feed*	Int	Utilizado para limitar o número de registos retirados do *feed*
Tipo de movimento do texto	String	Utilizado para indicar o tipo de movimento que o texto contém
Velocidade do movimento do texto	Double	Valor utilizado para definir a velocidade do texto

Tabela 4.4: Propriedades específicas do objeto texto.

4.4 Serviço Web

Para tornar possível a inserção dos recursos numa área de projeto remota e limitar o projeto a recursos só existentes nessa área, foi desenvolvido um Serviço Web que também torna possível a publicação dos projetos para o *BackOffice* da FutureboxTv e é ainda responsável por gerir a área de trabalho e a área de publicação. A Figura 4.3 mostra as diferentes interações do Serviço Web.

O Serviço Web responde aos seguintes pedidos:

Upload Necessita do nome do utilizador, da pasta de projeto e do ficheiro. Este pedido começa por verificar se todas as pastas existem e, se não existirem, cria-as. Caso o ficheiro com o mesmo nome já exista, este é sobreposto pelo novo. Com o sistema

Campos	Tipo	Propósito
Duração do elemento no ecrã	Int	Guarda a duração do elemento no ecrã
Tipo do efeito	String	Guarda o tipo do efeito
Lista de elementos	Lista de Strings	Lista de imagens

Tabela 4.5: Propriedades específicas do objeto imagens.

Campos	Tipo	Propósito
Cor de fundo	String	Guarda o código hexadecimal da cor do fundo
Informação se disponibiliza a data e/ou a hora	String	Guarda a informação se disponibiliza a data e/ou hora. Guarda também a informação sobre se é a data ou a hora que aparece em primeiro lugar
Formato da hora	String	Texto utilizado para formatar a hora
Formato da data	String	Texto utilizado para formatar a data

Tabela 4.6: Propriedades específicas do objeto relógio.

de diretorias criado, é criado o ficheiro;

Upload preview É utilizado para guardar a imagem de pré-visualização do projeto. Foi designada uma pasta padrão em todos os projetos onde esta imagem é armazenada. Sempre que uma nova imagem é guardada, a anterior é apagada. Este Serviço Web tem a responsabilidade de criar a diretoria, caso essa não exista, e apagar a imagem anterior, se existir. Para saber qual a pasta do projeto, é necessário passar a informação do utilizador e nome da pasta de projeto;

Listar recursos Necessita um nome de utilizador, o nome da pasta de projeto e o tipo de recurso (imagem, vídeo ou áudio) e procura todos os recursos desse género e retorna uma lista com o nome dos recursos;

Listar projetos A listagem de projeto necessita do nome do utilizador e pesquisa todos os projetos desse utilizador. Retorna uma lista onde cada elemento contém o nome do ficheiro do projeto, o nome do projeto especificado pelo utilizador e a descrição do projeto.

Publicar Para publicar um projeto, é necessário verificar se não existe já um projeto idêntico publicado e, se existir, é preciso eliminá-lo. Depois é copiado o projeto da área de trabalho para a área de publicação. Assim que todos os ficheiros estiverem na pasta de publicação, o projeto é publicado na FutureboxTv.

4.5 Editor

O Editor tem como função principal a criação de projetos. O utilizador pode criar os conteúdos dos projetos adicionando objetos multimédia à área de trabalho. Os objetos podem ser personalizados a partir de uma barra de propriedades que contém as propriedades do objeto selecionado.

Campos	**Tipo**	**Propósito**
Cor do fundo	String	Guarda o código hexadecimal da cor do fundo

Tabela 4.7: Propriedades específicas do objeto *background*.

Campos	**Tipo**	**Propósito**
URL do ficheiro	String	*URL* da localização do ficheiro
Número de vezes que repete	Int	Número de vezes que o vídeo repete
Está em ciclo	bool	Informação se está em ciclo infinito
Volume	Int	Volume do vídeo

Tabela 4.8: Propriedades específicas do objeto vídeo e do objeto áudio.

Quando o Editor é iniciado, o utilizador escolhe um projeto para trabalhar. Este projeto pode ser um projeto novo ou um projeto já existente. A Figura 4.4 ilustra a página inicial que permite estas operações. Esta página inicial bloqueia o resto do Editor e o utilizador só pode prosseguir depois de escolher uma das opções.

Como pode ser observado na Figura 4.4, para além do utilizador conseguir ver o nome do projeto e a sua descrição, pode também ver o aspeto real do projeto utilizando uma imagem da área de trabalho previamente guardada. Sempre que um projeto é salvo, esta imagem é gerada e guardada.

A Figura 4.5 mostra o Editor após a escolha do projeto. O Editor é composto pelos seguintes componentes: Barra de menus (01), Barra de *plugins* (02), Barra de objetos (03), Área de trabalho (04), Barra de propriedades (05).

4.5.1 Barra de menus

A barra de menus (zona 01 da Figura 4.5) permite ao utilizador gravar, visualizar e abrir um projeto. Estas ações não foram integradas nos outros painéis, pois não se encontram realmente no âmbito de nenhum deles. Esta barra situa-se no canto superior esquerdo, à semelhança da localização na maioria das aplicações, o que permite ao utilizador encontrá-la de forma intuitiva. Este painel permite efetuar as seguintes ações:

Novo projeto Esta ação é desencadeada ao pressionar o botão *New*. Nesta situação é perguntado ao utilizador se pretende salvar o seu trabalho. Após a decisão, o Editor apaga toda a informação da área de trabalho e cria um novo projeto. Após a criação do novo projeto, o Editor transmite esta informação ao Serviço Web para criação da área de projeto.

Abrir projeto Esta ação é desencadeada ao pressionar o botão *Open*. Nesta situação é perguntado ao utilizador se pretende salvar o seu trabalho. Após a decisão, o Editor pede ao Serviço Web a lista de projetos e o utilizador pode então escolher qual o projeto que pretende abrir.

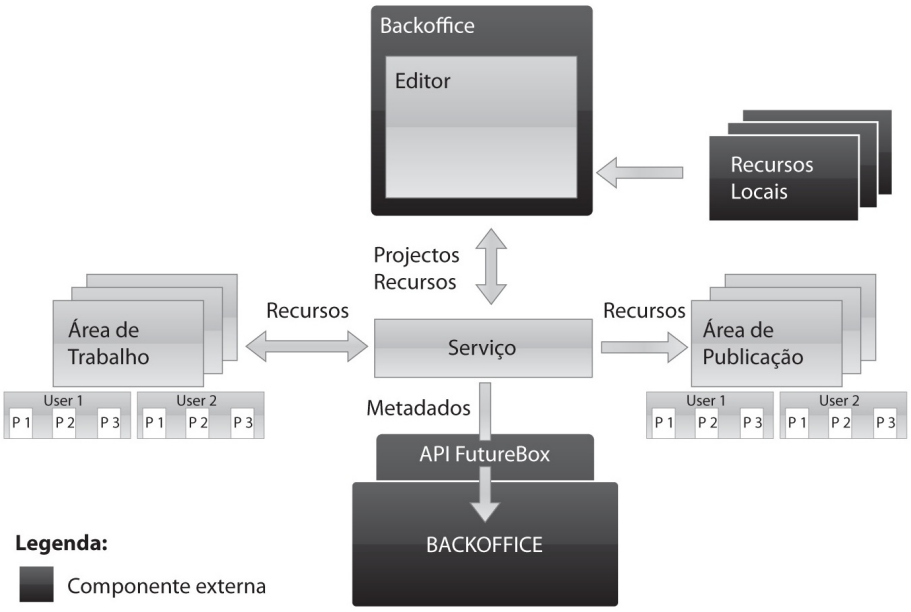

Figura 4.3: Interações do Serviço Web

Salvar um projeto Esta ação é desencadeada ao pressionar o botão *Save*. Quando o utilizador desencadeia esta ação, o Editor transmite a informação do projeto ao Serviço Web.

Definições do projeto Esta ação é desencadeada ao pressionar o botão *Settings*. Nesta situação aparece uma janela onde é possível alterar as propriedades do projeto.

Pré-visualizar um projeto Esta ação é desencadeada ao pressionar o botão *Preview*. Nesta situação é gerado um projeto com toda a informação da área de trabalho que é enviada ao Visualizador que o vai reproduzir.

Publicar um projeto Esta ação é desencadeada ao pressionar o botão *Publish*. Nesta situação são pedidos os dados de publicação e a confirmação de publicação. Após a confirmação do utilizador, o Editor faz um pedido ao Serviço Web para publicar o projeto que se encontra ativo.

Ajuda Esta ação é desencadeada ao pressionar o botão *Help*. Nesta situação é aberta uma nova página com informação de ajuda que o utilizador pode consultar.

A janela "Salvar projeto" aparece quando o utilizador está a gravar o projeto e desaparece assim que o projeto é guardado.

A janela de Definições do projeto permite ao utilizador alterar o nome do projeto e a sua descrição. Todas as alterações que o utilizador faça nesta janela são automaticamente guardadas.

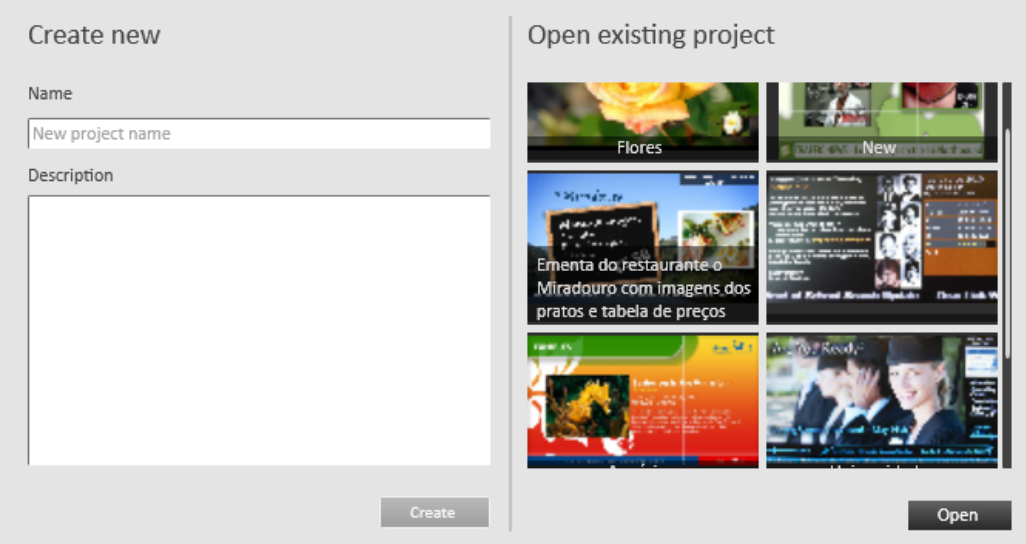

Figura 4.4: Página inicial

A janela de publicação é onde o utilizador pode definir o título que o projeto vai ter e confirmar se realmente quer publicar.

A janela "Abrir um projeto existente"(Figura 4.6) permite ao utilizador ver a lista de projetos disponíveis e selecionar um. Para além do nome e descrição o projeto é disponibilizado um *preview* de cada projeto disponível.

A janela "Criar novo" (Figura 4.7) é onde o utilizador pode criar um novo projeto. Tem de atribuir um título ao projeto e pode dar-lhe uma descrição.

4.5.2 Barra de *plugins*

A barra de *plugins* (zona 02 da Figura 4.5) contém os tipos de objetos que podem ser adicionados ao projeto e está situada à esquerda no Editor. Cada objeto é identificado por um nome e por uma imagem. Esta imagem é utilizada para identificar o tipo do objeto nos outros componentes do Editor.

Durante o seu desenho surgiram algumas questões críticas que se enumeram em seguida:

- Que informação disponibilizar para cada objeto? Como alguns objetos podem ser de um tipo muito específico, decidiu-se utilizar o nome de cada tipo de objeto para facilitar a sua identificação.
- Quantos objetos teria a barra? Como um dos requisitos da aplicação é esta ser extensível, é necessário considerar que a aplicação possa conter uma quantidade razoável de objetos.
- Que dimensão ocupar? Como o principal foco da aplicação não é a barra de *plugins*,

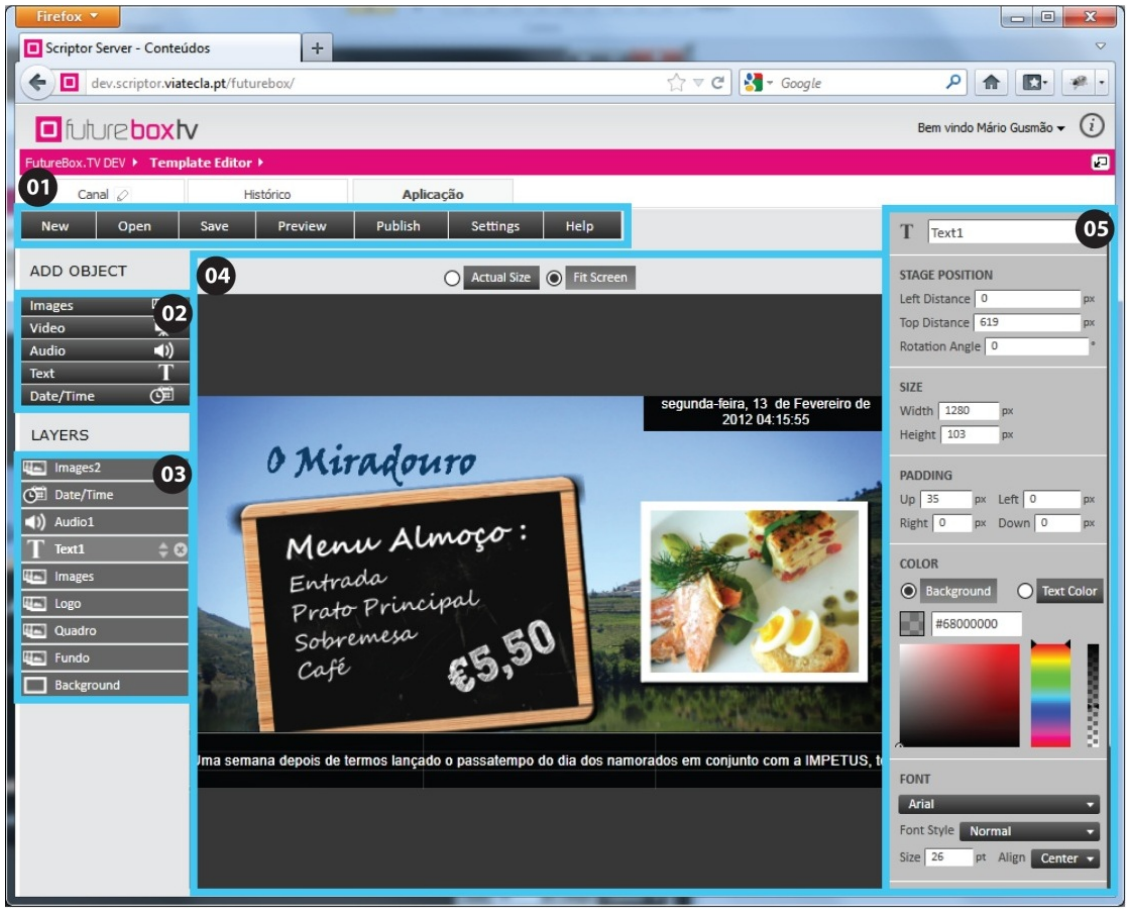

Figura 4.5: Editor

decidiu-se que esta deveria ocupar o menor espaço possível.

- Como fazer a interação com o utilizador? Como o utilizador consegue interagir diretamente com os objetos na área de trabalho, decidiu-se que cada tipo de objeto seria semelhante a um botão que, ao ser pressionado, adicionaria um objeto na área de trabalho com as definições padrão nas coordenadas x=0, y=0.

A relação com os outros objetos e a posição da barra na aplicação também foram fatores que influenciaram o desenho. Como deveria ocupar o menos espaço possível, a sua largura foi influenciada pela largura dos outros painéis que se encontram na mesma secção. Como em alguns painéis é necessário um ícone que represente o tipo de objeto, decidiu-se que cada elemento da barra de *plugins* deveria conter um ícone, e este seria o mesmo que os outros painéis iriam utilizar para representar os objetos.

Figura 4.6: Janela abrir um projeto.

4.5.3 Barra de objetos

A barra de objetos (zona 03 da Figura 4.5) permite ao utilizador ter noção de todos os objetos que fazem parte do projeto. Cada objeto é representado por um nome e por uma imagem que representa o seu tipo. Também existe uma cruz em cada objeto que permite ao utilizador remove-lo do projeto.

Como já referido, os objetos podem-se sobrepor uns aos outros na área de trabalho. Esta sobreposição pode ser alterada na barra de objetos selecionando um objeto e arrastando-o para a nova posição. Os objetos mais acima na barra de objetos encontram-se em posições superiores na área de trabalho.

4.5.4 Área de trabalho

A área de trabalho (zona 04 da Figura 4.5) é o elemento do Editor onde o utilizador foca mais a sua atenção e permite a interação direta com os objetos, tanto para alterar a sua posição como a sua dimensão.

Quando o utilizador insere um novo objeto no seu projeto, se este tiver uma componente visual, aparece na área de trabalho. Alguns objetos não necessitam de uma componente visual, como o caso do objeto áudio.

Quando um objeto é arrastado para cima de outro eles vão-se sobrepor. A ordem de sobreposição vai depender da ordem pela qual os objetos foram inseridos. A área de trabalho não permite alterar a ordem sobre a qual os objetos se sobrepõem. Esta ordem é alterada na barra de objetos (zona 03 da Figura 4.5). A área de trabalho disponibiliza

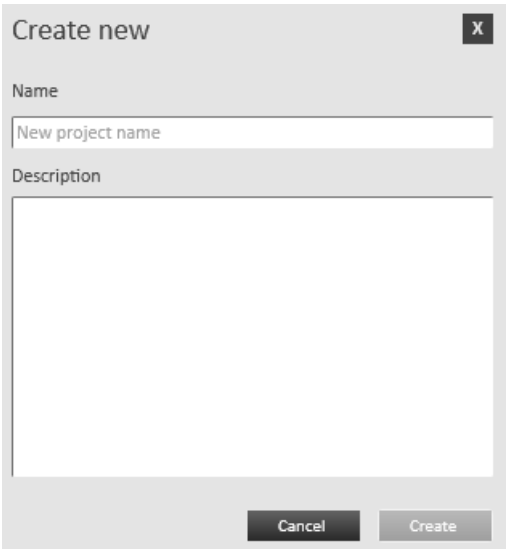

Figura 4.7: Janela criar novo projeto.

ao utilizador dois modos de visualização do projeto:

Ajuste no ecrã No modo ajuste no ecrã é aplicado um *zoom* ao projeto para que este ocupe a totalidade do ecrã conservando o rácio.

Tamanho real No modo de tamanho real, são disponibilizadas barras de *scroll* quando as dimensões da área do projeto ultrapassam as dimensões da área de trabalho. Quando a área de projeto é menor que a área de trabalho, a área de projeto aparece centrada na área de trabalho.

É importante que o objeto selecionado se destaque dos restantes na área de trabalho e que destaque as áreas onde o utilizador pode efetuar ações diretas sobre ele. As ações que o utilizador pode efetuar sobre o objeto são de redimensionar e de mover. A Figura 4.8 mostra as várias representações de interação.

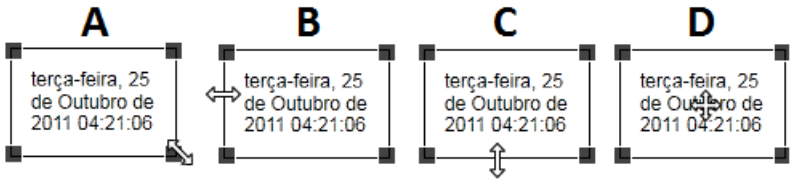

Figura 4.8: Representações das interações com os objetos na área de trabalho.

De forma a destacar que o utilizador pode redimensionar o objeto, foram inseridos quadrados nos cantos da seleção do objeto. Existem três formas de redimensionar os objetos:

1. Alterar a altura e a largura do objeto ao mesmo tempo. Este modo é ativado nos cantos dos objetos (Figura 4.8 A).

2. Alterar a largura do objeto. Este modo é ativado sobre as margens laterais dos objetos (Figura 4.8 B).

3. Alterar a altura do objeto. Este modo é ativado nas margens superior e inferior do objeto (Figura 4.8 C).

Para transmitir ao utilizador qual o modo de redimensionamento que está a utilizar, é alterado o cursor do rato para uma imagem que representa esta ação.

A ação de mover só pode ser realizada no interior da margem de seleção do objeto. Para reforçar ao utilizador que esta ação pode ser realizada, é alterado o cursor do rato (Figura 4.8 D).

Alguns objetos podem conter a necessidade de impedir que o utilizador os redimensione e/ou mova na área de trabalho; nestes casos são removidos do objeto todos os indicadores visuais da ação que se pretende limitar. Por exemplo, no caso de um objeto que tenha uma posição fixa, ao colocar o rato por cima do mesmo, o cursor não é alterado para a imagem que representa esta ação.

4.5.5 Barra de propriedades

A barra de propriedades (zona 05 da Figura 4.5) permite visualizar e alterar as propriedades dos objetos e está situada à direita do Editor, ocupando a totalidade da sua altura, pois não necessita de dividir o seu espaço com nenhum painel. A barra de propriedades está sempre visível para permitir a interação fácil com os objetos presentes na área de trabalho.

Sempre que não seja possível mostrar todas as propriedades de um objeto, é apresentada uma barra de *scroll*. Quando um objeto é selecionado, as suas propriedades são apresentadas na barra de propriedades.

Cada objeto contém uma lista das propriedades que devem ser mostradas na barra de propriedades. Quando um objeto é selecionado, esta lista é passada à barra de propriedades. Quando a barra de propriedades recebe uma lista com propriedades, elimina todas as propriedades que mostrava anteriormente e apresenta as novas. Para mostrar as novas propriedades é percorrida a lista e para cada uma, de acordo com o modelo de dados da propriedade, é inserido o componente visual correspondente.

Para que as alterações nas propriedades se propaguem até ao objeto selecionado, sempre que um objeto é selecionado, é passado à barra de propriedades. Como é realizado um *binding* entre o objeto e a barra de propriedades, quando uma propriedade é alterada na barra de propriedades, estas alterações refletem-se no modelo de dados do objeto e propagam-se a todos os componentes que lhe estão ligados.

4.6 Visualizador

A função do Visualizador é a de executar um projeto apresentando o seu conteúdo. Este componente está contido no Editor e nos clientes *WEB* e *Digital Signage*.

O utilizador, enquanto está a trabalhar no Editor, pode querer executar o Visualizador sem guardar o trabalho. Neste caso, o Visualizador recebe do Editor toda a informação do projeto.

Os clientes de *Digital Signage* e *WEB*, ao executarem o Visualizador, transmitem-lhe a informação do *URL* do projeto, que permite ao Visualizador buscar a informação do projeto e construir o conteúdo.

Quando o Visualizador recebe um projeto, começa por validá-lo. Se o projeto for válido, o Visualizador constrói o seu conteúdo percorrendo a lista de objetos existente no projeto para criar cada um dos objetos com as propriedades previamente definidas. Após serem gerados, os objetos são adicionados ao conteúdo.

É importante que o conteúdo exibido pelo Visualizador ocupe a maior área possível sem perder a proporção. Para conseguir isto, é aplicado *zoom* ao conteúdo do Visualizador. Como o *zoom* é aplicado de forma a manter a proporção do conteúdo, este nem sempre ocupa a totalidade da área disponível. As áreas adjacentes ao conteúdo estarão a preto para lhe dar destaque. A Figura 4.9 mostra o Visualizador a executar um projeto.

Figura 4.9: Visualizador a executar um projeto

4.7 Propriedades

A barra de propriedades, que foi criada para permitir alterar facilmente as propriedades dos objetos, é um conjunto de propriedades em sequência. Como não existe nenhum limite no número de propriedades nem no tamanho que uma propriedade pode ter, é adicionada uma barra de *scroll* para o utilizador poder visualizar a totalidade das propriedades.

Todos os objetos devolvem uma lista de modelos de propriedades que a barra de propriedades sabe implementar. Sempre que um novo objeto é selecionado, a barra de propriedades recebe uma nova lista de modelos de propriedades.

A Figura 4.10 apresenta a estrutura da barra de propriedades: ela contém a lista de modelos de propriedades e as várias propriedades. Cada propriedade tem um modelo de propriedade correspondente e constrói-se com as legendas corretas e com os valores que o objeto tem. Sempre que a barra de propriedades recebe uma nova lista de modelos de propriedades, remove as propriedades antigas e cria a nova lista de propriedades.

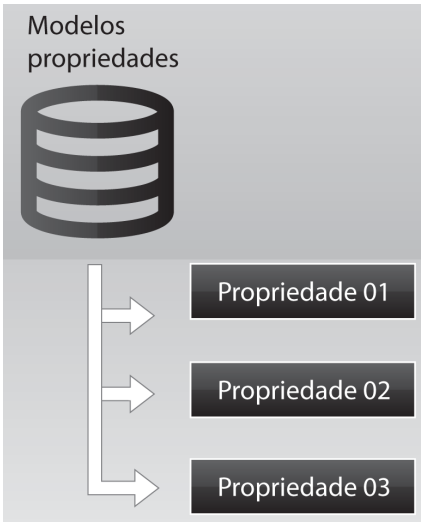

Figura 4.10: Componentes da barra de propriedades

Para cumprir todas as exigências dos objetos, foram desenvolvidas as seguintes propriedades:

- Campo de texto,
- Nome do objeto,
- *Slider*,
- Escolha de cor,
- Caixa de combinação,
- Propriedade dupla,
- Título de grupo,
- *Loop*,

- Tamanho,
- Visualizar o áudio e o vídeo,
- Visualizar elementos das imagens.

Criou-se um *popup* para mostrar as imagens, vídeos e ficheiros de áudio existentes no projeto. Esta informação não foi adicionada à barra de propriedades devido ao seu tamanho e por ser uma ação que o utilizador faz com pouca regularidade. Para manter a coerência foi criado mais um *popup* para definir o conteúdo do objeto de texto.

4.7.1 As propriedades

Campo de texto

A propriedade "Campo de texto"foi a primeira propriedade desenvolvida porque com ela é possível definir números e texto. O texto que aparece antes e depois da caixa de texto é definido pelo objeto que invoca a propriedade. Quando o objeto que invoca a propriedade não define nenhum valor para o texto, este aparece vazio e a caixa de texto expande para ocupar toda a área disponível.

A Figura 4.11 apresenta um exemplo da propriedade "Campo de texto"onde se definiu uma legenda "*Left Distance*" e uma unidade "px". O valor 0 corresponde ao valor atual da propriedade.

Figura 4.11: Propriedade Campo de texto

Nome do objeto

A propriedade "Nome do objeto"apresenta uma imagem do objeto e uma caixa de texto para definir o nome do objeto. A imagem do objeto é a mesma imagem que é utilizada para identificar o objeto nos outros componentes do Editor.

A Figura 4.12 apresenta dois exemplos da propriedade "Nome do objeto". O primeiro exemplo refere-se ao objeto *Background* onde foi atribuído o nome "Fundo". O segundo exemplo refere-se ao objeto "Imagens"onde foi atribuído o nome "Imagem Fundo".

Figura 4.12: Propriedade nome objeto

Slider

O "*Slider*"foi criado para escolher um valor entre um intervalo de possíveis valores. O *Slider* mostra o valor atual da propriedade, sendo possível atribuir uma unidade a esse valor, como por exemplo percentagem.

A Figura 4.13 apresenta três exemplos da utilização da propriedade "*Slider*". No primeiro exemplo, o mínimo e o máximo são palavras e o valor não tem uma unidade definida. No segundo exemplo, o mínimo e o máximo são números com uma unidade atribuída e o valor atual também tem uma unidade. No terceiro exemplo, todos os campos são números mas nenhum tem uma unidade definida.

Figura 4.13: Propriedade *Slider*

Escolha de cor

A propriedade "Escolha de cor"serve para ajudar o utilizador a atribuir cores visualmente. O utilizador contém 3 paletes: uma para definir a tonalidade da cor, outra para escolher a cor e a terceira é para definir a opacidade. Ainda é possível ao utilizador definir manualmente o código da cor.

Alguns objetos contêm duas propriedades para a "Escolha de cor". Como forma de economizar espaço na barra de propriedades, foi adicionada à propriedade de controlo de cor a opção para gerir duas propriedades de cor distintas. Quando a propriedade de cor tem de gerir duas cores, são adicionados no seu topo 2 botões rádio que contêm o nome da propriedade e onde o utilizador pode alterar entre eles para definir a cor pretendida.

A Figura 4.14 mostra dois exemplos da propriedade de "Escolha de cor". No primeiro exemplo, o objeto contém duas propriedades de cor, uma para o *background* do objeto (a que está selecionada) e outra para o texto do objeto. No segundo exemplo, o objeto só contém uma propriedade de cor.

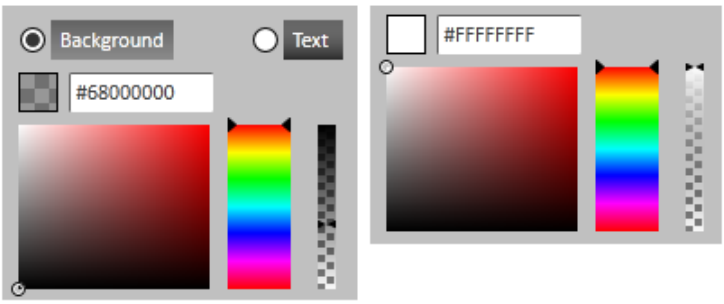

Figura 4.14: Propriedade Escolha de cor

Caixa de combinação

A "Caixa de combinação"permite apresentar um conjunto de possibilidades ao utilizador e impedir que este escolha mais que uma opção. Quando a caixa está fechada, ocupa muito pouco espaço na barra de propriedades e, quando está aberta, sobrepõe-se aos objetos já existentes. À "Caixa de combinação"desenvolvida ainda foi adicionada a possibilidade do objeto acrescentar uma legenda antes da caixa.

Todos os elementos da caixa são definidos pelo objeto que a gera e este sabe sempre qual o elemento selecionado. A caixa de combinação não contém qualquer limitação em relação ao número de elementos.

A Figura 4.15 apresenta duas "Caixas de combinação". A primeira caixa não tem legenda e contém oito elementos. Como não tem legenda, esta caixa ocupa a totalidade da largura da barra de propriedades e quando está aberta oculta quase a totalidade dos elementos abaixo dela. A segunda caixa tem uma legenda e o seu tamanho é mais reduzido.

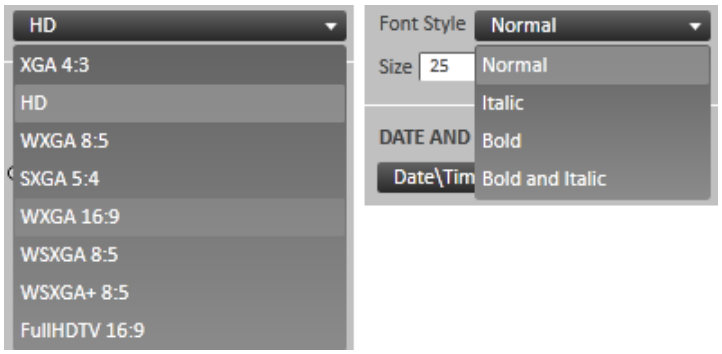

Figura 4.15: Propriedade Caixa de combinação

Propriedade dupla

A "Propriedade dupla"surgiu da ideia de querer juntar duas propriedades já existentes na mesma linha, e reutilizar trabalho já desenvolvido. A Figura 4.16 mostra dois exemplos

desta propriedade. No primeiro exemplo, a "propriedade dupla"contém dois "Campos de texto"e, no segundo exemplo, contém um "Campo de texto"e uma "Caixa de combinação". É preciso ter atenção ao tamanho das propriedades que se quer utilizar dentro da propriedade dupla pois este fica reduzido a menos de metade do que teria se estivesse na barra de propriedades.

Figura 4.16: Propriedade dupla

Título de grupo

Para melhor organizar as propriedades decidiu-se que estas estariam inseridas em grupos e que cada grupo conteria um número reduzido de propriedades. Como forma de separar e identificar os grupos foi criada a propriedade "Título de grupo". Esta propriedade contém uma linha que assinala o fim do grupo anterior e um título do novo grupo. É possível não definir título; este caso normalmente é utilizado após a última propriedade da barra de propriedades.

A Figura 4.17 contém dois exemplos da propriedade "Título de grupo". No primeiro exemplo é definido um título, no segundo não é definido nenhum título e apenas mostra a linha de fim de grupo.

Figura 4.17: Propriedade Título de grupo

Loop

Esta propriedade contém dois estados, um estado de "*Loop*"e um estado em que o utilizador pode definir o número de repetições. Quando a propriedade entra em estado de "*Loop*", a caixa de texto fica bloqueada. A caixa de texto serve para o utilizador definir o número de repetições.

A Figura 4.18 mostra duas imagens desta propriedade, no primeiro caso está no estado normal e no segundo está no estado *Loop*.

Figura 4.18: Propriedade de *Loop*

Tamanho

A propriedade "Tamanho"(Figura 4.19) permite definir o tamanho do objeto e em dois casos particulares, objeto Imagens e Vídeo, ainda permite colocar o objeto no seu tamanho natural. Para colocar o objeto no seu tamanho natural foi adicionado um botão que, quando pressionado, obtém o tamanho natural do objeto e redimensiona-o.

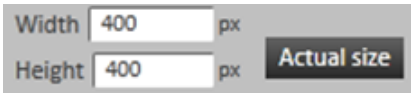

Figura 4.19: Propriedade de Tamanho

Visualizar o vídeo e o áudio

A propriedade para "Visualizar o vídeo e o áudio"serve para informar o utilizador sobre o conteúdo associado ao objeto. Esta propriedade contém o nome do ficheiro e no caso de o objeto ser vídeo contém um *preview* do vídeo. A Figura 4.20 mostra esta propriedade para um objeto de vídeo e um objeto de áudio.

Figura 4.20: Propriedade Visualizar o vídeo e o áudio

No topo desta propriedade existe um botão para atribuir ou substituir o elemento do objeto. Quando este botão é selecionado é apresentado ao utilizador um *popup* para escolher o ficheiro de áudio ou de vídeo.

Visualizar elementos do objeto Imagens

Esta propriedade contém um botão, um grupo de transição e a lista de elementos. A Figura 4.21 apresenta dois exemplos desta propriedade. O primeiro contém várias imagens e o segundo só contém uma imagem. O botão é utilizado para apresentar ao utilizador um *popup* que permite adicionar uma imagem, pela seleção do ficheiro relacionado.

O grupo de transição só está visível se o utilizador tiver mais que uma imagem. Contém um título, uma caixa de combinação e uma caixa de texto: o título é utilizado para identificar o grupo; a caixa de combinação permite ao utilizador definir o efeito de transição de imagem; a caixa de texto é utilizada para definir o tempo em que cada imagem fica visível antes de ser substituída pela seguinte.

Figura 4.21: Propriedade Visualizar elementos do objeto Imagens

A lista de imagens permite ao utilizador visualizar as imagens existentes no objeto, definir a sua ordem ou efetuar a sua remoção. O utilizador pode adicionar as imagens que pretender. Se estas forem num número elevado a barra de propriedades apresenta um *scroll*. Para alterar a posição de uma imagem basta clicar nela e arrastá-la para a posição pretendida.

4.7.2 Os popups

Popup escolher vídeo, áudio ou imagem

Os *popup* escolher vídeo, áudio ou imagem são idênticos entre si excepto nos elementos multimédia que apresentam. Todos eles, quando iniciam, pedem ao servidor uma lista dos respetivos recursos. Contêm um botão de *upload* que permite ao utilizador adicionar recursos existentes no seu computador local. Quando o utilizador carrega neste botão, é apresentado um diálogo para selecionar um recurso dos seus recursos locais. Se o utilizador selecionar um recurso com um nome igual a um recurso já existente, é perguntado ao utilizador se quer substituir o recurso existente.

Por uma questão de segurança, os serviços têm limite no tamanho dos ficheiros que recebem

e existe também um limite de tempo para o pedido a um serviço. Por estas razões, para fazer o *upload* dos ficheiros, primeiro é necessário dividi-los em pequenas partes.

O *popup* de vídeo (Figura 4.22) permite realizar o *upload* de ficheiros MP4 e WMV. Cada elemento da lista tem um botão para permitir ao utilizador pré-visualizar o vídeo diretamente da lista de objetos.

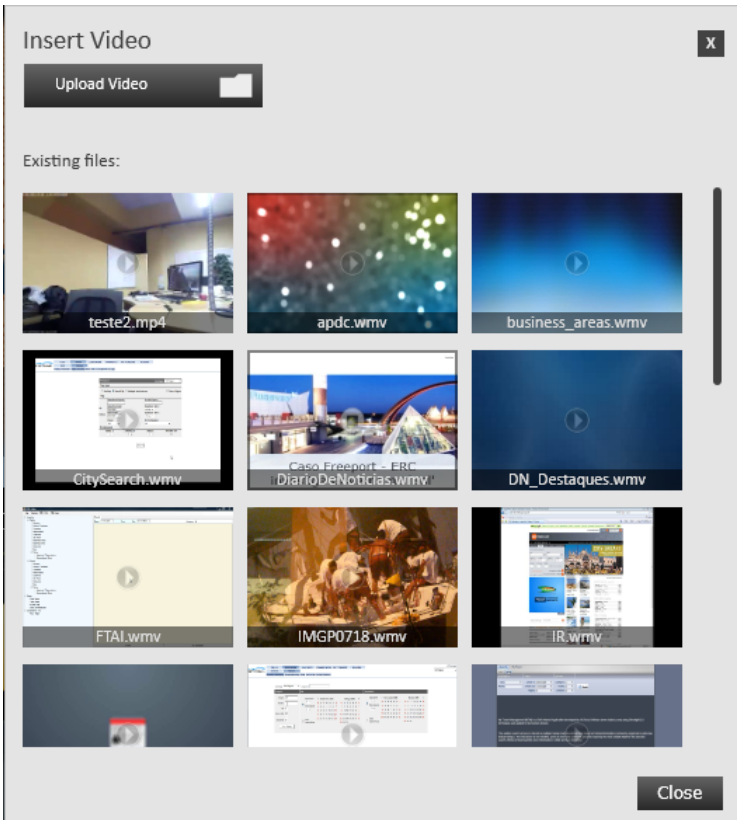

Figura 4.22: *Popup* de vídeo

O *popup* de áudio permite realizar o *upload* de ficheiros MP3 e WMA. Cada elemento da lista tem um botão para permitir ao utilizador ouvir o áudio diretamente da lista de objetos. Os objetos de áudio não têm componente visual pelo que foi adicionado um ícone para o representar para manter a coerência com o *popup* de vídeo e o *popup* de imagem.

O *popup* de imagem (Figura 4.23) permite realizar o *upload* de ficheiros JPEG, JPG e PNG. É o único *popup* em que o elemento selecionado é acrescentado à lista de objetos existentes em vez de substituir o que lá está.

Popup seleção de texto

O *popup* de texto é utilizado para definir a fonte do objeto de texto. Este *popup* é ativado através do botão de seleção de texto existente nas propriedades do objeto de texto. O utilizador pode escolher entre dois modos de entrada de texto, o modo *RSS Feed* e o modo

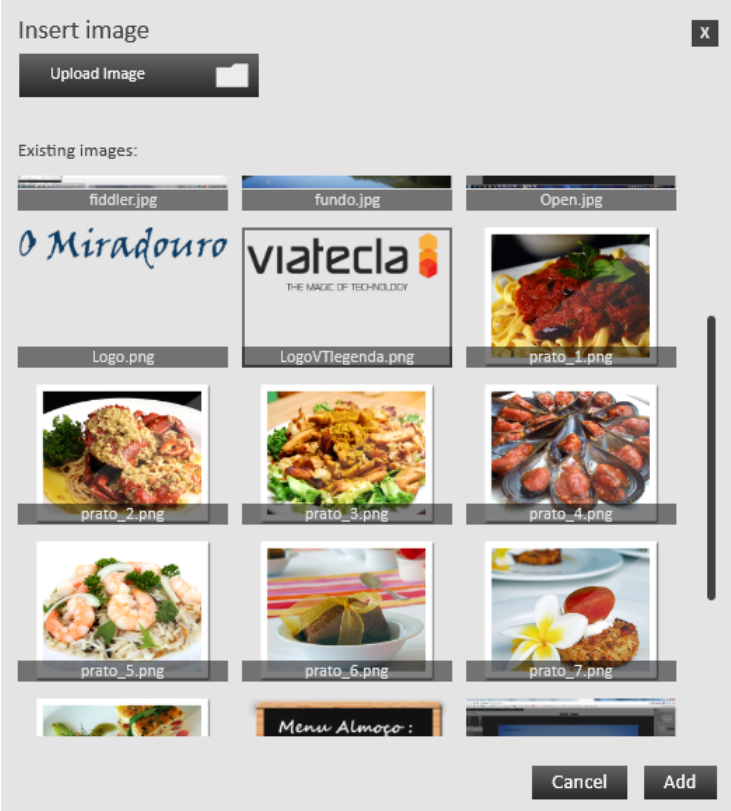

Figura 4.23: *Popup* de imagem

texto *input*.

Um *RSS Feed* representa um conteúdo *online* no formato *RSS*. Ao defini-lo como o modo principal, o Visualizador sempre que é inicializado, vai buscar os conteúdos mantendo-os atualizados. Neste modo (Figura 4.24) o utilizador determina qual o texto a extrair do *RSS Feed*: é definido um *URL* para indicar a fonte *RSS Feed* e um *XPath* para definir os campos que vão ser mostrados. Ainda é permitido ao utilizador definir o número de registos que pretende ir buscar.

O *XPath* é uma forma de filtrar a informação existente no *RSS Feed*, pois normalmente contém muita informação e nem toda interessa mostrar. Ao definir-se o caminho só a informação existente nesse caminho é utilizada.

Se o objetivo do utilizador for acrescentar um texto estático que não sofra alterações, o modo *text input* (Figura 4.25) é o modo ideal. Quando o objeto é construído é definido um conteúdo e o objeto utiliza sempre esse conteúdo.

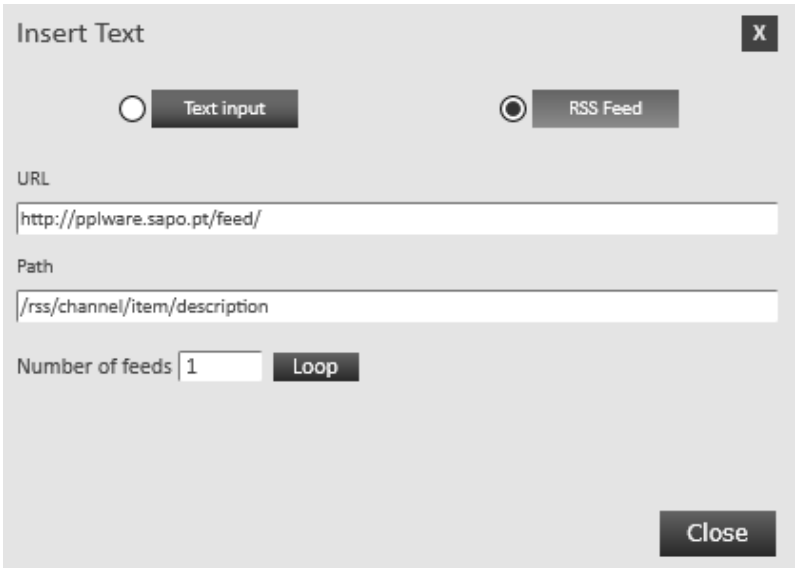

Figura 4.24: *Popup* seleção de texto no modo *RSS Feed*

4.8 Objetos

Foram analisados alguns exemplos de conteúdos para identificar diversos tipos de objetos. A base dos objetos criados, para efeitos de composição, teve como recursos principais texto, vídeo, áudio, imagem e indicadores de tempo. De acordo com esta análise, foram criados os seguintes tipos de objetos:

- Texto, utilizado para representar e animar texto;
- Vídeo, utilizado para permitir a reprodução de conteúdos de vídeo;
- Áudio, utilizado para permitir a reprodução de conteúdos de áudio;
- Imagem, utilizado para colocar imagens que permitam enriquecer e tornar o trabalho mais apelativo. Também pode ser utilizado como *slideshow* para representar um conjunto de imagens;
- Data e Hora, utilizado para representar a data e a hora;
- *Background*, utilizado como base para os outros objetos. Permite também alterar a resolução da área de trabalho.

Cada objeto tem associado um conjunto de propriedades que permitem ao utilizador personalizar o seu comportamento e aparência. Todos os objetos são constituídos por um modelo de dados e por uma *view*. O modelo de dados permite guardar os valores das propriedades dos objetos. A *view* é a representação visual do objeto. O modelo de dados associados deve representar o objeto de acordo com o valor das propriedades nele contidas.

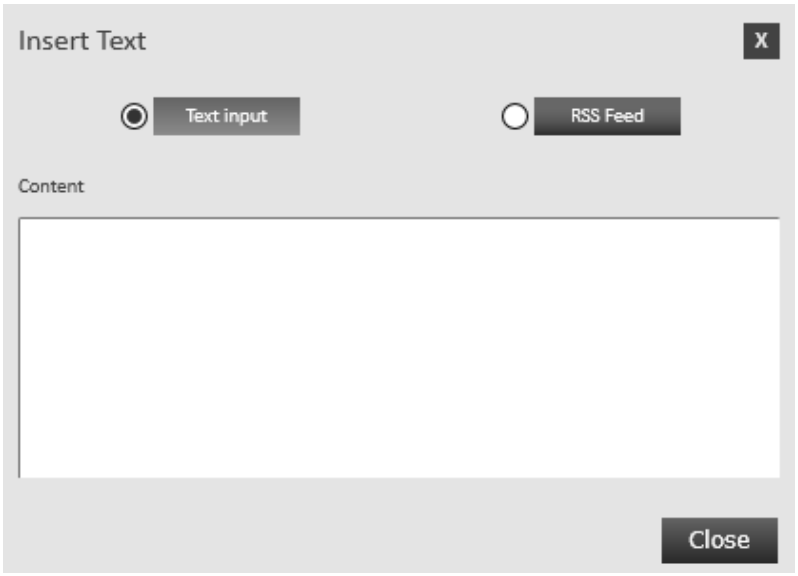

Figura 4.25: *Popup* seleção de texto no modo texto input

O painel de projeto é o elemento que contém todos os objetos existentes no projeto, o seu tamanho é definido pelo utilizador. Todos os objetos existentes no projeto estão contidos no painel de projeto. As posições dos objetos neste painel guiam-se por um sistema de coordenadas onde o ponto (0,0) se encontra no canto superior esquerdo do painel de projeto.

Para tornar os objetos independentes do Editor e do Visualizador, criou-se uma "Moldura", responsável por controlar o tamanho e a posição dos objetos. Quando se cria um objeto, é adicionada uma moldura ao painel de projeto que contém a *view* do objeto. A moldura obtém o tamanho, a posição e as propriedades do objeto.

Como os objetos são adicionados à área de projeto dentro de uma moldura e esta tem propriedades e comportamentos distintos dos objetos, os problemas de programação causados pela moldura são mais facilmente detetados.

Todos os objetos contêm dados iniciais, que são os que a *view* utiliza para representar o objeto quando este é iniciado. Os valores utilizados para este fim foram configurados para que o utilizador consiga perceber rapidamente o que o objeto é e qual a sua função.

Os objetos no Editor são estáticos e preparados para consumir a menor quantidade de recursos possível, pois o objetivo do Editor é construir o conteúdo. Os objetos no Visualizador podem conter animações e transmitir vídeos. Como o mesmo tipo de objeto pode ter um comportamento diferente no Editor e no Visualizador, decidiu-se criar objetos diferentes para cada um deles.

Todos os objetos são bibliotecas distintas estando isolados entre si, do Editor e do Visualizador. Esta funcionalidade também permite criar e adicionar objetos novos mais facilmente, e de forma independente do modo em como vão ser construídos e/ou executa-

dos.

4.8.1 Biblioteca de controlo de objetos

Para o controlo dos objetos existentes tanto no Editor como no Visualizador, foi criada uma biblioteca que contém a informação de todos os tipos de objetos existentes. Quando se desenvolve um tipo de objeto novo basta adicionar a esta biblioteca a informação do objeto e tanto o Visualizador como o Editor começam a conseguir interagir com o novo tipo de objeto.

Sempre que o Editor inicializa, necessita de saber quais os objetos que o utilizador pode ter à sua disposição. Como estes podem variar, o Editor efetua um pedido à biblioteca de Controlo de objetos. Com esta informação, o Editor preenche a sua barra de *plugins*. A Figura 4.26 mostra como o Editor interage com a biblioteca de controlo de objetos.

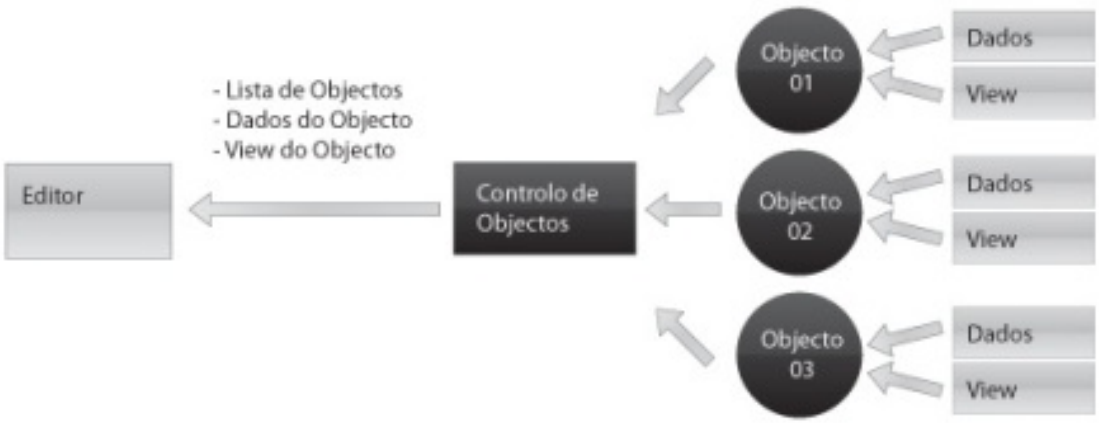

Figura 4.26: Interacção entre o Editor e a biblioteca de controlo de objetos

Quando o utilizador carrega num objeto da barra de *plugins*, o Editor solicita à biblioteca de Controlo de objetos o modelo de dados do objeto e a *view* do objeto. O modelo de dados do objeto é adicionado a uma lista que contém todos os modelos de dados dos objetos utilizados. A *view* do objeto é adicionada à área de trabalho do Editor. O objeto retornado nestas condições vem com as propriedades padrão.

Quando o Editor pretende um objeto com propriedades diferentes das propriedades padrão, pede o objeto à biblioteca de controlo de objetos, mas passa-lhe a informação das suas propriedades. A biblioteca inicializa o objeto com as propriedades especificadas e retorna ao Editor o modelo de dados do objeto e a *view* do objeto.

O Visualizador interage com a biblioteca para o controlo de objetos de uma forma idêntica ao Editor, a única diferença entre eles é que o Visualizador não necessita da lista de objetos nem de objetos com definições padrão.

4.8.2 *Data Binding*

O Editor necessita de mostrar a informação de um objeto em três lugares distintos: na área de trabalho, na barra de propriedades e na barra de objetos. Quando é alterada num destes componentes, a informação é propagada para todos os outros de uma forma quase instantânea através do *Data Binding*. Na Figura 4.27 pode-se ver os componentes que necessitam de interagir com o modelo de dados do objeto.

Figura 4.27: Interações do modelo de dados dos objetos

O *Data Binding* do Silverlight também tornou possível que os objetos fossem criados como entidades externas ao Editor. O Editor apenas tem conhecimento da classe base do modelo de dados dos objetos e da classe base do *view* dos objetos. Para representar o objeto é efetuado um *binding* entre o modelo de dados e o *view* do objeto. Ao se realizar este *binding*, a *view* pode mostrar e alterar os dados contidos no modelo de dados do objeto.

4.8.3 Os objetos

Background

Os objetos podem ser movidos para fora da área de projeto, logo é necessário que o utilizador tenha uma representação visual das fronteiras da área de projeto. Esta fronteira é delimitada pelo objeto de *background* que ocupa a totalidade da área de projeto.

O *background* é um objeto que existe em todos os projetos, em que só pode existir um e o utilizador não pode inserir mais nem apagar o existente. Está posicionado atrás de todos os outros objetos e todos os novos objetos são inseridos à sua frente.

Para dinamizar a funcionalidade de alteração das dimensões do projeto foi elaborada uma lista com as várias resoluções mais utilizadas que o utilizador pode escolher para redimensionar o projeto; no entanto também é possível definir as dimensões do projeto manualmente para criar resoluções mais específicas.

A cor por defeito do *background* é branca para se destacar bem da área de trabalho, no entanto é possível modifica-la escolhendo outra cor nas propriedades do *background*. Para utilizar uma imagem como fundo, adiciona-se uma imagem ao projeto que cubra a totalidade do *background*.

Vídeo

Como o vídeo é uma ótima forma de passar informação dinamicamente decidiu-se criar um objeto vídeo. Este objeto contém as seguintes propriedades:

- Texto de identificação. Identificadores personalizados permitem ao utilizador identificar mais facilmente um objeto específico no meio de um grupo grande de objetos;
- Controlo de volume. Possibilita controlar o nível do áudio ou colocar o vídeo sem áudio;
- Repetição. Permite definir o número de vezes que o elemento vai ser repetido ou se vai repetir-se em contínuo;
- Redimensionar. A área de vídeo pode ser redimensionada na barra de propriedades ou diretamente na área de trabalho;
- Tamanho natural. Altera as dimensões do objeto de vídeo para as dimensões naturais;
- Mover. O vídeo pode ser movido na barra de propriedades ou diretamente na área de trabalho;
- Rodar. Permite especificar um ângulo de rotação para o vídeo;
- Opacidade. Permite definir a opacidade do vídeo;
- *Upload*. Permite ao utilizador realizar o *upload* de ficheiros de vídeos que tenha no seu sistema de ficheiros local para a área de projeto;
- Selecionar um ficheiro. Permite ao utilizador selecionar um vídeo entre os vídeos existentes na área de projeto. Na lista de vídeos é apresentada uma miniatura do vídeo que pode ser reproduzido pelo utilizador.

Áudio

Colocar um som de fundo num conteúdo é uma forma de atrair a atenção das pessoas para esse conteúdo. Decidiu-se adicionar um objeto de áudio para ser possível adicionar sons ao conteúdo. O objeto áudio não tem componente visual nem as propriedades referentes à componente visual, mas tem as seguintes propriedades:

- Texto de identificação,
- Controlo de volume,
- Repetição,
- Redimensionar,
- *Upload*,
- Selecionar um ficheiro.

Imagens

O uso de imagens é indispensável quando se pretende criar conteúdos mais atrativos, tendo-se criado um objeto imagem. Posteriormente foi criado um objeto Imagens que conterá um conjunto de imagens, possibilitando uma transição entre os elementos.

O objeto Imagens tem as seguintes propriedades:

- Texto de identificação,
- Redimensionar,
- Tamanho natural,
- Mover,
- Rodar,
- Opacidade,
- *Upload*,
- Adicionar uma imagem. Permite que o utilizador adicione uma imagem a uma lista de imagens;
- Efeito de transição entre imagens. Quando existe mais que uma imagem é possível definir que efeito irá ser executado quando a imagem muda;
- Duração. Quando existe mais que uma imagem é possível definir quantos segundos a imagem irá permanecer no ecrã.

Texto

O texto é uma das melhores formas de passar informação. Uma das formas de se conseguir texto novo e sempre atualizado é ir buscar informação a um *RSS Feed*. O objeto de texto possui a funcionalidade de ir buscar o conteúdo a um *RSS Feed*. Para tal tem de ser definido o *URL* do *feed* e o seu caminho para o campo a ser mostrado. Como muitas vezes os *feeds* contém grandes quantidades de informação foi adicionada a propriedade número de *feeds* onde define o número de conteúdos (ou todos). Também é possível a criação do objeto definindo texto estático.

Uma vez que o texto pode não caber na área definida foi implementada a funcionalidade de movimento de texto. O texto só pode mover-se em direções que não interferem com a leitura do utilizador, ou seja, da esquerda para a direita e de baixo para cima. O utilizador na barra de propriedades encontra a opção para escolher a direção do movimento do texto e a velocidade do seu deslocamento.

O objeto texto também contém as seguintes propriedades:

- Texto de identificação,
- Redimensionar,

- Tamanho natural,
- Mover,
- Rodar,
- Opacidade,
- *Padding*. É possível definir a distância que o texto tem em relação às fronteiras (esquerda, direita, topo e do fundo) da caixa de texto;
- Fonte. Permite alterar a fonte do texto;
- Tamanho da fonte. Permite alterar o tamanho da fonte;
- Estilo da fonte. Permite alterar o estilo da fonte;
- Cor do texto e da caixa de texto. É possível definir a cor e a opacidade do texto e da caixa de texto.

Relógio

Um objeto que represente a hora e a data num conteúdo é uma forma de informar os espetadores. Um objeto deste género precisa de estar atualizado, caso contrário perde a sua utilidade. Para que isto aconteça decidiu-se que o objeto obteria a informação temporal a partir do sistema que o está a reproduzir.

O objeto relógio permite ao utilizador escolher se quer mostrar a data e a hora ou se pretende apenas mostrar uma delas. Com esta função e com dois objetos relógio, um definido para mostrar só a data e outro para mostrar só a hora, é possível ter um aspeto completamente diferente para cada um deles. Foi também acrescentada a opção para o utilizador definir o formato da data e o formato da hora.

A informação da hora e da data é representada por texto e é possível alterar algumas propriedades de texto para melhor personalizar o aspeto do relógio. O objeto relógio também contém as seguintes propriedades:

- Texto de identificação,
- Redimensionar,
- Tamanho natural,
- Mover,
- Rodar,
- Opacidade,
- *Padding*,
- Fonte,
- Tamanho da fonte,
- Estilo da fonte,
- Cor do texto e da caixa de texto.

Capítulo 5

Testes de usabilidade

Atualmente, para que uma aplicação tenha sucesso no mercado, é necessário que cumpra o seu propósito (uma condição já dada como adquirida por parte do utilizador final), tenha um *layout* atrativo e, mais importante, que tal seja realizado através de uma fácil usabilidade.

Uma aplicação com boa usabilidade motiva os seus utilizadores para a sua utilização, contrariamente ao que acontece no caso de uma aplicação com má usabilidade. De forma a testar as interfaces, é necessário que sejam realizados testes de usabilidade, tendo em conta os utilizadores finais a que a aplicação se destina.

Neste capítulo são referenciados os testes de usabilidade necessários para validar a interface do projeto desenvolvido. Estes testes não foram aplicados pois não foi possível juntar o número de candidatos necessários e efetuar os testes no prazo fornecido para desenvolver e entregar este projeto.

5.1 Público Alvo

O objetivo principal do desenvolvimento destes testes de usabilidade é validar se a aplicação desenvolvida é intuitiva, de fácil utilização e identificar futuras melhorias a efetuar. Para tal, a maioria dos utilizadores deste teste de usabilidade deverão ter pouca ou nenhuma experiência no uso de editores (secção 3.2) e conhecimentos básicos de informática numa perspetiva de utilizador final. Pretende-se que 35% dos utilizadores não tenham nenhuma experiência com editores, 45% tenham alguma experiência e 20% tenham muita experiência. O maior grupo é o de utilizadores com alguma experiência pois a maioria das pessoas encontra-se neste grupo. Consideram-se utilizadores:

- Com nenhuma experiência: utilizadores que nunca tenham utilizado um editor;
- Com alguma experiência: utilizadores que já tenham utilizado um editor;
- Com muita experiência: utilizadores que usem um editor frequentemente.

Os testes devem ser realizados por pessoas que tenham experiência na divulgação e promoção de produtos. Estima-se que 25% dos utilizadores sejam *designers*, 45% sejam gestores de informação e 30% sejam gestores de pequenos negócios. Entende-se por:

- Gestores de informação: utilizadores que trabalham no departamento de divulgação e promoção de produtos dentro das empresas;
- Gestores de pequenos negócios: responsáveis por pequenas empresas que sejam eles próprios a efetuar a divulgação e a promoção dos seus produtos;
- *Designers*: utilizadores que trabalhem na secção de *design*.

A tabela 5.1 mostra a interceção entre os 2 grupos.

	Designers	Gestores de informação	Gestores de pequenos negócios	**Total**
Sem experiência com editores	5%	20%	10%	**35%**
Alguma experiência com editores	10%	20%	15%	**45%**
Muita experiência com editores	10%	5%	5%	**20%**
Total	**25%**	**45%**	**30%**	**100%**

Tabela 5.1: Interceção entre os 2 grupos de utilizadores.

Para os resultados do teste serem significativos, estimou-se que seria necessário ter no mínimo um número de 80 inquiridos.

5.2 Metodologia

Para testar a aplicação, é pedido ao utilizador para criar um ecrã de informação, utilizando a aplicação desenvolvida.

A realização dos testes deve ser filmada e um monitor presente deve identificar e registar as dificuldades sentidas pelos utilizadores. Nos primeiros dois minutos do teste, será feita uma introdução à aplicação. Os utilizadores serão testados individualmente para evitar a troca de influências. O teste deve ter uma duração máxima de 25 minutos. O monitor deve cronometrar o tempo que o utilizador leva a realizar o teste.

Para criar o ecrã de informação, é necessário criar um novo projeto na aplicação, adicionar vários objetos à área de trabalho e alterar as suas propriedades.

Foi criado um *layout* de um suposto restaurante "O Miradorouro" com vários objetos que o utilizador deve tentar igualar. A Figura 5.1 mostra o *layout* desenvolvido, que é composto por quatro objetos de imagem, um objeto data e hora e um objeto texto.

Figura 5.1: *Layout* restaurante "O Miradorouro"

O objeto data e hora contém a data e a hora atual. O objeto texto vai buscar a informação a um *feed* de notícias e aplica uma animação ao texto para ir passando da esquerda para a direita. Os objetos imagem contêm: imagem de fundo, imagem do logótipo, imagem com a descrição do menu e respetivo preço e *slideshow* de pratos.

Para a criação deste *layout* são necessárias várias imagens que serão fornecidas ao utilizador durante a realização do teste. Também são fornecidas as informações para aceder ao *feed* do objeto de texto.

5.3 Recursos

Como o objetivo do teste não é tratar imagens, é necessário fornecer aos utilizadores um conjunto de imagens para o desenvolvimento do projeto. As imagens necessárias são:

- Fundo: Imagem de fundo do projeto.
- Logo: Logótipo da empresa.
- Quadro com o Menu: Imagem de um menu.
- Fotos dos pratos: Várias imagens de pratos para o *slideshow*.

O projeto também contém um objeto de texto que tem de ir buscar a informação a um *RSS Feed*. As configurações do *URL* e do *Path* podem ser encontradas no ficheiro "feed.txt".

5.4 Etapas para a realização do teste

Para completar o teste, o utilizador necessita de realizar as seguintes etapas:

1. Criar um novo projeto;
2. Colocar o projeto com a resolução: 1280*720;
3. Adicionar a imagem "fundo" e colocá-la a ocupar a totalidade da área de trabalho;
4. Adicionar a imagem "Quadro com o menu" e posicioná-la de acordo com o exemplo dado;
5. Adicionar um objeto imagem com as imagens de todos os pratos e posicioná-lo de acordo com o exemplo dado;
6. Adicionar um objeto relógio e configurá-lo de acordo com o exemplo;
7. Adicionar um objeto de texto e configurá-lo de acordo com o exemplo. A fonte do texto deve ser um *feed*, a informação para preencher os campos de *feed* encontra-se no ficheiro "feed.txt" na pasta de recursos.

A etapa 1 tem de ser a primeira a ser realizada pois é necessário ter um projeto criado para se poder trabalhar. É recomendado que o utilizador realize a etapa 2 em segundo lugar pois esta vai afetar o tamanho da área de trabalho e, como tal, a posição dos objetos. As etapas 3 à 8 podem ser realizadas em qualquer ordem.

5.5 Ambiente do teste

O teste de usabilidade deve ser realizado numa sala isolada que contenha uma câmara de filmar, uma secretária com um computador, um monitor a transmitir o resultado que se pretende que o participante atinja, uma cadeira para o participante, uma cadeira para o monitor do teste e uma folha guia. Na Figura 5.2 podemos ver a planta da sala e a posição de todos os elementos.

Como o resultado pretendido é um conteúdo com animações e efeitos visuais, não é possível representá-lo com uma imagem estática, para tal é necessário um monitor a transmitir o resultado. Deve existir uma câmara de vídeo a apontar para o monitor de forma a gravar todas as ações do utilizador.

O computador deve ter um *browser* a correr o Editor, e uma pasta na área de trabalho com todos os recursos necessários para a realização do teste. O monitor do teste deve estar situado a uma distância de 50cm no ângulo de 45 graus [29]. É importante que o monitor esteja perto do participante mas não tão perto que o distraia. Deve estar no raio de visão do participante para que este sinta que o monitor está lá. Como vai ser a primeira vez que o utilizador utiliza o Editor, é importante ter este grau de proximidade.

A folha guia contém a localização da pasta de recursos em discos, as várias etapas necessárias para a realização do teste e as instruções para sua realização.

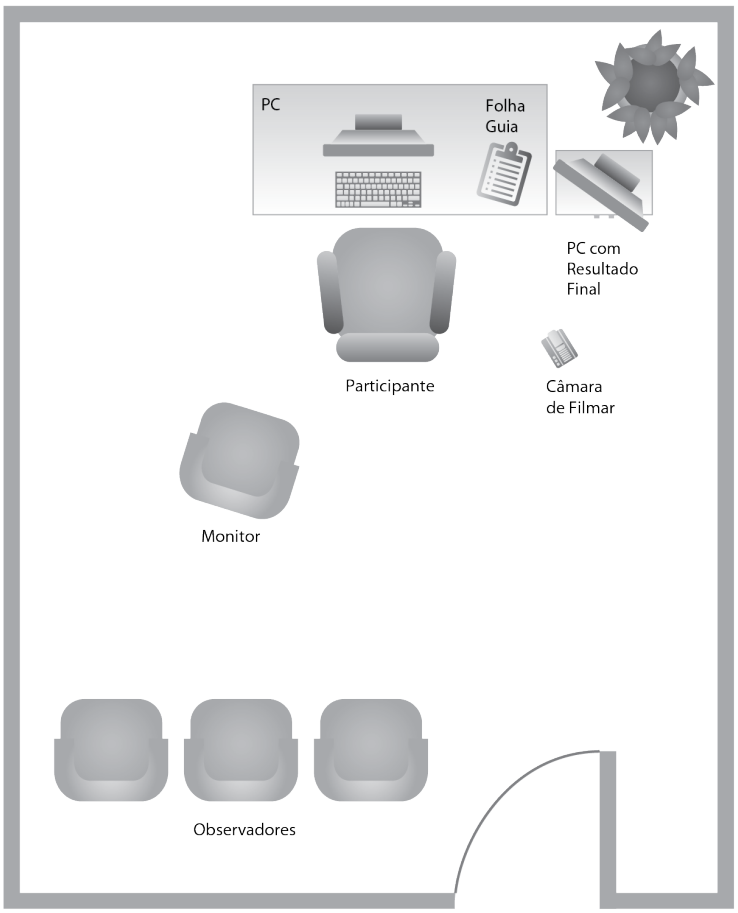

Figura 5.2: Planta da sala de testes de usabilidade

5.6 Inquérito

O inquérito é composto por uma primeira parte, com o intuito de identificar o candidato, uma segunda parte para avaliar a aplicação desenvolvida e uma terceira parte para identificar novas funcionalidades.

O inquérito é anónimo para que os candidatos não se sintam influenciados nas respostas fornecidas.

5.6.1 Identificação do candidato

Para ser possível situar o candidato na amostra, é necessário que forneça algumas informações pessoais. Não é pedido ao candidato o Nome e o Email para garantir a sua confidencialidade.

O candidato deve fornecer a seguinte informação pessoal: idade, género, nacionalidade, profissão, localidade onde trabalha, nível de experiência a trabalhar com editores de objetos (escala de 1 a 5) e nome dos editores com os quais trabalhou.

5.6.2 Avaliação da aplicação

Para avaliar a aplicação são utilizados os testes IsoMetrics2 [11]. Estes testes são baseados no padrão ISO 9241 Part 10 e são compostos por 7 categorias. Em cada questão o utilizador, para além de avaliar a aplicação, ainda pode dar a sua opinião. As 7 categorias são:

- Adequação para a tarefa (A)- esta secção permite avaliar se a interface é adequada para a tarefa a desenvolver;
- Autodescritividade (B)- esta secção permite avaliar se a interface é auto-descritiva, e se o utilizador consegue facilmente perceber todas as componentes da mesma;
- Controlabilidade (C)- esta secção permite avaliar se o utilizador consegue adaptar a interface para facilitar o uso da aplicação;
- Conformidade com as expectativas do utilizador (D)- esta secção permite avaliar se a aplicação está de acordo com as expetativas do utilizador;
- Tolerância a erros (E)- esta secção permite avaliar se a aplicação tem tolerância a erros e se o utilizador consegue desfazer uma ação facilmente;
- Adequação para a personalização (F)- esta secção permite avaliar se a aplicação pode ser personalizada para as tarefas a realizar;
- Adequação para a aprendizagem (G)- esta secção permite avaliar se os utilizadores conseguem facilmente aprender a utilizar a aplicação e se todas as ações são consistentes.

Apesar do teste ter todas estas secções, as secções C e F não serão utilizadas na avaliação desta aplicação, pois não foi desenvolvida para considerar estes pontos.

5.6.3 Novas funcionalidades

No fim do questionário, após o candidato ter respondido a todas as questões sobre a aplicação, é-lhe pedido que sugira uma lista de novos objetos que ache pertinente incluir numa aplicação e uma lista de funcionalidades que, na sua opinião, deveriam ser adicionadas para melhorar a aplicação.

Capítulo 6

Conclusões e trabalho futuro

Ao longo desta dissertação, foram apresentadas as motivações e os objetivos que guiaram o desenvolvimento deste trabalho. Descreveram-se conceitos, tecnologias e ferramentas utilizados para o seu desenvolvimento, bem como os aspetos a ter em conta no desenho de um editor, testes de usabilidade, trabalhos relacionados e outros editores bastante conhecidos. Foi proposta uma solução para a composição, interpretação e apresentação de ecrãs de informação aplicados a ambientes de *digital signage*. Para além do estudo do estado da arte, e da definição conceptual dos requisitos e arquitetura da plataforma, este foi desenvolvido e implementado na plataforma FutureboxTv de onde se salientam as seguintes contribuições:

- Foi criada uma linguagem declarativa que permite guardar a informação dos projetos bem como transmitir a informação dos projetos entre as aplicações desenvolvidas. Esta é definida por uma estrutura de dados predefinida e contém toda a informação do projeto e dos objetos do projeto;
- Foi desenvolvido um Serviço Web para tornar possível a inserção de recursos numa área de projeto remota, de modo a limitar o acesso dos utilizadores e dos projetos aos próprios recursos;
- Foi desenvolvido um Editor para permitir a criação e edição de conteúdos, sendo ele uma aplicação WYSIWYG, permitindo aos utilizadores terem sempre uma noção da aparência final do seu trabalho;
- Foi desenvolvido um Visualizador para executar os projetos desenvolvidos, e que foi integrado no Editor para permitir executar os conteúdos durante o seu desenvolvimento. Também foi integrado na FutureboxTv, que é uma solução de apresentação de conteúdos *digital signage*;

- Foi desenvolvida uma barra de propriedades e um conjunto de propriedades para permitir personalizar os conteúdos dos projetos de uma forma simples, fácil e elegante;

- Foi criado um conjunto de objetos com comportamentos personalizáveis que permitem o desenvolvimento dos conteúdos. Cada conteúdo pode conter um ou mais objetos e cada objeto tem um conjunto de propriedades que permitem a sua personalização;

- Foram planeados testes de usabilidade para avaliar a aplicação desenvolvida e descobrir melhorias.

6.1 Conclusões

Nesta dissertação estudou-se e concretizou-se uma solução para a composição, interpretação e apresentação de ecrãs de informação.

Após a fase mais teórica do estudo de vários editores e dos aspetos a ter em conta no desenho de um editor, detetou-se uma série de padrões e conceitos que os vários editores utilizam. Estes conceitos permitem que os utilizadores consigam identificar mais facilmente as regras de funcionamento dos editores e foram utilizados durante o desenvolvimento do *layout* do Editor.

As propriedades do projeto e dos vários objetos estavam sempre a ser atualizadas e por isso a linguagem declarativa de suporte sofreu alterações constantes durante o desenvolvimento do projeto, a sua estrutura foi pensada para que estas alterações pudessem ser feitas facilmente, mantendo a eficiência e a coerência da informação que necessitava de armazenar.

O Serviço Web foi a solução ideal para o controlo e para o armazenamento dos recursos dos projetos. Este também tornou possível a publicação dos projetos com o *BackOffice* da FutureboxTv, pois esta nunca poderia ser realizada diretamente através do Editor, por questões de segurança (tornava possível a interceção da comunicação com *BackOffice* da FutureboxTv).

O Editor passou por várias fases e aparências durante o seu desenvolvimento. Para facilitar estas transações, todos os componentes que o compõem foram desenvolvidos em separado e a comunicação entre eles foi limitada ao mínimo possível. O processo de passar a informação dos objetos a todos os componentes do Editor foi bastante facilitado pelo *Data Binding*. A criação de um objeto genérico responsável pela localização, tamanho, opacidade e rotação permitiu retirar a implementação destas propriedades dos vários *plugins*. Para que isso fosse possível, cada *plugin* é inserido no Editor dentro de um objeto genérico.

O Visualizador teve de ser suficientemente genérico para poder ser integrado no Editor e

na FutureboxTv. A sua integração no Editor foi muito importante, pois permite que os utilizadores vejam o resultado e comportamentos do seu trabalho quando for publicado.

A barra de propriedades é uma ótima forma de alterar as propriedades dos objetos. Por estar sempre visível permite que os utilizadores tenham sempre a noção das propriedades dos objetos. O desenvolvimento de um conjunto de propriedades suficientemente genéricas para serem reutilizadas em diversas situações facilitou bastante a adição de novas propriedades e a personalização das mesmas.

O uso de diferentes objetos para cada tipo de conteúdo que se pretende adicionar facilita o desenvolvimento dos mesmos e a correção de erros que cada um possa ter. Como os objetos têm comportamentos diferentes no Editor e no Visualizador, a criação de dois objetos do mesmo tipo, um para o Editor e outro para o Visualizador, diminui a complexidade dos objetos.

Os testes de usabilidade são fundamentais para a avaliação de uma aplicação. Apesar de não ter sido possível realizar testes de usabilidade no âmbito deste projeto, foi criado e descrito um teste de usabilidade que irá permitir a avaliação da aplicação e encontrar futuros aperfeiçoamentos.

6.2 Trabalho Futuro

Existem diversas áreas do trabalho desenvolvido suscetíveis de trabalho futuro, quer a nível da interação, quer a nível de funcionalidades.

- Em primeiro lugar, é necessário criar documentação detalhada com exemplos reais, para ajudar, apoiar e tirar dúvidas aos utilizadores, durante a realização do seu trabalho. É necessário pensar numa estrutura para o documento de ajuda e desenvolver uma forma de a integrar com o Editor.
- Realizar o teste de usabilidade criado para recolher a opinião de utilizadores e detetar problemas e futuras melhorias.
- Tornar a aplicação tolerante a erros do utilizador implementando um sistema que guarde todas as ações do utilizador e permita desfazer as últimas ações efetuadas. Este sistema irá permitir ao utilizador uma maior confiança no uso da aplicação e maior confiança na sua explorar.
- O sistema de *zoom* na área de trabalho do Editor só contém dois modos: o modo Tamanho Real e o modo Ajuste no Ecrã. Poderia ser desenvolvido um sistema de *zoom* mais complexo que permitisse ao utilizador especificar o nível de *zoom* pretendido.
- Para manter a coerência, foi desenvolvido o Visualizador em Silverlight, mas durante o desenvolvimento deste trabalho o fim do Silverlight foi anunciado e o HTML5 cres-

ceu. Como forma de estender a aplicação, mas ao mesmo tempo aproveitar o trabalho desenvolvido, poder-se-ia desenvolver um Visualizador em HTML5, ou noutra tecnologia, pois este só precisa de conseguir interpretar a linguagem declarativa criada para executar os projetos.

- Desenvolvimento de mais objetos como, por exemplo, um carrossel de vídeos. Aumentar as funcionalidades dos objetos desenvolvidos como, por exemplo, permitir que os objetos de áudio possam ter mais que um conteúdo.

- Introduzir lógica de animação para controlar a entrada e saída dos vários objetos do *template*, podendo este controle ser realizado através de inserção de uma barra temporal de eventos.

Bibliografia

[1] Adobe. Adobe flash platform, Consultado em Março 2012. http://www.adobe.com/flashplatform/?promoid=EENAL.

[2] C-nario, Consultado em Janeiro 2011. http://www.c-nario.com/.

[3] World Wide Web Consortium, Consultado em Janeiro 2011. http://www.w3.org/.

[4] World Wide Web Consortium. Extensible markup language (xml), Consultado em Janeiro 2011. http://www.w3.org/XML/.

[5] World Wide Web Consortium. Xml schema, Consultado em Janeiro 2011. http://www.w3.org/XML/Schema.

[6] HARRIS Creator, Consultado em Janeiro 2011. http://digitalsignage.harris.com/products/creator.htm.

[7] SCALA Designer, Consultado em Janeiro 2011. http://www.scala.com/digital-signage-products/designer.html.

[8] DMGNA, Consultado em Janeiro 2012. http://www.dmgna.com/online-video-marketing-and-optimization/.

[9] FutureboxTv, Consultado em Janeiro 2011. http://futurebox.tv/.

[10] GIMP, Consultado em Janeiro 2011. http://www.gimp.org/.

[11] K.-C. Hamborg. Isometrics, Consultado em Março 2012. http://www.isometrics.uni-osnabrueck.de/.

[12] HARRIS, Consultado em Janeiro 2011. http://harris.com/.

[13] Java, Consultado em Janeiro 2011. http://java.com/.

[14] C. Leeds, E. Kosinska, and M. Inc. *Microsoft Expression Blend 4 Step by Step*. Step by Step. Microsoft Press, 2011.

[15] D. McClelland. *Adobe Photoshop CS5 One-on-One*. O'Reilly Series. O'Reilly Media, 2010.

[16] Messenger, Consultado em Janeiro 2011. http://www.c-nario.com/category/Messenger.

[17] Microsoft. Data binding, Consultado em Janeiro 2011. http://msdn.microsoft.com/en-us/library/cc278072(v=vs.95).aspx.

[18] Microsoft. Expression blend, Consultado em Janeiro 2011. http://www.microsoft.com/expression/products/blend_overview.aspx.

[19] Microsoft. Microsoft silverlight, Consultado em Janeiro 2011. http://www.silverlight.net/.

[20] Microsoft. .net framework conceptual overview, Consultado em Janeiro 2011. http://msdn.microsoft.com/library/zw4w595w.aspx.

[21] Microsoft. Visual c#, Consultado em Janeiro 2011. http://msdn.microsoft.com/.

[22] Microsoft. Visual studio, Consultado em Janeiro 2011. http://msdn.microsoft.com/en-us/vstudio/aa718325.

[23] Microsoft. Xaml overview, Consultado em Janeiro 2011. http://msdn.microsoft.com/en-us/library/ms752059.aspx.

[24] Microsoft. Expression studio, Consultado em Novembro 2011. http://www.microsoft.com/expression/.

[25] Adobe photoshop, Consultado em Janeiro 2012. http://www.photoshop.com/.

[26] HARRIS Player, Consultado em Janeiro 2011. http://digitalsignage.harris.com/products/player.htm.

[27] SCALA Player, Consultado em Janeiro 2011. http://www.scala.com/digital-signage-products/player.

[28] PowerPoint, Consultado em Dezembro 2011. http://office.microsoft.com/en-us/powerpoint/.

[29] J. Rubin, D. Chisnell, and J. Spool. *Handbook of Usability Testing: Howto Plan, Design, and Conduct Effective Tests*. Wiley, 2008.

[30] SCALA, Consultado em Janeiro 2011. http://www.scala.com/.

[31] J. Tidwell. *Designing Interfaces*. O'Reilly Series. O'Reilly Media, 2005.

[32] J. Tidwell. *Designing Interfaces*. O'Reilly Series. O'Reilly Media, 2 edition, 2010.

[33] Viatecla, Consultado em Janeiro 2011. http://www.viatecla.com.

[34] W3C. Html5, Consultado em Janeiro 2012. http://www.w3.org/TR/html5/.