FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Estudo e aplicação de paradigmas multitoque no interface humano-computador de um sistema SCADA

Carlos André de Matos Esteves

Mestrado Integrado em Engenharia Informática e Computação

Orientador: Rui Rodrigues (Professor Auxiliar Convidado)

Fevereiro de 2012

Estudo e aplicação de paradigmas multitoque no interface humano-computador de um sistema SCADA

Carlos André de Matos Esteves

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: A. Augusto de Sousa (Professor Associado)

Vogal Externo: Maximino Bessa (Professor Auxiliar)

Orientador: Rui Rodrigues (Professor Auxiliar Convidado)

_____

17 de Fevereiro de 2012

Resumo

O objetivo fundamental do trabalho apresentado neste documento relaciona-se com a implementação de uma solução SCADA, com aplicação de tecnologia e paradigmas multitoque no contexto de automação. A tecnologia multitoque evoluiu imenso ao longo da última década, estabelecendo-se como um paradigma bastante utilizado nos mais diversos contextos.

O trabalho realizado passou pelo desenvolvimento de um protótipo demonstrável de um sistema SCADA, criando controlos de suporte à navegação e à exploração da interface. Para favorecer a interação e a experiência de utilização foram desenvolvidos controlos de interface e controlos embebidos nas próprias páginas. Estes controlos são um suporte importante na execução das tarefas, funcionando como um complemento à exploração e navegação. Houve uma preocupação ao longo do desenvolvimento em tornar a solução flexível e costumizável, aumentando o poder de configuração da plataforma e tornando o utilizador um elemento importante no processo de desenvolvimento.

Por fim foi efetuado um estudo de usabilidade com o intuito de verificar e validar a aplicação, confrontando-a com uma solução semelhante. Estes testes foram efetuados recorrendo a dois grupos de diferentes utilizadores-alvo, que efetuaram diversos testes exemplo de uma utilização frequente nas duas aplicações. Os resultados revelaram indícios de uma melhoria na eficiência de utilização, quanto ao número de interações totais utilizadas e também na percentagem destas que foram realizadas com sucesso. O tempo de execução e percentagem de sucesso de execução das tarefas foi semelhante.

O trabalho desenvolvido contribui para a área do multitoque e da automação com a descrição de um caso experimental e do resultado do mesmo. Com base nos resultados dos testes efetuados pode-se concluir que a aplicação destes paradigmas de interação na área da automação industrial é perfeitamente viável existindo melhoria na produtividade, eficiência e na experiência de utilização.

Abstract

The main goal of this project is related to the implementation of a SCADA solution, based on multi-touch technology and user interface paradigms, in the context of industrial automation. The multi-touch technology has been in permanent evolution in the last decade, settling has a paradigm used in different contexts.

The work involved developing a demonstrable prototype solution of a SCADA system, by creating user controls to support the navigation and exploration of the user interface. To improve the interaction and user experience were designed user interface controls and also embebbed controls. These ones are an important support to the task execution, complementing the exploration and the navigation of interface. There was a certain concern in making the solution flexible and customizable, increasing the user interference in the configuration of the platform, making him an important element in the development process.

Ultimately we developed a usability study with the objective to verify and validate the application, confronting it with a similar professional solution. These tests were realized using two different groups of target users which completed several example tasks of a normal user of both applications. The results revealed evidence of improvement in the use efficiency related to the number of total used interactions and the percentage of these that were realized with success. It should be noted that the execution time and success of the tasks in both solutions were similar.

The work developed contributes to the area of multi-touch and automation with the description of an experimental case. Based on the test results we can conclude that the application of this paradigm and technology in the area of industrial automation is perfectly viable, being noted an improvement in the productivity, efficiency and user experience.

Agradecimentos

Em primeiro lugar gostaria de agradecer ao Professor Doutor Rui Rodrigues pelo apoio, disponibilidade e pela orientação fornecida durante todo o trabalho. Agradeço também pelos seus comentários, sugestões e pelas críticas que tornaram esta tese de dissertação melhor do que poderia vir a ser.

Ao Engenheiro Cláudio Silva que me orientou na Efacec e que foi parte importante neste trabalho. Agradeço a disponibilidade e o auxílio fornecido durante todo o projeto.

Ao Engenheiro Rogério Paulo pelas inúmeras sugestões no intuito de melhorar o trabalho desenvolvido.

Aos meus colegas de equipa de trabalho, o Engenheiro Pedro Portilha, o Engenheiro Pedro Franco, a Engenheira Ana Pinto, a Engenheira Elisabete Ferreira, o Engenheiro Nuno Milhases, o Engenheiro Jorge Silva e aos restantes colegas de trabalho pela ajuda no desenvolvimento desta dissertação e por me possibilitarem uma fácil integração na equipa.

Aos meus amigos pelo constante apoio e pela força que me deram no realizar desta dissertação.

À Filipa pela paciência e carinho dado ao longo de todos estes anos e acima de tudo pelo incondicional apoio que sempre demonstrou.

Aos meus pais, irmão e avó pelo constante interesse, preocupação e motivação que me deram para terminar os meus estudos. Agradeço muito o apoio que me deram, especialmente nesta última fase de trabalho, e pelo facto de terem sempre acreditado pacientemente nas minhas capacidades. Sem o seu apoio não me teria sido possível terminar um curso superior.

Por último, mas em primeiro plano, ao meu avô que sempre se interessou e preocupou com o meu desempenho académico, sabendo que o facto de eu atingir este marco o vai deixar eternamente feliz.

Carlos André de Matos Esteves

Índice

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

API	Application Programming Interface
AS	Automation Studio
HMI	Human-Machine Interface
IPC	Interação pessoa-computador
SCADA	System Control And Data Analysis
SDK	Software Development Kit
WIMP	Windows Icons Menus Pointers
WYSIWYG	What You See Is What You Get

Capítulo 1

1. Introdução

1.1 Contexto e Enquadramento

Um dos grandes desafios que a ciência da computação tem, no presente, é o de aprimorar a interação do utilizador com computadores ou sistemas tecnológicos. O estudo, nesta área nos últimos vinte anos, desenvolveu-se em geral com base numa interação baseada em dois dispositivos, o rato e o teclado. Registaram-se avanços muito importantes na rapidez de computação, na forma como passámos a trocar informação e a comunicar com os outros, mas a forma como interagimos não acompanhou essa evolução. Apesar de a partir de 1980 o número de investigadores na área de Interação Pessoa-Computador (IPC) ter aumentado consideravelmente, com o consequente aumento da publicação de artigos e investigações (Shackel 2009)(Shackel 2009), as empresas de soluções tecnológicas não acompanharam esse mesmo fluxo de inovação. A interação sofreu alterações mínimas e mesmo o tipo de interface WIMP (*Windows*, *Icons*, *Menus*, *Pointers*) desenvolvido na altura, mantém-se, ainda na atualidade, sendo já considerado como *standard*. Este paradigma serviu para permitir uma utilização estável e fácil da computação mas é incerto que sirva para acompanhar a miríade de formas, fatores e uso dos computadores acuais e futuros. Constata-se que a sua evolução tornou-os mais pequenos e ubíquos e a sua utilização está em franco crescimento, tornando-os mais presentes no nosso dia-a-dia (Turk 2000)(Turk… 2000).

As superfícies tácteis surgiram para proporcionar uma interação mais natural com o ser humano. Inicialmente uni-toque, passaram a suportar toques múltiplos – multitoque - e manipulação de vários graus de liberdade (GDL), potenciando um vasto leque de possibilidades à disposição dos *designers* de sistemas tecnológicos. Constata-se que é cada vez mais frequente as empresas recorrerem a esta tecnologia nos seus produtos. A Apple com o desenvolvimento do iPhone e do respetivo sistema operativo (iOS) alterou o panorama e motivou a aposta dos fabricantes nesta tecnologia. Atualmente é normal a sua presença nos telemóveis, bem como na

forma de publicitar e promover a interação dos clientes com o produto (painéis publicitários, quiosques de informação). Os próprios sistemas operativos, para computadores pessoais, aproveitaram esta oportunidade para promoverem o suporte dos seus *softwares* à tecnologia multitoque (Mac OS X, Windows 7 e em Linux). A nível empresarial esta mudança tem sido lenta. O tecido económico tem sido reticente quanto à integração das soluções multitoque nos seus produtos, dado que necessita de soluções sólidas e infalíveis. Com o multitoque a atingir finalmente uma fase de maturidade assiste-se a uma maior procura na nova abordagem tecnológica, com o intuito de obter uma interface mais eficaz, com um consequente aumento de produtividade.

O *multitoque* é um paradigma muito poderoso que, quando bem integrado, poderá permitir um crescimento da produtividade do ser humano, seja a nível laboral, seja enquanto utilizador em momento de lazer. O fator crítico encontra-se na interligação entre o aspeto tecnológico e a teoria existente da área de IPC. É necessário extrair todas as virtudes e vantagens de um novo tipo de interação, sem contudo prejudicar o rendimento do trabalhador.

1.2 Projeto

Este documento científico tem origem num projeto de aplicação de paradigmas e tecnologia multitoque num contexto profissional, desenvolvido na empresa Efacec. Este projeto pretende criar um protótipo funcional demonstrável que possa expandir o módulo de uma ferramenta já existente com interação baseada em rato e teclado, podendo servir de base para uma nova aplicação. Acima de tudo, a solução desenvolvida tem que preencher determinados requisitos de utilização e oferecer uma solução adequada ao contexto de utilização e ao tipo de utilizador.

Pretende-se no final avaliar a aplicação num contexto prático de utilização e estudar as possíveis vantagens e desvantagens em relação à solução já existente. Pretende-se avaliar se o multitoque pode ser uma mais-valia no contexto da automação.

1.3 Motivação e Objetivos

A empresa Efacec possui uma plataforma de automação para sistemas energéticos e de controlo cuja interação se limita ao rato e teclado. Esta plataforma, Automation Studio (Figura 1), serve para desenhar e projetar sistemas. Um componente desta aplicação é o editor de *mimics* (objetos que simulam entidades físicas) que permite uma visualização do tipo WYSIWYG de componentes e símbolos da própria plataforma.

1. Introdução

Com este trabalho pretende-se efetuar um estudo sobre as diferentes soluções possíveis de aplicar num sistema empresarial, de forma a integrar a tecnologia multitoque, com reconhecimento de gestos. Estas soluções serão abordadas sob duas áreas importantes: sob o ponto de vista da matéria de IPC (usabilidade, gestos, menus e controlos) e sob o ponto de vista das soluções de *software* multitoque existentes no mercado (plataformas de desenvolvimento e

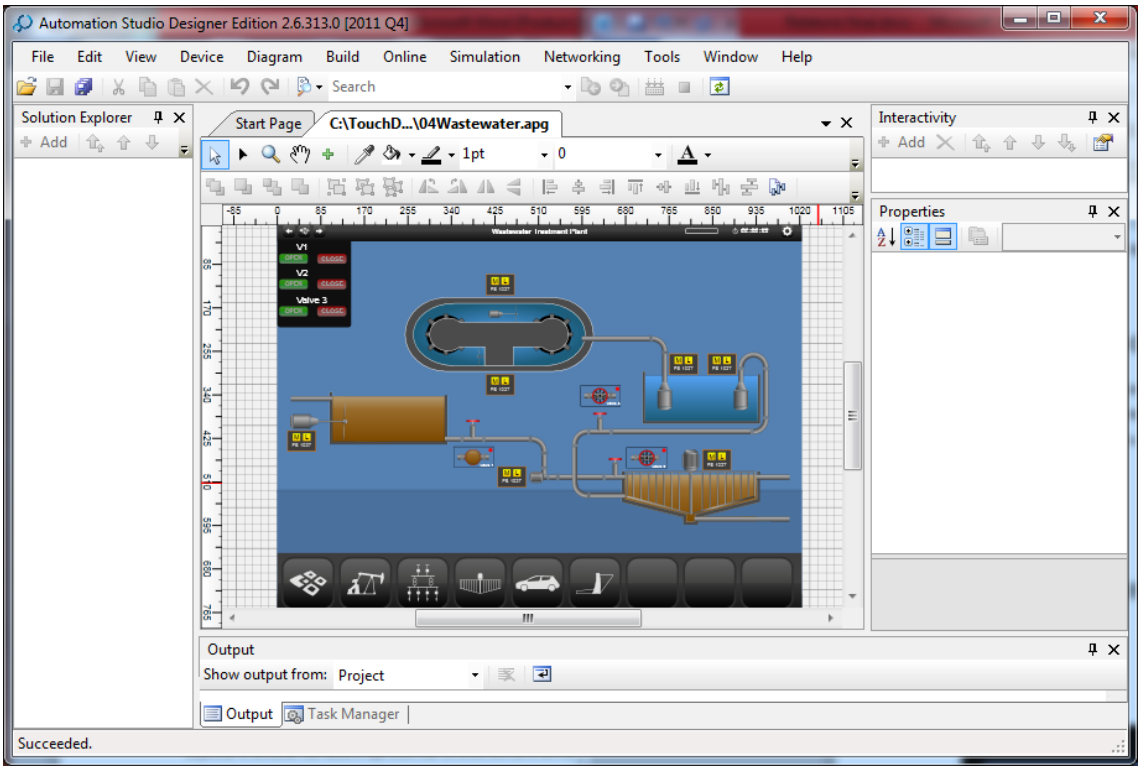


Figura 1 - Automation Studio

soluções semelhantes).

A proposta a desenvolver pretende que sejam integradas tecnologias *multitoque*, de modo a criar uma alteração na forma de interação com esta plataforma. Neste contexto, pretende-se que sejam estudados novos tipos de interação, sendo que no final será implementado um protótipo demonstrável, que reconheça gestos multitoque em superfícies tácteis. A empresa pretende, ainda, que a plataforma de automação seja dotada de bibliotecas que permitam a utilização de interação multitoque noutros projetos da empresa.

A tecnologia multitoque encontra-se em constante evolução, fazendo parte da área de IPC, havendo diversas vertentes desta área que são fulcrais no seu desenvolvimento (design e paradigmas de interação, metáforas de interface, etc.). É um tema que suscita bastante interesse para quem é investigador na área, mas também para quem utiliza os conhecimentos disponibilizados e os aplica em projetos reais.

É um desafio interessante e motivador poder aplicar novos paradigmas de interação em ferramentas utilizadas a nível empresarial, ainda que inicialmente numa fase de prototipagem.

1.4 Estrutura da Dissertação

A dissertação foi organizada de forma a seguir uma sequência lógica de investigação, a fim de proporcionar uma visão mais clara e eficiente para o leitor.

O relatório inicia-se com a apresentação do âmbito do projeto de dissertação, sendo seguido da introdução do mesmo. No capítulo 1 é efetuado um enquadramento a nível tecnológico do documento, fazendo também um resumo dos temas abordados. De seguida, são delineados os objetivos e a motivação do projeto, realçando o que é proposto ser alcançado no decurso do mesmo.

No segundo capítulo é feita uma revisão bibliográfica da literatura e dos temas diretamente relacionados com a área de interação pessoa-computador (design de interação, metáforas de interface e paradigmas de interação), seguidos do levantamento das propostas de autores na área do multitoque. Com este capítulo pretende-se sintetizar as principais abordagens e formas de interação possíveis e adequáveis a este contexto.

O assunto subsequente é o da abordagem tecnológica referente ao software a utilizar. É feito um estudo das *frameworks* multitoque existentes, enumerando as características, vantagens e desvantagens de cada uma. Para além disso, são listadas as ferramentas de software que serão utilizadas na implementação do projeto.

Por último, é apresentado o planeamento e descritas as várias fases no qual o projeto se divide e respetivas fases de início e término.

Capítulo 2

2. Revisão Bibliográfica

2.1 Introdução

O domínio de interação entre e o utilizador e computador tem registado um forte crescimento de investigação nos últimos anos, seja pela vulgarização de novas tecnologias multitoque, seja pela necessidade de se proporcionar uma experiência mais natural e intuitiva.

O estudo da interação multitoque aborda duas áreas fulcrais: a de *design* (orientada para o utilizador) e a de investigação tecnológica (*tabletop, surface, multitoque*). No que respeita ao primeiro ponto, o desenvolvimento do *design* de interação tem um papel fundamental para a melhoria da usabilidade tendo em conta o paradigma de multitoque. Este paradigma necessita que as capacidades do utilizador sejam estudadas ao pormenor, seja em termos de ergonomia ou mesmo na carga cognitiva exigida na utilização. Ao alterar a quantidade de *inputs,* aumentam-se os GDL, o que por si também aumenta a complexidade de interação. Assim sendo, é necessário que o utilizador entenda o que pode fazer sem necessidade de ler extensos manuais. Para esse efeito é necessário recorrer a estratégias específicas de *design* ou mesmo de exploração de metáforas de interface que facilitem a compreensão, através, por exemplo, do paralelismo com a realidade.

O estilo de interação WIMP (Lafon 2000), desenvolvido nos anos 80 e ainda utilizado atualmente, foi originalmente planeado para utilização com teclado e rato permitindo uma seleção de objetos mais precisa. Este facto não se verifica quando se passam a utilizar os dedos para selecionar e manipular conteúdos. Esta transição na interação exige, em paralelo, o desenvolvimento e investigação de técnicas de interação adequadas. O uso de gestos mais complexos com um ou mais dedos preenche uma lacuna mas não é suficiente. Os objetos a que estávamos habituados (menus, botões, janelas) têm que ser reformulados ou substituídos para permitir uma interação adequada ao meio utilizado. Neste caso, existe uma necessidade clara de

adaptação da tecnologia ao ser humano, contrariando uma tendência inicial focada na tecnologia a desenvolver.

Neste capítulo exploramos estas duas áreas com indicação das alternativas em cada caso e uma correspondente análise crítica. Na primeira fase é feito um estudo da área de IPC com foco no *design* e nos paradigmas de interação e na segunda fase são revistas propostas alternativas de interação em termos de interface.

2.2 Interação pessoa-computador

Um sistema informático compreende duas componentes essenciais, a interface com o utilizador e a lógica funcional do mesmo. A interface com o utilizador é bastante complexa de desenvolver e manter, dado que agrega vários aspetos dos sistemas interativos (Goldin and Smolka 2006). Para auxiliar esse processo surgiu uma área de investigação focada no utilizador e no estudo das suas características psicológicas e cognitivas, a IPC. Esta área é bastante abrangente e multidisciplinar englobando conceitos de psicologia, ciências sociais e ciência da computação. O seu objetivo é o de ajudar os profissionais da área da computação a melhor perceber o utilizador e a melhorar os seus produtos, de forma a que a sua utilização se torne mais natural e intuitiva.

A evolução dos sistemas informáticos tornou o utilizador parte integrante do sistema. Esta transição deveu-se à necessidade dos designers de sistemas integrarem o processo de interação, deixando apenas de se preocupar com algoritmos e linguagens. O paradigma de manipulação direta (*direct manipulation*) acabou por se tornar num fator que alterou por completo o funcionamento da área de IPC. Este conceito consiste num paradigma em que o utilizador manipula diretamente os objetos que se lhe apresentam, através de ações similares ao que utiliza no mundo real. Por exemplo, no mundo real um utilizador quando pretende efetuar uma operação sobre um determinado objeto primeiro pega nele e de seguida efetua o que pretende. O paradigma de *direct manipulation* traça um paralelismo com este comportamento permitindo que um utilizador selecione um objeto virtual e aplique uma determinada ação sobre este. Schneiderman (Shneiderman 1981) explica que a manipulação direta é caracterizada por apresentar uma representação contínua do objeto de interesse e por permitir uma fácil reversão de ações e comandos. Estas propriedades foram complementadas com o conceito de *directness* – sentimento de imediato que o utilizador tem quando lida com uma interface (Hutchins, Hollan et al. 1985). Surgiu devido à necessidade de analisar o ambiente que rodeia o utilizador para tentar perceber o que o leva a sentir-se mais ambientado e mais envolvido com a tecnologia (Hutchins, Hollan et al. 1985). Passou a ser necessário analisar com mais acuidade a ação (a nível biomecânico mas também a nível dos periféricos utilizados) e, acima de tudo, potenciar a forma como a informação mostrada pelos computadores passou a ser transmitida ao utilizador.

Esta melhoria no processo comunicacional entre máquina e utilizador tornou-se num desígnio importante para aumentar a produtividade do uso da tecnologia.

2.2.1 Paradigmas de interação

A investigação na área da computação na última década, tem-se caracterizado por uma constante evolução e descoberta de novas formas de interagir e comunicar. Esta necessidade tem levado a que sejam criados novos paradigmas de interação, que otimizem a comunicação existente entre o utilizador e o computador.

Um desses paradigmas é o de computação ubíqua (*ubiquitous computing*). Foi introduzido por Mark Weiser, que lançou a discussão do desaparecimento da computação (Weiser 1999). A ideia fulcral por detrás deste conceito é o de que a tecnologia evoluiu ao ponto de que é possível a sua introdução em qualquer lado, em qualquer objeto, em qualquer lugar. A computação passa a estar presente nos objetos do nosso dia-a-dia, misturando-se com a realidade e tornando-se invisível. *"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it"* (Weiser 1999). Weiser considerava que esta nova forma de computação, ao contrário de tecnologias como a realidade virtual que criavam um novo mundo de interação entre representações humanas, a computação ubíqua apenas residia no mundo humano não havendo restrições nem barreiras à interação (Weiser 1999). Esta forma de computação torna necessária uma nova abordagem da tecnologia em relação às nossas vidas, de forma a não se tornar um obstáculo (Weiser and Brown 1996). Esta visão levantou algumas vozes discordantes. O facto de a tecnologia se tornar invisível deixa-nos num vazio. Se não a vemos, como é que a podemos controlar (Goldin and Smolka 2006)? Se não a podemos controlar nem visualizar perdemos a capacidade te tomar decisões, ficando dependentes dos computadores no nosso dia-a-dia (Rogers 2006). É portanto natural que seja necessário analisar e estudar os aspetos individuais, sociais, culturais e físicos da experiência humana, relacionados com o espaço em que este habita (Ciolfi 2004).

Um paradigma que se começa a tornar comum é o das superfícies multi-tácteis. Este surgiu como forma de tornar a interação com o utilizador mais natural fazendo com que exista menor necessidade de utilização de outros dispositivos (rato e teclado). Com o desenvolvimento de produtos como o iPhone, a tecnologia chegou às massas passando a ser considerada para situações de lazer, trabalho corporativo e interação social. Neste trabalho pretende-se abordar este paradigma numa perspetiva empresarial, utilizando tecnologia que suporte interação multitoque.

Outros paradigmas foram sendo criados e explorados, tentando interligar objetos físicos com tecnologia. Entre esses casos podemos encontrar o de computação tangível (Ullmer 1997) e Fitzmaurice and Ishii 1995), o *wearable computing* (Mistry and Maes 2009) ou mesmo os que utilizam gestos e voz para comunicar com um sistema computacional (Tse, Shen et al. 2006;

Hilliges, Izadi et al. 2009). Estes paradigmas recorrem, por vezes, a sensores e câmaras para recolha e tratamento de informação.

2.2.2 Design de interação

> *"Interaction design is the art of instigating and guiding behaviors (or interaction dynamics) by means of proper static or dynamic stimuli" (Valli 2008)(Valli 2008)*

A área de IPC é considerada por muitos autores como uma área orientada ao design, envolvendo tanto os investigadores da área como os responsáveis pelo design industrial (Fallman 2003).

O nascimento dos sistemas informáticos com interface gráfica levou à expansão da área de *design* de interface. Tornou-se fulcral no desenvolvimento de computadores que fossem acessíveis e usáveis por pessoas comuns sem necessidade de conhecimentos técnicos específicos (Sharp, Rogers et al. 2007). Os engenheiros e cientistas sentiram necessidade de estudar e melhorar a interação desenvolvendo conceitos, ideias e objetivos de *design*. O poder de decisão passou para o utilizador permitindo-lhe manipular objetos de interesse, tornando a interação mais natural (Shneiderman 1981). O computador deixou de ser apenas um agente que recebe parâmetros e emite resultados consoante os dados de entrada, para se transformar numa máquina que interage com o mundo real. O utilizador passou de um papel de observador para o de operador, tornando-se parte cativa dessa mesma aplicação (Dina Goldin 2006).

Contudo, deu origem a novas questões relacionadas com as representações gráficas, pois obrigou a um esforço adicional por parte dos utilizadores que foram forçados a entendê-las, aumentando a carga cognitiva. Os próprios *designers* e programadores passaram a ter um cuidado acrescido, com a colocação de elementos no écran (quer ao nível da posição quer da quantidade), com o tipo de interação necessária, etc. Em resumo, foi necessário planear a forma de comunicar entre um ser humano e uma máquina.

Processo de design

A introdução da manipulação direta mudou a forma de interagir com o computador criando um vazio na área do *design*, que levou ao nascimento do design de interação (*interaction design*). *"In a nutshell, interaction design is related to software engineering in the same way as architecture is related to civil engineering."* (Sharp, Rogers et al. 2007).

Esta forma de desenhar e projetar aplicações de interação não pode ser dissociada do conceito de usabilidade. Shackel (Shackel 2009) considera atualmente que usabilidade é, de forma reduzida, a capacidade de algo ser utilizado pelo ser humano com facilidade e eficácia. Acima de tudo deve ser usada como uma métrica para definir o quanto um produto pode ser

usado por vários utilizadores (que representem grupos específicos) cumprindo objetivos tais como: a eficiência, a eficácia ou mesmo a satisfação.

O paradigma da manipulação direta levou a que o próprio conceito de usabilidade evoluísse. Passou a ser considerado fulcral no desenvolvimento de sistemas e foram propostos princípios para auxiliar os responsáveis pela criação de sistemas informáticos. Gould e Lewis sugerem três ideias fundamentais: que seja dada atenção desde o início do desenvolvimento ao utilizador e às tarefas expectáveis de serem executadas, que os protótipos e os testes de simulação sejam executados ao longo do desenvolvimento, havendo medições de performance e, finalmente, que o processo de *design* seja iterativo (Gould 1985).

O processo de *design* de interação evoluiu com estas ideias, dando particular ênfase à intervenção do utilizador e à prototipagem e simulação. Este processo divide-se em 4 fases que permitem esquematizar e orientar o processo de desenho:

1. Identificar as necessidades do projeto e definir requisitos;
2. Desenvolver *designs* que suportem os requisitos previamente definidos. (usualmente são desenvolvidos vários designs diferentes de forma a aumentar o leque de escolhas a validar);
3. Construção de versões interativas dos *designs* de forma a poder ser visualizado e avaliado o que foi planeado - forma de comunicar com os responsáveis pelo desenvolvimento e de lhes mostrar o que foi projetado, podendo ou não recorrer utilizadores-alvo;
4. Avaliação do que foi desenvolvido durante o processo

Pretende-se que sejam alcançados objetivos tanto de usabilidade (facilidade de uso, de compreensão, eficiência), bem como da experiência do utilizador (satisfação, entretenimento, felicidade na utilização) (Sharp, Rogers et al. 2007).

No processo de prototipagem é usual haver uma fase inicial de esboço (*sketching*), no qual o *designer*, exterioriza a sua forma de pensar (Fallman 2003). Ele exterioriza as suas ideias para a interface através da criação de protótipos de baixo nível, sem grandes preocupações e com pouca precisão. Esta incerteza encoraja o *designer* a explorar novas ideias sem se sentir pressionado a ter que decidir relativamente aos pormenores (cores, alinhamento, etc.), preservando portanto a fluidez de processo (Landay 2001).

É essencial que estas fases sejam desenvolvidas com permanente validação e acompanhamento do utilizador, por forma a que seja alcançada uma elevada fiabilidade e consistência com o que foi projetado. O processo de desenvolvimento com utilizadores-alvo foi inicialmente referido por Gould e Lewis tendo sido apelidado de *design* interativo (*interactive design*) (Gould 1985). Esta metodologia consiste no acompanhamento e participação dos utilizadores em permanência, no processo de *design* através da validação de protótipos e simulação. Este conceito foi evoluindo ao longo do tempo, tendo surgido diferentes modelos de

desenvolvimento tais como o *user-centered design* e *participatory design* (centrados no utilizador) (Carroll 1997) e o *learn-centered design* (especificação das necessidades do utilizador e do seu meio de aprendizagem) (Soloway and Guzdial 1994). Estes modelos caracterizam-se por envolverem os utilizadores no estabelecimento de objetivos e planeamento de protótipos, em vez de apenas figurarem no processo inicial de prototipagem.

Devido à complexidade de aprendizagem por parte do utilizador têm sido desenvolvidos diversos estudos a fim de atenuar esse obstáculo. Um desses estudos, explica que a melhor forma de simplificar o processo, consiste em dividi-lo em pequenas tarefas, começando pelas mais básicas. Esta abordagem permite que o processo de aprendizagem seja mais eficiente e a carga cognitiva seja muito menor (Van Merriënboer, Kirschner et al. 2003). Com o mesmo objetivo surgiu o conceito de interface natural para o utilizador (*natural user interface*), baseando-se no facto de a interação do utilizador com o sistema, encontrar um paralelo na relação com a natureza. A natureza tem um sentido de continuidade em que as ações não se desenvolvem por passos facilmente separáveis, *"Nature does not progress by leaps"* (Valli 2008). Como tal, o processo de aprendizagem deve ser contínuo e com a complexidade a aumentar à medida que o conhecimento se torna consistente.

Foi necessário alargar os paradigmas do *design*, movendo o foco para a experiência dos sítios e espaços no qual se procede a interação. A análise dos aspetos estruturais das funcionalidades pode dar uma contribuição importante no desenvolvimento de sistemas ubíquos. Não se trata apenas de pensar no espaço físico, mas também nas atividades e experiências que o utilizador presencia, mediadas pela tecnologia (Ciolfi 2004).

Princípios de design

Os princípios do *design* de interação assentam em dois conceitos fundamentais: por um lado, caracterizar e analisar o utilizador, e por outro, perceber onde é que o produto vai ser utilizado. Este estudo leva a que o desenvolvimento seja apropriado e personalizado (a nível global) para um certo grupo de utilizadores. Uma abordagem correta, para o sucesso consiste em analisar o que é processado a nível humano, quando se dá a interação com o produto em causa (Sharp, Rogers et al. 2007). No processo de *design* de interfaces multitoque, existem diversos desafios que carecem de resolução, seja do ponto de vista do utilizador, seja da tecnologia. Estes desafios centram-se ao nível da interação com o écran, ao nível da interação com o utilizador (a nível ergonómico e de diferenças entre grupos de utilizadores) e, finalmente, com os resultantes das várias formas possíveis de interação (gestos, suporte a multiutilizador, entrada de dados) (Bachl, Tomitsch et al. 2010).

Uma aplicação interativa baseia-se na troca e manipulação de informação. Mas este processo, para ser efetuado com sucesso, exige a necessidade de constante *feedback* do sistema com o utilizador. A rapidez do *feedback* na alteração do comportamento de objetos que estão a ser manipulados, ajuda bastante o utilizador a nível cognitivo, dado que uma representação contínua do estado do sistema, remove a perceção do computador como um intermediário,

facilitando a naturalidade da interação (Hutchins, Hollan et al. 1985). Um dos principais problemas relacionados com o *feedback* resulta da ambiguidade no toque em superfícies multitoque. Quando o utilizador tem a intenção de realizar uma ação e esta resulta num comportamento inesperado, acaba por haver um aumento da confusão e frustração do utilizador (Wigdor, Williams et al. 2009).

Para além do *feedback* é necessário que exista uma noção de continuidade e controlo da aplicação. Esta noção pode ser conseguida através da criação de ações que devem ser reversíveis (na medida do possível), sendo que as que são impossíveis de reverter e que possam causar danos de desempenho, devem ser dificultadas ao máximo (Norman 1983)(Norman 1983).

Para que a interação se desenvolva com toda a naturalidade possível, é fulcral que todo o *design* seja guiado e acompanhado por métricas de usabilidade. A usabilidade é geralmente aceite como garante de que os produtos interativos são fáceis e agradáveis de utilizar por parte do utilizador (Sharp, Rogers et al. 2007). O estudo da usabilidade envolve a definição de como a medir, a definição de objetivos a alcançar, e a metodologia utilizada para os atingir. Estes princípios tornam a usabilidade numa parte integrante no sistema em vez de a utilizarem como remendo final num projeto (Wixon 1985).

Por forma a manter a usabilidade, é fulcral que a relação entre o utilizador e o objeto a utilizar, seja facilmente percetível e automática. A este conceito dá-se o nome de *affordance*. Consiste na relação entre o objeto no mundo real e as intenções, perceções, e as capacidades da pessoa, possibilitando que as ações sejam automaticamente percetíveis e intuitivas (Weiser and Brown 1996). O *design* de interfaces multitoque baseia-se na *affordance* do écran por parte do utilizador. É necessário que o utilizador entenda que, sem a utilização de um meio de interação a que esteja mais habituado (tal como o rato ou teclado), o próprio *écran* que se encontra à sua frente é o meio de interação. Se o *designer* não conseguir influenciar a disposição do hardware, necessita de recorrer ao uso de instruções simbólicas ou escritas para fornecer pistas visuais (Bachl, Tomitsch et al. 2010).

Outros princípios referem-se a aspetos também importantes no *design* de interfaces e interação. É necessário que o *design* respeite normas de consistência, pelo que as ações e a interação devem ser planeadas, utilizando elementos e lógicas de execução similares (Sharp, Rogers et al. 2007). É também importante que sejam definidos limites nas possibilidades de interação. O utilizador tem que ser impedido de interagir em certos contextos, sendo necessária uma restrição clara por parte do sistema.

Avaliação

Existem vários modelos de avaliação e testes de usabilidade de um sistema. As características enunciadas e explicadas anteriormente foram em parte listadas por Nielsen (Sharp, Rogers et al. 2007). O método heurístico é um método de avaliação empírico. É uma forma rápida e barata de avaliar a interface de uma aplicação, apenas necessitando que um

pequeno grupo de especialistas se reúna para avaliar cada heurística. No final, retira-se a conclusão consoante a média das notas dadas.

Outra forma de avaliar, consiste na utilização de grupos de utilizadores. Estes são por norma o utilizador-alvo do sistema, e permitem validar o que se está a fazer, alertando para aspetos que possam passar ao lado dos *designers*. Pode ser feito recorrendo a simples inquéritos, à observação do seu comportamento, no uso de um sistema, ou mesmo procedendo à criação de testes de utilização (Sharp, Rogers et al. 2007).

É também necessário avaliar outros aspetos mais concretos da interface. Por exemplo, Fitt desenvolveu um modelo matemático que consegue prever o tempo necessário para um utilizador mover o objeto apontador (dedo, rato, etc.) para um determinado sítio, em função da distância ao alvo e do tamanho do mesmo (Seow 2005). Um dos seus maiores benefícios é o de ajudar os *designers* a decidir onde se devem localizar os botões, que tamanho devem ter, e a que distâncias devem estar no mesmo écran (Sharp, Rogers et al. 2007). Mas nem só de movimento se faz a interação. O utilizador tem também que tomar decisões relativamente às diversas alternativas de decisão que se lhe apresentam. Neste sentido Hick criou uma lei que descreve o tempo que uma pessoa demora a tomar uma decisão em função do número de alternativas que identifica.

O processo de *design* de interfaces e de interação atualmente nas empresas, revela falta de planeamento, ou aparece muito tarde no processo de design. De modo a serem desenvolvidos produtos mais ajustados, deve ser dado mais tempo a entender o utilizador, como ele trabalha, como se sente e como a tecnologia poderá ajustar-se às restrições na sua vida (Hong 2011).

2.2.3 Metáforas de interface

"Metaphors are the tools we use to link highly technical, complex software with the user's everyday world." (Weinschenk and Jamar 1997)

A metáfora de interface consiste na criação de um modelo conceptual, que é desenvolvido de forma a incorporar ou ser semelhante em certos aspetos a uma entidade física. Apesar deste aspeto, esta mantém os seus próprios comportamentos e prioridades (Sharp, Rogers et al. 2007). As metáforas descrevem a aparência e comportamento que os elementos devem ter devendo ser facilmente percetíveis pelo utilizador da forma inicialmente proposta (Sharp, Rogers et al. 2007).

A introdução dos sistemas de interface gráfica fez com que o utilizador tivesse mais dificuldade na absorção de elementos gráficos inovadores e por vezes pouco lógicos. Surgiu uma necessidade de minimizar o esforço mental necessário para utilizar um sistema tentando fazer com que o utilizador o integrasse de forma mais próxima - envolvimento direto (*direct engagement*) (Hutchins, Hollan et al. 1985).

O conceito de metáfora é bastante semelhante ao de *affordance,* pois o seu objetivo acaba por ser o mesmo (Blackwell 2006) - promover uma identificação intuitiva e automática da interface pelo utilizador. As pessoas consideram que é mais simples e fácil de aprender, quando estão a trabalhar com base em termos que consideram familiares (Sharp, Rogers et al. 2007).

Em termos gerais, pode-se considerar que existem dois tipos de metáfora: uma baseada no modelo físico e da representação da realidade e outra baseada no modelo de conversação. O modelo físico utiliza uma representação de objetos reais por forma a não haver intermediários nessa interação, aumentando a identificação do utilizador com a mesma (exemplo: sistema *BumpTop*) (BumpTop 2011). O modelo baseado em conversação é mais abstrato. Nesse modelo, a interface é a ferramenta linguística utilizada para o utilizador e o sistema manterem uma conversação (interação), que não se encontra explicitamente representada no mundo real (Hutchins, Hollan et al. 1985). Um exemplo deste tipo de metáfora é o da ferramenta de ajuda do IKEA (AI4US 2011) que responde a perguntas efetuadas pelo utilizador.

Exemplos de metáforas de interface

O Xerox Star foi o primeiro sistema criado com interface gráfica. Foi desenvolvido no PARC (Palo Alto Research Center) e recorreu a uma metáfora de interface que se tornou universal, a metáfora de *Desktop* (Star 2007). O ambiente gráfico foi buscar elementos tradicionais de um escritório de trabalho (pastas de ficheiros, caixote-do-lixo, caixa-de-correio, etc.) (Sharp, Rogers et al. 2007) e adaptou-os a uma interface gráfica, baseando-se no modelo físico.

Apesar do sucesso do Xerox Star o processo de criação de uma metáfora de interface não é algo linear nem previsível. Por exemplo, o Magic Cap (Figura 2) foi criado como um ambiente

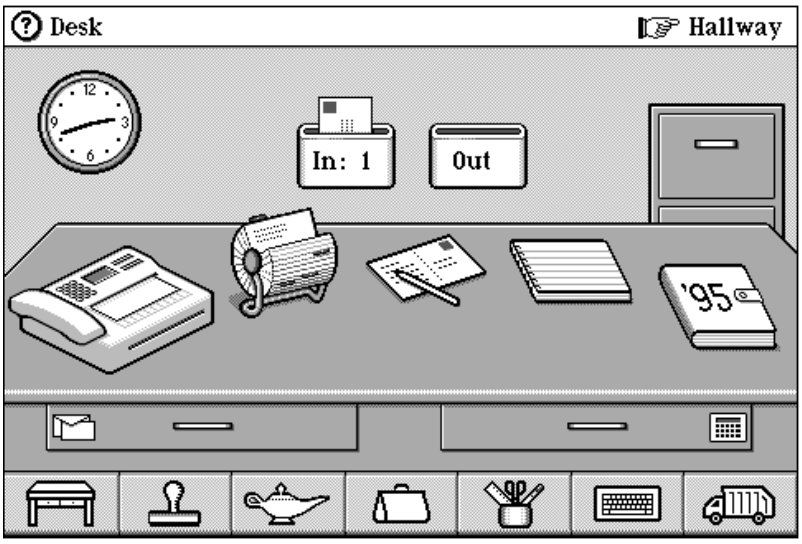


Figura 2 - Magic Cap

gráfico para PDA's (Personal Digital Assistant). Promovia a metáfora de salas, em que o utilizador podia navegar entre divisões de um edifício.

Apesar de ser uma metáfora perfeitamente válida não obteve grande aceitação, pois o esforço cognitivo exigido ao utilizador era bastante elevado. Quando se tenta criar uma metáfora de interface cujo aspeto e comportamento é uma cópia exata da entidade física, perde-se o efeito real de comparação e associação (Sharp, Rogers et al. 2007). Deixa de ser um modelo mental e conceptual para se tornar numa representação exata, o que levanta barreiras complexas de compreensão para o utilizador. Este exemplo demonstra que escolher a representação e os elementos corretos não é um processo nada simples, sendo é aconselhável um extenso período de teste (Shneiderman 1981). No geral, aconselha-se também o uso de metáforas que primam pela simplicidade de representação de elementos, exprimindo conceitos presentes no dia-a-dia do utilizador e permitindo uma interação direta com os dados (Dina Goldin 2006).

No processo de design do Microsoft Surface, a equipa de desenvolvimento responsável pelo projeto recorreu à criação de metáforas de interface. Estabeleceram como objetivo a criação de uma interface de fácil compreensão e previsão para os utilizadores. Neste processo recorreram à criação de diversos protótipos utilizando diferentes modelos conceptuais. Cada protótipo tinha uma lógica de funcionamento, com regras como se se tratasse de um mundo particular. Foram criadas diversas propostas: *Unfold* (simulava o uso de uma revista), *Magnet* (quadro magnético) e o *Sphere* (simulação do universo). Os testes com os utilizadores produziram resultados surpreendentes. Eles perceberam e conseguiram descrever as metáforas na generalidade mas no *Magnet,* por exemplo, o seu entendimento da metáfora foi distinto do que foi pensado e planeado pelos investigadores. Mesmo apesar de não terem compreendido a metáfora na totalidade, esta obteve resultados de aceitação bastantes elevados. Os utilizadores apreciarem os elementos de interação e a rapidez de resposta do sistema. Este exemplo sublinha a importância de equilibrar as opiniões dos especialistas com o teste com utilizadores (Hofmeester 2010).

Este exemplo acentua a ideia de que a criação de um modelo conceptual não pode ser um processo rígido e com normas perfeitamente definidas.

Modelos conceptuais

O modelo baseado em metáforas concretas é composto por uma analogia entre objetos conhecidos pelo utilizador no seu dia-a-dia e aproveita esse conhecimento dos objetos do mundo real (L'Abbate 1998). Este modelo é extremamente rígido, sendo criticado por exercer demasiadas limitações ao utilizador, podendo ir contra os princípios do *design*, e de limitar a imaginação do designer (Sharp, Rogers et al. 2007). Conduz a que a descoberta de novos paradigmas ou modelos de interação desapareça da área do *design*.

Como alternativa a uma representação mais estática de uma metáfora de interface, foi criado o conceito de metáfora elástica (Khoury 2003). Foi desenvolvida traçando um paralelismo com as relações e estruturas sociais a que estamos habituados. A sua estrutura é composta por atores, regras, recursos e ações. Os atores representam os intervenientes na

estrutura social, estrutura esta que possui recursos e regras pré-definidas, que se manifestam pela execução de determinadas ações. Contrariamente às metáforas baseadas no mundo físico, estas relações caracterizam-se por serem maleáveis, adaptando-se, de forma a refletirem o comportamento de uma grande variedade de sistemas. Este modo abstrato de planear uma interface pode permitir uma maior correspondência com as ações e lógica de pensamento do utilizador.

Foram também propostos outros modelos para complementar a interação do utilizador. Esta não se limita apenas à visão do écran, diz também respeito ao toque. Para esta interação foram criados modelos abstratos gestuais, baseados em metáforas. Certos gestos misturam-se com a perceção que a pessoa tem a nível sensorial: por exemplo o padrão "aumentar" é logo associado a um gesto de "subir" tal como o de "diminuir" é assumido como "descer". Este é um exemplo de metáforas primárias relativas a associações entre gestos e ações imediatas. Elas resultam da correlação entre experiências sensoriais e motoras que o utilizador tem no dia-a-dia. O seu reconhecimento é feito de forma automática e subconsciente (Hurtienne, Stößel et al. 2010), tendo a vantagem de ser assumido de igual forma, independentemente da idade ou da experiência de vida do utilizador, aumentando a consistência do sistema.

Independentemente da tecnologia, de forma a obter melhorias na performance e no prazer do utilizador, as futuras interfaces devem ser capazes de preencher certas exigências: modelos mentais, navegação, apresentação, interação e metáforas. As metáforas perfeitas permitem que os utilizadores compreendam, usem e se lembrem da informação mais rapidamente, com maior profundidade e satisfação, cumprindo com as expectativas geradas (Marcus 1998).

2.3 Multitoque

Nesta secção pretende-se analisar o paradigma de interação multitoque. Este tipo de interação promove diversas vantagens para o utilizador, seja através da naturalidade do contacto com os dedos, seja do sentimento mais natural de manipulação. A mudança de paradigma obriga no entanto a uma intensa reformulação do desenvolvimento de *software* em diversos aspetos de forma a otimizar a interação.

Antes de mais é necessário caracterizar a interação. Esta decorre do contacto dos dedos do utilizador na superfície, podendo ser feita recorrendo a um dedo, múltiplos dedos (de uma mão ou de ambas) ou mesmo com vários utilizadores em simultâneo. A complexidade/riqueza da interação está diretamente relacionada com os GDL. Numa típica interação 2D com um rato num PC temos 2 GDL dado que o ponteiro do rato refere a posição usando duas métricas (X e Y). Se se pretender interagir numa superfície multitoque com 2 dedos por exemplo, obtém-se 4 GDL (2 por cada dedo). A interação torna-se mais complexa consoante este número aumenta, tal como se processa na interação com o mundo real, permitindo assim explorar as habilidades adquiridas pelo utilizador no dia-a-dia (Buxton 2007).

É necessário também separar a própria natureza da interação dado que esta pode ser discreta ou contínua (Wu, Shen et al. 2006). Interação discreta caracteriza-se por registar apenas uma posição para cada dedo, não havendo movimentos nem gestos. Um exemplo de interação deste género é o de "mover objeto", no qual o utilizador com um dedo seleciona o objeto que pretende (mantendo o dedo pressionado) e com outro dedo carrega no sítio para o qual pretende mover. Já a interação contínua caracteriza-se por uma alteração contínua da posição dos dedos do utilizador tal como acontece nos gestos de *zoom* utilizados no IOS da Apple. Com este gesto, o utilizador aplica dois dedos em contacto com a superfície e, sem os levantar, aproxima-os ou afasta-os (*zoom-out* e *zoom-in* respetivamente). Pode existir um cenário que interligue estes dois tipos de interação, por exemplo, na simulação da utilização de um programa de desenho. O utilizador, com um dedo, seleciona a ferramenta e, com outro dedo, executa a ação (pintar por exemplo), havendo portanto interação contínua e discreta.

Dado que, tipicamente não existem periféricos num sistema multitoque para introdução de dados é necessário repensar este processo. Deve-se recorrer a soluções a nível de *software* (*soft keyboards* por exemplo) cuja interação se faça através do toque na superfície.

Outro aspeto importante é o de adequar as ações que seria possível realizar com um rato, de forma a poderem ser interpretadas por gestos. Como exemplo, as ações do tipo *mouse-over* não são possíveis de realizar com o toque em superfícies multi-tácteis. É portanto fulcral saber adaptar as funcionalidades ao tipo de interação mais apropriado, seja a nível de toque ou de gesto. Numa interface gráfica do paradigma WIMP é comum utilizar-se o botão de minimizar para esconder uma janela, o que se torna mais complicado com toque, dado que o botão do rato é bastante mais preciso. Neste caso, um gesto de deslizar sobre a janela poderá ser mais eficaz e intuitivo. Este exemplo levanta o problema da interface WIMP (padrão na maior parte das aplicações atualmente) não ser aconselhável para a interação multitoque. É necessário pensar a interface e otimizá-la para este tipo de interação, seja a nível do tamanho dos objetos a manipular, seja a nível dos elementos que a compõem. Menus, *scrollers,* janelas são exemplos de componentes desenvolvidos para o WIMP, que devem ser alterados, para que sejam intuitivos de usar, e para que consigam aproveitar todas as vantagens da interação multitoque.

2.3.1 Limitações da tecnologia

A tecnologia *multitoque* tem imensas vantagens a nível de interação mas, apesar de tudo, também tem limitações. Muitas destas limitações refletem-se na génese, na interação e, portanto, é necessário saber quais são e como as evitar de forma a melhorar a experiência de utilização.

Oclusão

O facto de o utilizador usar os dedos para interagir obriga a que a sua mão acabe por tapar

Figura 3 - Menu que evita oclusão do dedo (Brandl, Leitner et al. 2009)

em parte ou na totalidade a superfície. Imagine-se um botão que, quando pressionado, abre uma caixa de texto imediatamente por baixo. O utilizador, quando carrega, pode não conseguir visualizar a caixa de texto e concluir que a ação não obteve efeito. Pode também acontecer que o próprio dedo tape o botão que é pressionado, o que é apelidado de problema do dedo gordo (*fat finger problem*) (Wigdor, Williams et al. 2009). Como a área de contacto é reduzida a um único ponto de seleção, pode levar acontecer que a pessoa carregue num botão e não perceba porque é que este não registou a ação, dado que não existe *feedback* do resultado (por exemplo, ações *mouse-pressed* com interação com rato). Para resolver este problema, os investigadores têm proposto diversas soluções. Relativamente ao *feedback,* o Ripples (Wigdor, Williams et al. 2009) propõe a visualização permanente de todos os contactos com a superfície. Este comportamento contribui para manter o utilizador alinhado com o sistema. Outra proposta consiste na identificação pelo sistema das áreas que estão a ser tapadas pela mão e braço do utilizador, fazendo aparecer os conteúdos em bolhas de informação (Vogel 2010). Esta ideia necessita que o sistema multitoque seja complementado com outro tipo de *hardware*, por forma a reconhecer objetos acima da superfície sem estarem em contacto com ela. Dessa forma é possível estimar a direção do toque e perceber a área sobreposta. Com base no mesmo conceito, foi também proposto um menu circular (Figura 3). Este estima a posição do utilizador, de forma a que os itens do menu não apareçam sobrepostos (Brandl, Leitner et al. 2009).

O menu encontra-se sempre completamente visível dado que, no espaço onde vai estar a mão, não são inseridos botões. Outra forma de detectar a posição do utilizador é recorrer ao estudo do registo de contacto do dedo do utilizador na superfície (Wang, Cao et al. 2009). Recorrendo a algoritmos sobre o padrão do toque, é possível estimar a direção em que o utilizador se encontra, orientando os conteúdos com que ele está a interagir.

Num contexto mais específico de utilização, foram sugeridas diferentes técnicas para evitar oclusão em dispositivos tácteis de pequena dimensão. O *Shift* (Vogel and Baudisch 2007) utiliza

a ideia das bolhas de informação já apresentadas, criando uma lupa acima do ponto de contacto no qual é apresentada a informação oculta. Já o *Escape* (Yatani, Partridge et al. 2008) recorre à direção do gesto para reconhecer o objeto que se deseja selecionar. O conteúdo mantém-se oculto, mas oferece uma forma de o utilizador poder efetuar uma seleção, mesmo quando o toque ocorre sobre diversos objetos ao mesmo tempo.

Precisão

A interação com rato é uma interação indireta no sentido em que utiliza um dispositivo para impor uma posição no écran. Esta posição é representada por um cursor que, ao contrário do dedo humano, é bastante preciso. Quando se passa para uma interação multitoque é fundamental perceber que se perde o poder de precisão e seleção, dado que o contacto do ponteiro do rato é de menor dimensão do que o resultante da interação com toque. As dimensões por omissão de botões, barras de título ou de menus utilizados em GUI são adaptadas ao ponteiro do rato, o que não permite uma seleção precisa pelo toque (Ryall, Forlines et al. 2006). É aconselhável uma restruturação da interface gráfica das aplicações, nomeadamente na alteração na alteração do tamanho dos objetos manipuláveis. Os alvos de interação devem ser de tamanho superior a 11.52mm, de lado, para objetos quadrados (Wang and Ren 2009). Caso não seja possível este redesenhar da interface, podem ser utilizadas diversas técnicas que consigam contornar ou minimizar este problema. Potter et. Al (Potter, Weldon et al. 1988) sugeriram três técnicas distintas para seleção precisa em écran táctil que tiveram algum apoio da comunidade científica: *land-on* que consiste na interpretação do toque inicial, *first-contact* que regista o primeiro objeto a ter contacto e *take-off* que consiste no registo da posição em que o utilizador levanta o dedo da superfície.

Estas estratégias não têm em conta o problema das dimensões dos objetos, nas interfaces desenhadas para interação com rato. Para esses casos é necessária uma abordagem diferente.

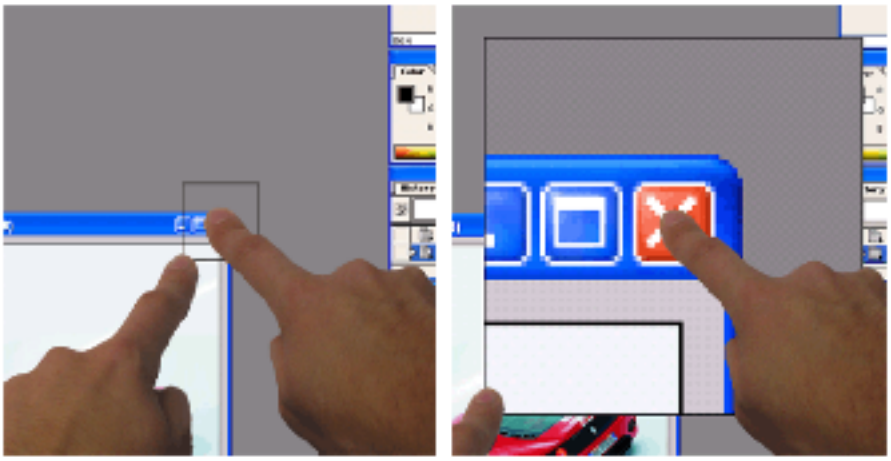


Figura 4 - Dual Finger Stretch (Benko and Wilson… 2006)

Albinsson e Zhai (Albinsson 2003) propuseram diferentes métodos *Cross-keys* , *Precision-handle* e *Zoom-pointing* para permitir seleção ao nível do pixel. As duas primeiras baseiam-se na utilização de um cursor com controlos embutidos para permitir melhor seleção. O último método refere-se à utilização de *zoom* para permitir aumentar o tamanho do alvo em causa. Olwal et al. (Olwal and Feiner 2008) propuseram dois géneros de técnicas também baseadas no *zoom*, *rubbing* e *tapping*. O deslizar do dedo na diagonal noroeste-sudeste repetitivamente faz *zoom-in* e o gesto na direção nordeste-sudoeste executa *zoom-out*. Estes métodos utilizam outro dedo (preferencialmente da outra mão) para seleção. Ainda nos métodos com base em *zoom*, Benko et. Al (Benko and Wilson 2006) desenvolveram uma técnica que utiliza dois dedos para *zoom* e seleção, *Dual Finger Stretch* (Figura 4).

Este método permite que um dedo efetue *zoom* de uma parte do écran, enquanto que o outro dedo efetua a seleção desejada. Outro método de seleção proposto recorre à representação do cursor numa certa posição utilizando dois dedos para o efeito. Esta corresponde ao ponto médio entre os dois dedos, o que permite uma visualização do alvo ao nível do pixel.

Ergonomia

A interação com um écran táctil faz com que o esforço físico imposto ao utilizador seja maior. É importante que a superfície não esteja em posições extremas: a 90 graus leva a que o utilizador tenha permanentemente os braços em suspensão originando fadiga muscular numa utilização de longo tempo (*Gorilla Arm*) (Wigdor, Penn et al. 2007) e se estiver no tampo de uma mesa (como o Microsoft Surface) obriga o utilizador a curvar-se demasiado, podendo ser bastante desconfortável. É necessário também dar atenção aos gestos que são exigidos ao utilizador. É aconselhável utilizar gestos no qual o movimento do braço seja mínimo ou, caso

seja possível, substituí-los pelo uso apenas de *tapping* (Wang and Ren 2009). Os gestos devem ser fáceis de reproduzir, não exigindo grande complexidade a nível de elasticidade dos dedos. Acima de tudo devem ser movimentos naturais e intuitivos.

É necessário também relembrar que, num dispositivo que suporte multitoque todos os toques contam. Devido a este aspeto, surge um problema - distinguir a interação da intenção, que ocorre por engano. O toque acidental, seja pela colocação do braço ou do cotovelo no écran, é também interpretado como interação válida e, tal facto pode causar ambiguidade (Ryall, Forlines et al. 2006). Este aspeto carece de um estudo mais apurado, a fim de ser feita a diferenciação entre o que é intencional e o que é casual.

2.3.2 Desafios específicos do multitoque em HMI/SCADA

Interface

A interface nos dispositivos multitoque, como já foi abordado, necessita de um cuidado especial e uma reorientação em função da interação. Nesse sentido é fulcral adaptar novos paradigmas de disposição de informação, de componentes e de controlo. Dado que o WIMP não é apropriado a este tipo de interação, existe uma necessidade de procurar soluções mais naturais e integradas. Nas interfaces pós-WIMP é proposta a eliminação de menus, formulários, barras de ferramentas e basear toda a interação em gestos e na voz - *"The ideal interface is no interface"* (Van Dam 1997). A interação ideal é pensarmos no que queremos e o sistema conseguir perceber e antecipar as nossas necessidades. Esta é a base das *Perceptual User interfaces*. São caracterizadas por técnicas de interação que tentam combinar a compreensão das capacidades naturais do ser humano, com as capacidades de perceção e raciocínio de um sistema informático (Turk 2000). Enquanto algo semelhante não é possível, devemos desenvolver esforços para minimizar os mecanismos de manipulação, diminuindo a distância cognitiva entre a interface e a execução.

Com estes conceitos como base, foram criados paradigmas de interface que procuram dar um salto relativamente às interfaces WIMP. Um paradigma interessante de aplicar é o de *semantic zoom* (Bederson, Hollan et al. 1996), no qual a interface é apresentada com diferentes níveis de profundidade. Quando é feito *zoom* sobre uma área do écran, toda a interface tem que ser reavaliada, por forma a representar as dimensões adequadas. É natural observar mais detalhes à medida que o *zoom* é maior, e é esta reavaliação que deve ser efetuada. Este paradigma foi experimentado no desenvolvimento do Pad++ (Bederson, Hollan et al. 1996) que é basicamente uma superfície de dados que permite ao utilizador navegar através de gestos multitoque de *zoom-in*, *zoom-out* e *pan*. Este paradigma foi aproveitado para a criação do ZOIL (Jetter, Engl et al. 2008) *(Zoomable Object-Oriented Information Landscape)* que foi projetado para servir como substituto do WIMP. Distingue-se do *semantic zooming,* por ser orientado a objetos, na medida em que para aceder às funções e detalhes de um objeto é necessário fazer *zoom-in* sobre o objeto. Este paradigma foi também utilizado na navegação em diagramas de

grande dimensão, no qual se concluiu que a navegação se tornou mais rápida e precisa do que a que se fazia com rato (Marinos, Geiger et al. 2010).

Um problema de que o WIMP sofre é o de oclusão de janelas e objetos que se acumulam na interface. Para corrigir essa limitação foram propostos diversos conceitos distintos, sendo alguns baseados na transparência dos objetos (*content-aware free-space transparency*) (Ishak 2004) e outros na aplicação de metáforas naturais (*paper-based interaction*) (Dragicevic 2004). O primeiro caso referido aplica algoritmos que determinam as áreas mais e menos importantes, usando transparência quando é necessário revelar uma área de interesse. O segundo aplica uma técnica de *drag-n-drop* entre janelas, na qual os objetos se comportam como folhas de papel. Quando são puxadas, (*fold*) revelam o que se encontra por trás, permitindo também interação com esses objetos. Besacier et al. (Besacier, Rey et al. 2007) desenvolveram também metáforas de interface baseadas em papel, utilizando a dobra (*fold* e *peeling*) como inspiração. Este conceito permite agrupar objetos como se se encontrassem numa capa, permite transmiti-los, recorrendo ao conceito de *slots* (caixas que servem para arrumar ou transmitir informação), e permite, ainda, empilhar objetos como se se tratasse de uma pilha de papéis.

Menus

Os menus típicos da interação WIMP requerem um nível de precisão que é inapropriado à interação táctil. O utilizador pretende que os menus suportem o maior número de opções, que sejam de acesso rápido e que, acima de tudo, sejam consistentes. Neste paradigma, os mesmos são ativados por um clique no botão direito do rato (ou equivalente) e, posteriormente, a escolha é efetuada com um clique no botão esquerdo. Como nas interfaces tácteis, para referenciar qual dos dedos foi ativado é necessário recorrer ao registo e interpretação de imagem em tempo real

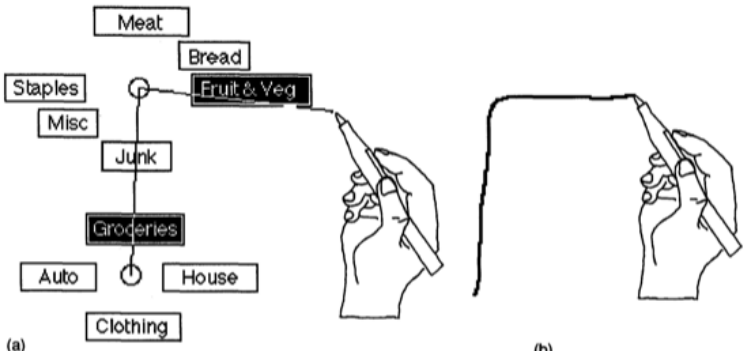


Figura 5 - Exemplo de *marking menu*. (Kurtenbach… 1993)

(pesado a nível computacional), é fundamental encontrar outra abordagem mais simplista. Kurtenbach e Buxton desenvolveram em 1993 o *marking menu* (Kurtenbach 1993), um tipo de menus que permite que o utilizador escolha a opção que deseja sem precisar sequer de o chamar. Os *marking menus* (Figura 5) são ativados pelo utilizador e aparecem geralmente em forma circular, sendo as opções ativadas recorrendo a gestos *take-off* (analisado em 2.2.1). *"Gestures are not only faster but also less error-prone"* (Roudaut, Bailly et al. 2009). O

utilizador seleciona uma opção, ativando o menu com o toque e, sem o levantar, desenha uma linha reta até à opção pretendida (marca composta). À medida que vai sendo utilizado, a pessoa consegue recordar as marcas associadas a cada opção, podendo reproduzi-las sem necessidade de ativar o menu. Em termos de formato gráfico, foi utilizado neste tipo de menus o de *pie menu*, no qual as opções se encontram divididas de forma circular, como se se tratasse de uma tarte. Em comparação com a representação linear de menu, esta permite uma menor distância desde o ponto de ativação até à posição alvo, para além do espaço que cada opção ocupa ser mínimo (Callahan, Hopkins et al. 1988). São bastante fiáveis e permitem uma utilização mais rápida, tanto por, por utilizadores novatos, como por utilizadores experientes (Hopkins 1991). Por outro lado, o menu no seu total ocupa bastante mais espaço que um linear, e tem um número mais limitado de itens.

Inicialmente, os investigadores aconselharam que os menus radiais contivessem no limite oito opções, até ao máximo de dois níveis, dado que a precisão na seleção depende do número de itens (Kurtenbach 1993; Zhao and Agrawala 2006). Esta limitação obrigou a uma natural evolução do conceito de *marking menu*. Seguindo o objetivo de aumentar o número de opções, foram desenvolvidos os *Zone, Polygon* e *Flower Menus* (Zhao and Agrawala 2006). São menus de marca múltipla (marcas sucessivas) que funcionam recorrendo à posição relativa e à orientação da marca. Ao contrário dos menus baseados em marca composta, estes não exigem que o utilizador permaneça sempre com o dedo em contacto com a superfície táctil.

Os menus anteriormente apresentados (*Zone, Polygon* e *Flower*) servem, acima de tudo, para melhorar a ação de utilizadores experientes. De forma a melhorar o modo de utilizadores com pouca experiência, foi criado o *Wave Menu* (Figura 6) (Bailly, Lecolinet et al. 2007). É um

Figura 6 - Exemplo de *Wave Menu* (Bailly, Lecolinet et al. 2007)(Bailly, Lecolinet et al. 2007)

menu em formato de anel, que permite ter sub-anéis hierárquicos, utilizando o conceito de marcas compostas dos originais *marking menus* e as marcas múltiplas do *Zone Menu*. O menu desenvolvido tem um comportamento eficaz, mesmo quando é utilizado por um utilizador

experiente. Ainda assim requer imenso espaço (ocupa em média o dobro do espaço de um menu linear), aumentando gradualmente à medida que são acrescentados subníveis. A experiência efetuada pelos investigadores apontou para ganhos significativos em termos de rapidez de execução, mas apontou também para um excessivo número de erros nos menus baseados em marca múltipla (cerca de sete vezes mais do que as outras técnicas).

O facto de os *marking menus* ocuparem bastante espaço, levou ao surgimento de novos conceitos no posicionamento e na criação de menus. Foram estudadas as preferências do utilizador e a mão utilizada para interação (Hancock 2004), de forma a tornar o processo de escolha mais natural. Com base nesse estudo foi desenvolvido o *MultiTouch Menu* (MTM) (Bailly, Demeure et al. 2008), menu hierárquico por natureza (Figura 7). É ativado utilizando o polegar, no qual é disposto um menu circular, que pode conter até oito comandos ordenados. O clique do polegar num deles, ativa o submenu correspondente que é afixado sobre a projeção dos restantes quatro dedos da mão.

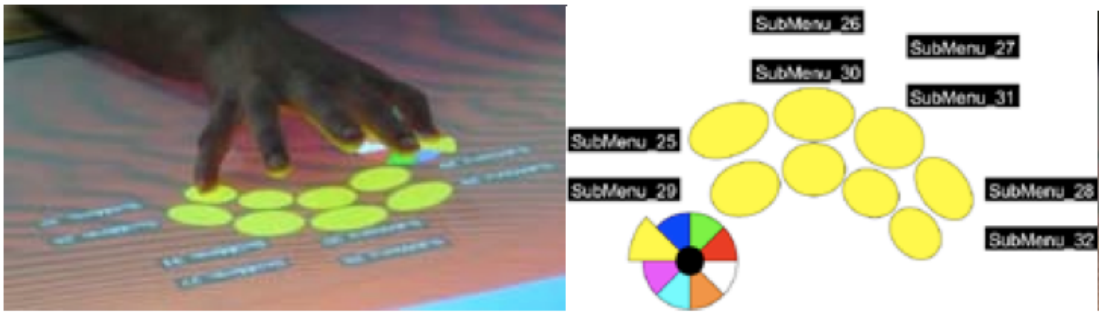


Figura 7 - *MultiTouch Menu* (Bailly, Demeure et al. 2008)

O sucesso deste tipo de representação baseia-se na capacidade inerente ao ser humano de distinguir os seus dedos, e de facilmente associar comandos a cada dedo respetivamente. É adequado a sistemas baseados em mesa, pois a nível ergonómico é bastante mais natural o gesto de apoiar a mão numa mesa, do que a manter em suspensão num écran vertical.

Controlos

A inserção de dados em *multitoque* pode ocorrer de três formas distintas: teclado físico, reconhecimento de escrita e teclado baseado em *software* (*soft keyboard*). O reconhecimento de escrita com recurso ao toque tem pouca previsão, não sendo adequado para longas passagens de texto (processo lento de reconhecimento), mas pode ser suficiente para pequenas anotações (Hinrichs, Hancock et al. 2007). Por outro lado, permite que o utilizador não necessite de qualquer tipo de habituação pois a forma como escreve mantém-se igual. A última opção mais indicada para quem já tem familiaridade com escrita em teclados físicos (principalmente a nível de disposição das teclas). Permite uma maior portabilidade do sistema e elimina por completo a utilização de periféricos de manipulação. A utilização de um sistema de escrita com *soft keyboard* (Figura 8), não provoca grandes diferenças de produtividade (Wigdor, Penn et al. 2007), podendo, portanto, ser adotado sem receio. Estudos demonstraram que esta solução se

comporta melhor em termos de precisão, tempo de execução de tarefas e satisfação de

Figura 8 - Exemplo de soft keyboard

utilização, quando comparada com técnicas de reconhecimento de gestos (Kleid 1996). Não existem diferenças de comportamento relativamente ao sexo ou mão preferencial do utilizador.

O formato pode ser adaptado consoante as necessidades preferenciais do utilizador, sendo utilizada por omissão a disposição QWERTY. O tamanho das teclas é também um pormenor a ter em conta. Esta necessidade de adaptar o tamanho das teclas ao utilizador, advém da preocupação de tornar os *soft keyboards* mais usáveis e intuitivos. Gunawardana et al. (Gunawardana, Paek et al. 2010) sugerem uma nova forma de dividir a área-alvo dos botões do teclado, de forma a reduzir a quantidade de erros que surgem na escrita. Para isso recorrem a algoritmos preditivos que alteram dinamicamente as áreas de alvo de cada botão.

Ao longo do tempo foram criadas ferramentas para interface que, apesar de não terem sido pensadas objetivamente para multitoque, podem ser adaptadas com sucesso. Os *sliders* são muito utilizados no iOS para configurar valores sem necessidade de entrada de dados. São uma evolução natural do componente utilizado em WIMP (caixa de texto para preenchimento, com setas para aumentar ou diminuir valor) e permitem uma representação gráfica imediata (*feedback)*. A combinação da representação gráfica do parâmetro e do respetivo resultado vai de encontro ao conceito de *direct manipulation* (Ahlberg, Williamson et al. 1992) (abordado em 2.1).

Outros autores recorrem à introdução de novas técnicas de interação que permitem explorar novas representações gráficas. Os *floating palettes* são painéis de botões (representam comandos) que, quando selecionados, permitem aplicar essa ação a representações gráficas (Mackay 2002). Uma técnica que utiliza estas propriedades é a de *toolglass* (Bier, Stone et al. 1993). Trata-se de um painel flutuante com propriedades de transparência, que permite ver e efetuar ações através do próprio painel. Ao contrário dos *floating palettes*, estes não se cingem a apenas uma aplicação, podendo ser aplicados em qualquer objeto da interface. Esta propriedade

aumenta a consistência da aplicação, permitindo que diversos programas partilhem funcionalidades comuns (Bier, Stone et al. 1993). Aproveitando ainda a noção de transparência, os mesmos autores criaram o *Magic lenses*, painel transparente que também pode ser aplicado a qualquer objeto da interface. Este painel, quando se encontra por cima de um objeto, permite revelar pormenores que não estão disponíveis sem a ação da ferramenta. O conceito é semelhante ao de *zoomable interfaces,* no qual para se observarem pormenores de um objeto é necessário efetuar uma ação sobre ele. Há que referir que esta técnica foi desenvolvida para utilização com dois periféricos: um controla a posição do painel, e outro controla a aplicação da ação ao objeto em causa. Pode ser aplicada no contexto do multitoque, mas requer algumas adaptações a nível de mecanismo de interação.

Gestos

A grande vantagem de uma interface multitoque é a de permitir que o utilizador interaja com a superfície táctil, recorrendo a toques simultâneos e a gestos para interpretar ações. Enquanto que com um rato apenas temos um dispositivo apontador, numa interface multitoque podemos selecionar e manipular objetos dispersos com as duas mãos. Com o rato podemos recorrer ao toque no objeto e ao *lasso* para seleção individual e em grupo (Dwyer, Fisher et al. 2011), apesar de só termos um dispositivo apontador. No multitoque podemos também recorrer ao toque, mas utilizamos vários apontadores simultaneamente (dedos).

Os gestos de interação podem ser executados recorrendo a uma ou duas mãos. Estudos efetuados referem que os utilizadores por vezes sentem dificuldades a trabalhar com as duas mãos de forma independente (Wobbrock and Morris 2009; Dwyer, Fisher et al. 2011). Por esse motivo, é essencial efetuar um estudo aprofundado do *design* dos gestos e da interação pretendida. Wu et. al desenvolveram conceitos que se revelaram fundamentais para entender esta problemática. Introduziram três princípios fundamentais de *design*: *gesture registration* (registo), *relaxation* (relaxamento) e *reuse* (reutilização de primitivas de gestos) (Wu, Shen et al. 2006). Juntando estes princípios, é possível criar grupos e sequências de gestos que se mantêm consistentes dentro de uma aplicação e que libertam o utilizador da complexidade de interação. Lao et al. (Lao, Heng et al. 2009) criaram um modelo de *design* de interação gestual para multitoque, com base nestes conceitos. Separaram o modelo em três níveis: *action, motivation* e *computing level*. O primeiro é independente da aplicação e assenta na definição de gestos simples (*pressing, tapping* e *dragging*) e complexos (compostos por gestos simples). Na segunda fase (*motivation*) é feito um estudo do objetivo da aplicação e da motivação do utilizador (independente do sistema ou da plataforma) (Lao, Heng et al. 2009). Por fim, no *computing level,* é efetuado um levantamento das características de hardware e software que suportam o sistema multitoque (reconhecimento de gestos, de toque, de pontos de interação, etc.). Estas três fases aumentam consideravelmente a consistência do planeamento de gestos de um sistema e a própria reutilização dos mesmos.

É frequente a interação ser pensada e desenhada pelos *designers* de interação que, posteriormente, explicam ao utilizador como esta funciona. Esta forma de pensar retira o utilizador do processo de interação, dando origem a gestos que não refletem o seu comportamento. Wobbrock et al. inverteram o processo, promovendo uma atividade no qual os próprios utilizadores criavam e sugeriam gestos, consoante a atividade ou ação pretendida. Este processo originou uma lista de *user-defined gestures* (Anexo A)(Wobbrock and Morris 2009), alvos de estudo e análise para retirar conclusões quanto às preferências do utilizador. No final, concluiu-se que os gestos efetuados com apenas uma única mão foram da preferência da larga maioria dos presentes. Percebeu-se também que a reutilização de gestos é algo a que os utilizadores recorrem por vezes inconscientemente e que ajuda no processo de aprendizagem. Conseguiu-se também observar que os gestos não devem ser distinguidos relativamente ao número de dedos. As pessoas geralmente não ligam ao número de dedos que utilizam quando executam uma tarefa qualquer no mundo real (tocar piano, etc.) (Wobbrock and Morris 2009). Relativamente ao número de dedos, percebeu-se que quatro dedos deve ser o limite de utilização num único gesto.

Após este estudo, Morris et al tentaram perceber concretamente as preferências dos utilizadores. Para isso compararam os gestos criados no *user-defined gestures,* com um conjunto de gestos desenvolvidos por três investigadores da área de IPC (Morris and Wobbrock 2010). Recorreram a vinte e uma pessoas que validaram ambos os conjuntos e efetuaram a sua análise crítica. No final, concluíram que os participantes demonstraram maior preferência por gestos que fossem fáceis de executar e que exigissem menor esforço cognitivo, em vez de gestos mais complexos. Os gestos de execução com apenas uma mão foram escolhidos por mais participantes, do que os que necessitam da intervenção de ambas as mãos. No mesmo sentido, os gestos que utilizam um dedo, foram de maior agrado do que os que utilizam múltiplos dedos. Esta conclusão é interessante pois, Moscovich e Hughes (Moscovich 2008) tinham concluído que, a melhor abordagem para o controlo de dois pontos se verifica com a utilização das duas mãos (um dedo em cada). Já para controlar uma única posição, afirmaram que era aconselhável usar dois dedos de uma mão, permitindo fazer *scale* ou *rotate*. Esta visão valida em parte os estudos anteriores, em relação a gestos com interação com uma e duas mãos.

No que diz respeito aos gestos, é visível a criação de gestos que se começam a tornar padrão em várias aplicações. Por exemplo, o *tap* para selecionar objetos, o *pinch* para efetuar *zoom*, ou mesmo o *drag*/*pan* para mover objetos. Num estudo de Wu e Blakrishnan (Wu 2003), o gesto de *rotate* é realizado com recurso à rotação de dois dedos, sendo este procedimento utilizado também no iOS e também num modelo de interação proposto por Kim et al. (Kim, Kim et al. 2006). Já no estudo dos *user-defined gestures*, a rotação é efetuada selecionando o canto do objeto, rodando até à posição desejada. O gesto de *tap* é realizado recorrendo a um único dedo (para selecionar uma única unidade) ou a dois dedos deixando o primeiro pressionado sobre um objeto, sendo o segundo responsável por selecionar os restantes. Em relação aos gestos de *pinch* e *rotate*, estes necessitam da utilização de dois dedos (uma ou duas

mãos). Apesar de pequenas diferenças, os utilizadores têm representações mentais e conceptuais de como se processam estas ações no mundo real.

Ainda no estudo de Morris et al, em termos conceptuais os gestos baseados em analogias físicas foram alvo de preferência em detrimento dos que se baseavam em analogias abstratas (Morris and Wobbrock 2010). Por fim, os autores concluíram que os gestos propostos pelos peritos em IPC eram mais criativos e visualmente mais agradáveis, mas no final os participantes demonstraram preferência pelos mais simples (propostos pelo grupo de utilizadores, no anterior estudo). Este artigo demonstra que o processo de *design* de interação tem mais sucesso, se for acompanhado por um grupo de utilizadores-alvo, que possam sugerir e testar os gestos propostos.

Emulação do rato

A adoção de ações do rato na interação multitoque pode ser entendida como um retrocesso no processo de interação. São poucas as aplicações que foram totalmente planeadas (ou refeitas) para uma interação multitoque exclusiva, por essa razão não se podem ignorar as possibilidades que se perdem quando se abandona o rato para interagir (*mouse over,* clique com o botão direito, etc.). De forma a eliminar este problema, desenvolveram-se conceitos de emulação de ações do cursor. Moscovich e Hughes (Moscovich 2006) propuseram um método que utiliza um mapeamento de posições para simular os parâmetros de um cursor (*hand cursor*). Cada dedo controla um parâmetro independente associado ao rato, o que permite manipular objetos como se estes fossem físicos. Como em geral apenas controlamos um objeto de cada vez, os investigadores desenvolveram o conceito de *similarity cursor,* no qual os parâmetros de uma mão são reduzidos apenas a uma posição (Moscovich 2006). Os parâmetros são controlados por dois dedos e representados por um cursor. Desta forma é possível selecionar um objeto e aplicar-lhe diversas ações simultaneamente (mover e rodar). Obtém-se *feedback* permanente através da representação do cursor, à qual são aplicadas as mesmas ações produzidas pelo toque (rodar, mover, etc.).

Com a mesma preocupação, Matejka et al. (Matejka, Grossman et al. 2009), criaram alternativas gestuais para ações do rato. Estas alternativas propõem-se a suportar o contínuo registo de percurso sem toque (*mouse-over)*, a emular a ação de clique simultâneo nos botões do rato (*middle click*) e o uso da *scroll wheel*. Sugeriram diversas técnicas, entre as quais o *SDMouse*, para a representação destas ações, recorrendo todas ao número de dedos em contacto permanente com a superfície táctil. Os estudos concluíram que, utilizando estas técnicas os utilizadores cometem mais erros, possivelmente pela dificuldade na utilização do mapeamento dos dedos no écran (Matejka, Grossman et al. 2009).

2.4 Sistemas HMI/SCADA

Os sistemas SCADA são plataformas de aquisição e controlo de dados para automação. Permitem um controlo de componentes físicos através de representações gráficas abstratas, com o objetivo de manter uma supervisão rápida e eficiente de processos industriais.

2.4.1 Automation Studio

O Automation Studio (AS) é uma solução de *software* desenvolvida pela equipa de I&D da empresa Efacec. Permite a configuração de dispositivos de automação de forma gráfica através um ambiente de desenvolvimento de simples execução. Possibilita também a programação destas unidades, através do carregamento para as mesmas, de código e informações relevantes no seu processo de execução. No fundo, esta aplicação permite especificar, desenhar e gerir todo o ciclo de vida de máquinas eletrónicas. É utilizado maioritariamente por operadores especializados para o efeito, com conhecimento e formação na plataforma e na área de automação.

A aplicação está dividida em cinco módulos essenciais: Server, Designer, Online, Simulation e VM.

O módulo Server é responsável pela comunicação e atualização de dados e variáveis na máquina, tendo em conta a configuração executada. Funciona também como um elo de ligação entre a unidade de representação gráfica e a máquina de automação, atualizando as animações e elementos gráficos à medida que os valores na sua base de dados são alterados. É também o componente para o qual são enviados os eventos de execução de controlos e variáveis.

O módulo Designer permite a criação de páginas dinâmicas em linguagem XAML (eXtensible Application Markup Language) com objetos gráficos vetoriais 2D. Esta linguagem é baseada em XML sendo utilizada para criação de interfaces gráficas em ambiente WPF. O módulo Designer possibilita atribuir aos objetos desenhados neste tipo de páginas diversos tipos de animações 2D, tais como as de rotação, escalamento, translação ou mesmo de alteração de visibilidade. As animações podem estar dependentes do estado de uma variável do sistema, havendo um reconhecimento visual e automático quando existe uma mudança de estado. Os objetos dos sinópticos podem ser agrupados em símbolos, podendo haver uma reutilização de componentes entre páginas, aplicações ou mesmo configurações distintas. É ainda possível associar a eventos de um objeto (clique do rato, por exemplo) determinados comandos de execução tais como os de escrita de valor numa variável da base de dados, execução de um controlo ou mesmo ação de navegação para outra página. Estas interatividades são geridas e tratadas pelo módulo Server já analisado anteriormente.

O módulo Online permite fazer a ligação a um dispositivo eletrónico e executar a configuração da máquina numa sessão virtual, evitando que o operador se tenha que deslocar até

à máquina. Toda a interação funciona de forma semelhante, sendo que o único obstáculo poderá ser algum atraso na comunicação derivado às condições de rede.

O módulo Simulation permite testar e executar a configuração de uma máquina eletrónica no próprio AS simulando o comportamento na mesma. É usado para teste no processo de desenho e construção das configurações desejadas. É semelhante ao conceito do modo Online, pois permite executar uma configuração num computador normal, embora no modo Simulation, a configuração seja executada nativamente. Este módulo é crucial, dado que as configurações inseridas nas máquinas são componentes críticos que necessitam de ser testados intensivamente, à procura de falhas que o comprometam. O componente de simulação obtém os dados através de uma máquina virtual, que se comporta como um sistema físico real.

A configuração, após criada e testada, é carregada para a máquina em causa, na qual o programa gerado é executado, transformando-se no meio de interação entre a máquina e o operador.

2.4.2 Aplicações semelhantes

O paradigma de interação multitoque foi utilizado neste contexto na aplicação GENESIS (ICONICS 2011), desenvolvido pela empresa ICONICS. Trata-se de uma plataforma de automação e de visualização de informação em 3D desenvolvida para Windows 7 com

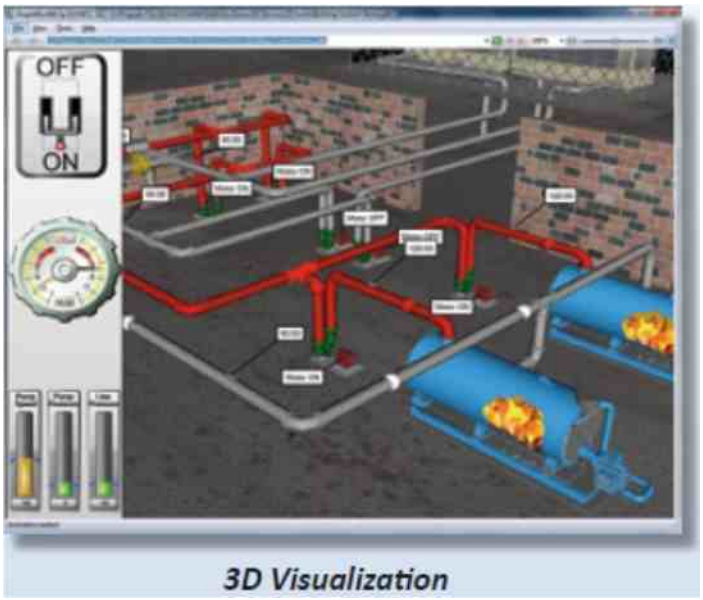


Figura 9 - ICONICS Genesis 64 (ICONICS 2011)

tecnologia Silverlight (Figura 9). O facto de ter sido desenvolvida nesta plataforma permite executar a aplicação em diferentes navegadores de internet e em sistemas operativos distintos. É possível navegar na interface através de *zoom* e *pan*, estando esta orientada relativamente aos objetos representados. O utilizador consegue selecionar um objeto, deparando-se-lhe com os

parâmetros possíveis de serem manipulados. Existe ainda a possibilidade de entrar dentro dos objetos selecionados (entrar numa turbina, por exemplo), sendo que este conceito recorre ao paradigma de interface ZOIL já analisado. Este sistema permite, também, adicionar a objetos representados, determinadas ações de execução, ações essas que são embebidas em controlos existentes no sistema. Estes controlos podem ser do tipo de regulação escalar de valores (discretos ou contínuos) ou mesmo de controlo de estado binário (*on* ou *off*, por exemplo). Os controlos de regulação de valores podem ser lineares (representados na horizontal ou na vertical) ou rotacionais. Quando se deseja abrir ou alterar um destes, deve-se primeiro tocar no objeto em causa, abrindo uma janela com o controlo de mudança de estado ou valor.

A aplicação possui um módulo de alarmes geral que notifica o utilizador de todas as alterações no sistema. Este módulo é global a toda a aplicação e tem uma grande preocupação grande com a fiabilidade, garantindo que não existem alarmes que se perdem em caso de falha do sistema.

Esta é a única solução comercial desenvolvida até ao momento que utiliza o paradigma do multitoque. Diferencia-se ainda assim da proposta deste projeto, dado que a sua interface se baseia numa visualização e navegação em três dimensões, enquanto que a que se pretende desenvolver pretende-se que seja apenas em 2D.

No que diz respeito a soluções SCADA sem multitoque, existem várias alternativas disponíveis: LAquis (LCDS 2011), WinCC (Abacus 2011), OPC Systems.NET (Systems 2011), Cimplicity (GE 2011), etc. Em geral oferecem visualização de entidades representativas de componentes industriais (algumas em 2D apenas, enquanto outras em 2D e 3D), suporte para criação e desenho de gráficos e controlo de parâmetros.

2.5 Software e frameworks

Nesta secção são apresentadas soluções de *software* que foram identificadas como utilizáveis no âmbito da dissertação. Estas soluções dividem-se em ferramentas para desenvolvimento e implementação e em *frameworks*/API de *software* referentes à tecnologia *multitoque*.

2.5.1 Frameworks multitoque

Windows 7 API

O Windows 7 é uma plataforma cujo padrão de interação difere das suas versões anteriores. No seu desenvolvimento foram criadas API que suportam o registo de vários pontos de contacto em simultâneo, permitindo o uso de multitoque e gestos como meio de interação (Microsoft 2011).

O tratamento dos gestos é feito recorrendo a objetos *WM_Gesture* que são passados pela arquitetura quando um padrão é detetado. Gestos básicos como *zoom, panning* ou mesmo de rotação são suportados por defeito pela plataforma sendo a interação realizada com um ou dois dedos (Kiriaty 2009). Existe também a possibilidade de estender os gestos e criar novos a partir do tratamento de objetos *WM_Touch,* que devolvem a informação sobre o respetivo contacto e pontos no qual foi detetado (Kiriaty 2010).

Além do multitoque foi também desenvolvido um motor de física, chamado *Inertia API,* que é aplicado às animações sobre objetos criando transições mais suaves e naturais ao utilizador. Também pode ser utilizado em conjunto com o *IManipulation Processor* para garantir que os objetos manipulados se mantenham dentro dos limites do écran.

Esta API apenas está disponível em Windows 7, com a versão 4 da plataforma de desenvolvimento Windows Presentation Foundation (WPF), não sendo compatível com versões anteriores.

Existe uma plataforma bastante semelhante desenvolvida pela Microsoft, o Surface SDK (para WPF 3.5). Foi desenvolvida para dar suporte ao produto Microsoft Surface Table. Os controlos de multitoque suportados estão também disponíveis na plataforma Windows Touch (Microsoft 2010). Há também que salientar que, apesar das duas plataformas serem bastante semelhantes, as aplicações que são desenvolvidas para Windows 7 não são compatíveis no Surface e o contrário também se verifica (Microsoft 2011).

Miria

Miria é uma *framework* de código aberto (licença GNU/GPL) para C#/WPF. Possui gestos pré-definidos para o utilizador, para além de controlos possíveis de serem adaptados a aplicações já existentes. Permite também que sejam criados novos gestos utilizando a deteção de coordenadas de contacto, caminho percorrido, ângulo, etc. Possui interligação com a plataforma *monoMig* que possibilita a integração com dispositivos como o *Wii Remote, Kinect Sensor* ou outros dispositivos multitoque do mesmo género (Generoso 2011).

Breeze

A *framework* Breeze é uma solução *open-source* com licença GNU/GPL que foi inicialmente desenvolvida para possibilitar interação multitoque em WPF 3.5 (atualmente já suporta também a versão 4.0) que não é possível por omissão. Possibilita o uso de gestos pré-definidos (*scale, rotate, panning*) aliados a um sistema de inércia que aumenta a fiabilidade das animações, tornando as transições mais suaves e naturais. É também possível expandir a plataforma criando novos padrões de gestos através da deteção das coordenadas de contacto.

Um aspeto que a distingue das outras *frameworks* descritas anteriormente é o de permitir a visualização da informação recebida em cada contacto e do centro de rotação nos gestos de *rotate, scale* e *move*.

2.5.2 Ferramentas de auxílio ao desenvolvimento

Visual Studio 2010

O *Visual Studio 2010* é um pacote de ferramentas para o desenvolvimento de *software* (IDE) para plataformas *Windows* (Windows 7, Windows Phone, etc.). Suporta diversas linguagens, entre as quais C# e permite a um desenvolvedor planear, implementar e testar as suas soluções no mesmo ambiente. As funcionalidades do IDE permitem simplificar todo o processo de desenvolvimento desde a fase de desenho à de instalação (Microsoft 2011). Para esse efeito conta com ferramentas de gestão de projeto, de teste e mesmo de desenho interfaces com o utilizador.

Microsoft Blend Studio

Microsoft Expression Blend é uma ferramenta integrada no pacote Microsoft Blend Studio e serve para planear e desenvolver interfaces gráficas para várias tecnologias (C#, VB, etc.). Permite separar o modelo de desenho do modelo de implementação (semelhante ao modelo MVC), aumentando a consistência das aplicações e minimizando a dificuldade de interação entre programadores e *designers*. Com esta ferramenta é relativamente fácil e rápido de criar protótipos de interfaces e de as tornar em algo mais concreto. Permite também a criação de *behaviours* (animações, comportamentos, etc.) sobre objetos sem recurso a qualquer linha de código (Microsoft 2011).

2.5.3 Escolhas tecnológicas

Em termos tecnológicos, foi necessário fazer uma escolha relativamente à *framework* a utilizar na implementação do projeto. É uma escolha de importância extrema, pois todo o projeto vai ser desenvolvido e implementado recorrendo a uma ferramenta que suporte gestos multitoque, sendo crítica a sua fiabilidade. Numa aplicação para o nível empresarial da Efacec, é obrigatório que o espaço para falhas e erros seja mínimo. Para esse efeito, devem-se selecionar as ferramentas cuja qualidade seja comprovada e que, acima de tudo, não falhem quando são necessárias. Para além deste fator, é necessário focar que um dos requisitos é o de ter compatibilidade total com a plataforma já existente, que foi criada em C# e WPF.

Essa escolha recaiu sobre a plataforma da Microsoft, Windows 7 API. Esta plataforma, permite uma total compatibilidade com a solução já existente, para além de ser uma *framework* bastante extensível. Existe, ainda, a possibilidade de criar novos gestos e novas formas de interação, utilizando as bibliotecas criadas para o efeito. Outra vantagem é a de não ser necessária a instalação de bibliotecas externas, pois esta API está disponível por omissão em qualquer instalação do sistema operativo Windows 7. A confiança na plataforma é total, não só

pela empresa que a produz, mas também pelo contínuo desenvolvimento a que está sujeita. Possui ainda uma extensa documentação que facilita em muito o trabalho do desenvolvedor. Este aspeto é diferenciador em relação às outras *frameworks* apresentadas que, salvo raras exceções, não possuíam documentação minimamente aprofundada.

Por todas estas razões, crê-se que a escolha foi acertada e que a *framework* representou um papel fundamental no desenrolar deste projeto.

Relativamente às ferramentas de suporte ao desenvolvimento, optou-se pela utilização da IDE Visual Studio 2010. É uma solução bastante sólida e adequada para o desenvolvimento de aplicações em C# e WPF, requisitos que suportam a qualidade que se pretendeu no projeto. Por outro lado, por forma a manter uma separação entre *design* da interface gráfica e o código, optou-se por utilizar a aplicação Expression Blend. Permite uma rápida alteração de elementos gráficos, facto que se torna relevante no *design* iterativo e prototipagem. Uma interface construída de forma separável permite um controlo de abstração, (Took 1990) isto é, torna-se capaz de receber e manipular as ações do utilizador sem envolver a aplicação (a nível gráfico). No fundo, a separação entre a interface e o código torna o desenvolvimento mais ágil e modular.

2.6 Resumo

Neste capítulo foram analisadas diversas áreas havendo uma base de sustentação importante para o planeamento e desenvolvimento do projeto. Foi estudado o processo de design de interação de forma a perceber qual a forma ideal de planear a lógica a implementar e como a planear. Analisaram-se as diversas metáforas de interface pesando as vantagens e desvantagens e a sua aplicabilidade no contexto das interfaces gráficas.

O multitoque foi alvo de um estudo aprofundado identificando-se as debilidades e propostas formas de as evitar. Este tema é de extrema importância para uma melhor perceção das limitações no processo de *design* de interação e para uma melhor perspetiva de solução. Num contexto mais específico foi analisada a interação no que diz respeito ao toque e aos gestos já alvos de estudo e aplicação em situações práticas. Pode-se concluir que os utilizadores preferem gestos de simples execução e que não envolvam grande esforço físico (de execução) nem esforço mental (de fixação).

A nível de *software* foram listadas as soluções HMI/SCADA existentes atualmente. Foram analisadas as suas características e modos de funcionamento, tendo sido alvo de maior análise a única solução que aplica tecnologia multitoque. Procedeu-se também ao estudo das *frameworks* multitoque de desenvolvimento para Windows (C#/WPF) e ao levantamento das aplicações que iriam ser necessárias durante o processo de estudo e desenvolvimento. Para cada uma foi feita uma análise crítica das suas características, vantagens e desvantagens.

Por fim foi descrito o processo de teste de usabilidade, descrevendo o seu objetivo, os diversos tipos existentes e o protocolo usual de teste.

Capítulo 3

3. Simulation.Touch: Multitoque em automação

3.1 Introdução

Como foi discutido e analisado no capítulo 2, o paradigma do multitoque tem aspetos positivos e negativos que devem ser ponderados, com cuidado, na sua utilização no contexto da Automação. O processo de desenvolvimento do Simulation Touch teve esses aspetos em conta, tendo sido efetuado um estudo e um planeamento minucioso. O objetivo fulcral era o de tentar não alterar significativamente o aspeto de uma aplicação de monitoria e controlo, adicionando-lhe comportamentos multitoque com a intenção de melhorar a própria experiência de utilização.

Foi efetuado um levantamento dos requisitos associados, sendo ordenados por grau de importância. De seguida, foram propostas soluções para preencher estes requisitos, estudadas as vantagens e limitações de cada uma e definida uma solução final.

3.2 Definição do problema

O objetivo principal foi o de desenvolver um interface alternativo para um módulo de uma aplicação já existente, no contexto de uma *suite* de software desenvolvida pela equipa de I&D da Efacec. O módulo existente (analisado no capítulo 2.4.1) é responsável por simular o comportamento de uma configuração criada no AS, utilizando como meio de interação rato e teclado.

Esta proposta tinha como objetivo substituir esse tipo de interação pela tecnologia multitoque, fazendo um aproveitamento correto das vantagens que este tipo de paradigma oferece. Pretendia-se, acima de tudo, desenvolver uma solução que mantivesse o mesmo tipo de

funcionalidades e possibilidades já existentes, reformulando-as. Foi também requisitada a adição de novas funcionalidades que pudessem tirar partido do que o multitoque pode oferecer, expandido as opções disponíveis no módulo de simulação.

A solução existente está integrada na *suite* AS cuja arquitetura está representada na Figura 10. Na proposta de resolução decidiu manter-se a mesma estrutura, sendo que o módulo de simulação existente seria substituído pelo Simulation.Touch.

No processo de implementação foi necessário criar novas formas de interagir e manipular recorrendo ao toque. Para manter a navegabilidade existente no módulo anterior foi necessário pensar numa solução de navegação com *zoom* através da utilização do multitoque. Foi também necessário criar processos alternativos de apoio ao utilizador, nomeadamente na criação de controlos visíveis na interface. Estes controlos permitem a execução de funcionalidades

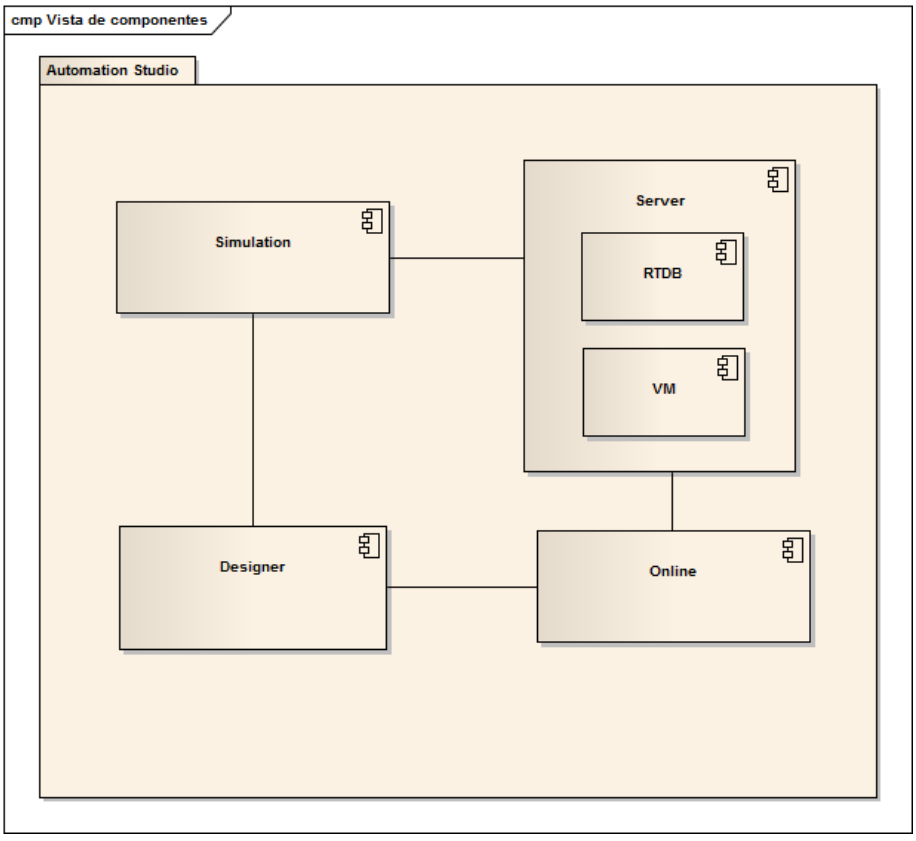


Figura 10 - Arquitetura da plataforma Automation Studio

avançadas da aplicação tal como a criação de múltiplas vistas de monitorização ou a inserção de notas textuais nas páginas.

3.2.1 Funcionalidades existentes

A solução desenvolvida teve como objetivo reformular funcionalidades já disponíveis no módulo de simulação do AS. Entre estas funcionalidades a mais importante é a de carregamento

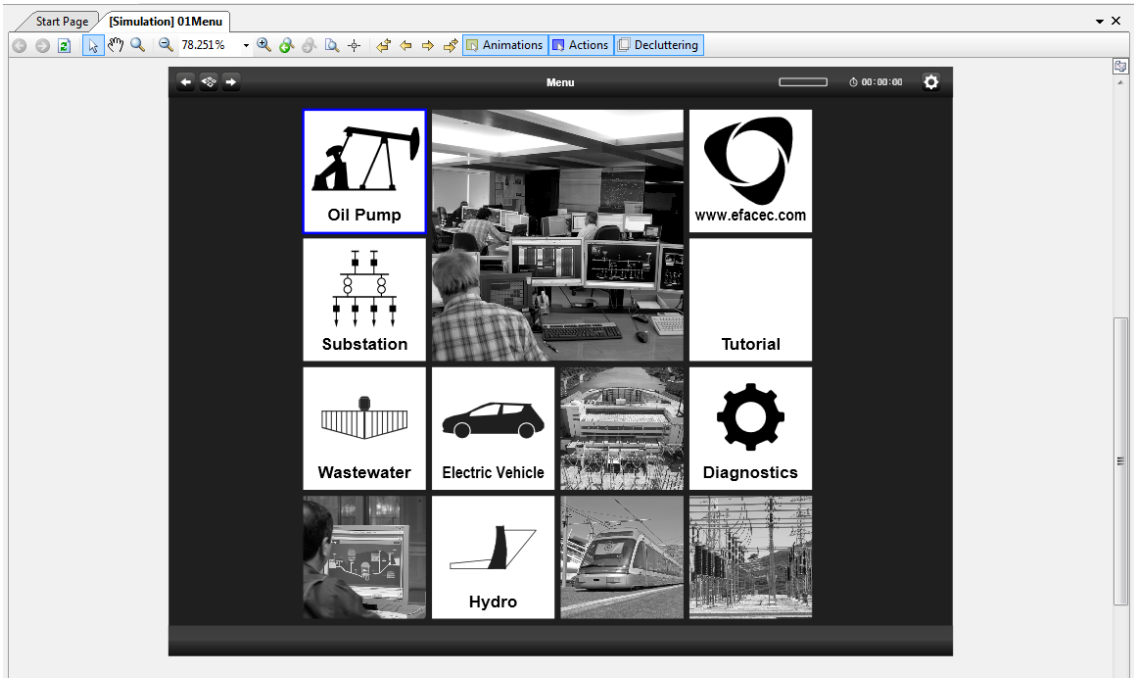


Figura 11 – Exemplo de página no modo de simulação

e interação com páginas dinâmicas XAML. Na Figura 11 pode-se observar um exemplo de uma página XAML com diversos tipos de objetos 2D (ex.: figuras geométricas, texto, imagens vetoriais, etc.). Estas páginas são criadas no módulo Designer do AS e são compostas por objetos vetoriais, cujo aspeto pode ser customizado. Alguns destes objetos podem possuir interatividades associadas, como por exemplo a navegação entre páginas ou a execução de controlos.

Para suportar a navegação nestas páginas, o módulo de simulação fornece uma barra de ferramentas (Figura 12) com diversos botões de ação. Entre as ações mais importantes podem-se destacar a de navegar para página seguinte/anterior, navegar para a primeira/última página, *zoom-in*/*zoom-out* e *zoom-to-fit*.

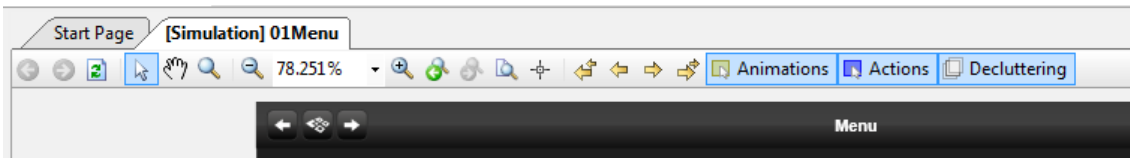


Figura 12 – Controlos de suporte à navegação no modo de simulação

A interação com configuração faz-se recorrendo a ações de execução de controlo e de escrita de valor (Figura 13 e Figura 14). A ação de execução controlo executa um determinado procedimento com um valor definido. Já a ação de escrita de variável escreve um determinado valor numa variável presente na base de dados. Estas ações podem ser silenciosas (executadas após o clique no objeto da página) ou não silenciosas (abre janela de confirmação representada na Figura 13 e Figura 14).

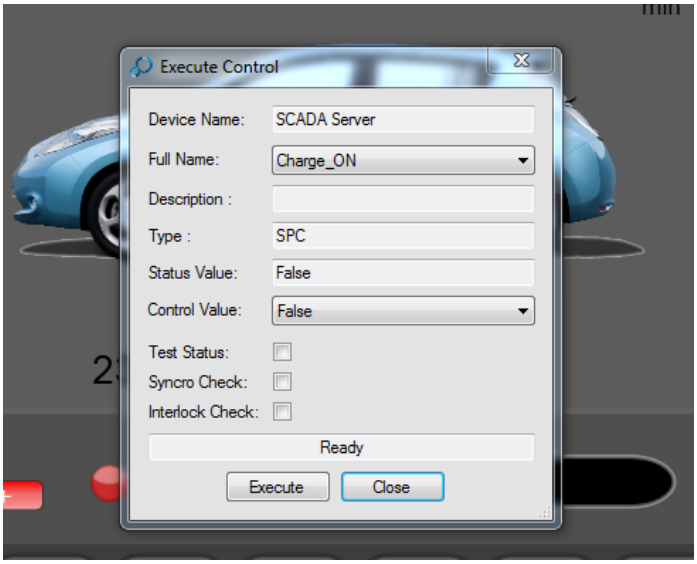


Figura 13 – Execução de controlo no modo de simulação

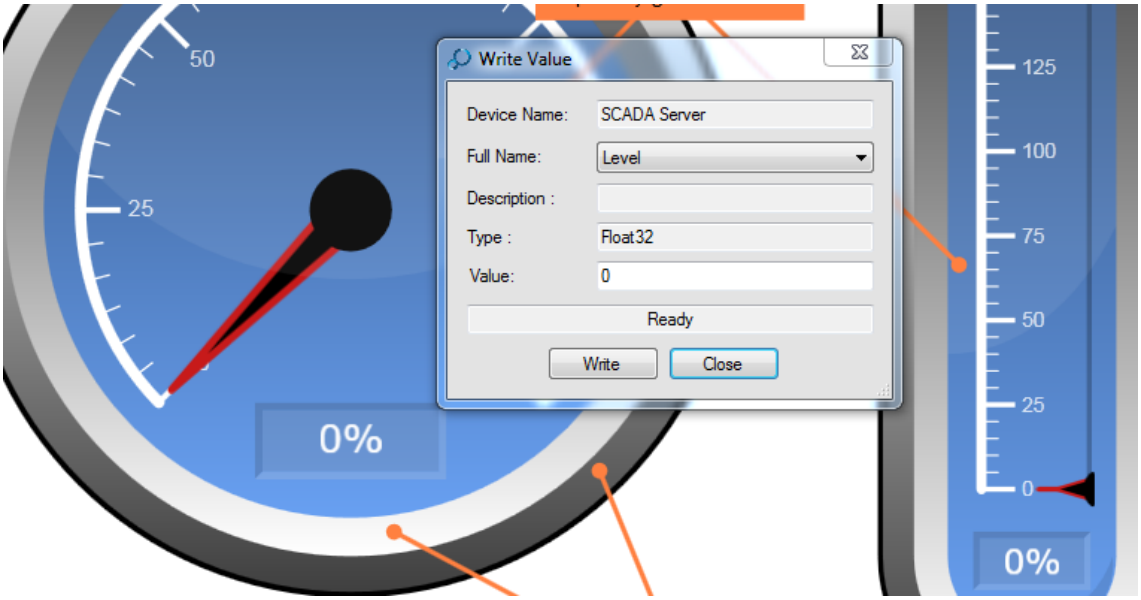


Figura 14 – Execução de escrita de valor em variável no modo de simulação

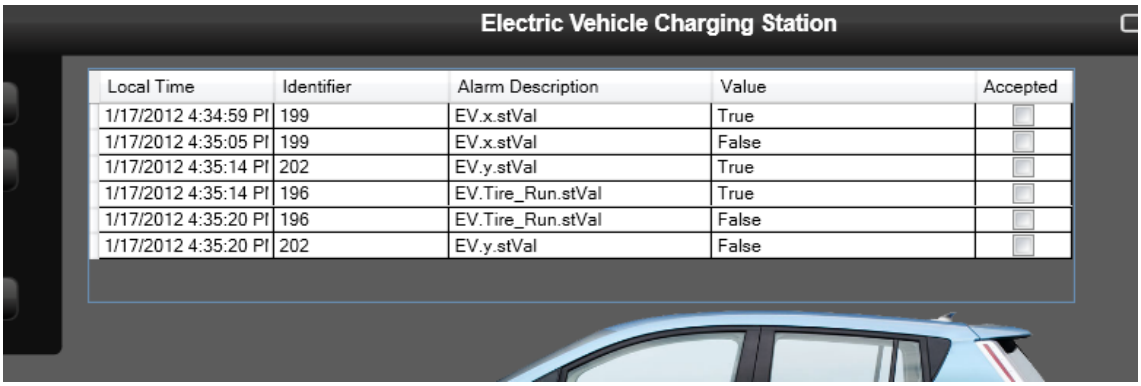


Figura 15 – Lista de alarmes em página no modo de simulação

Uma das funcionalidades que deve ser destacada é a de criação de listas de alarmes em páginas XAML (Figura 15). As listas são compostas por tabelas que são automaticamente preenchidas, no modo de simulação, quando surge um novo alarme, podendo este ser aceite. No projeto pretendeu-se expandir este conceito aplicando-lhe interação multitoque e adicionando novas opções de interação.

3.2.2 Requisitos

Os requisitos foram definidos com a equipa de desenvolvimento, tendo como base a solução já existente à qual se adicionaram novas funcionalidades, que se pretendiam suportar. Estes requisitos encontram-se sintetizados de seguida:

- Abertura e representação de páginas XAML
- Visualização e interação com controlos de página
- Criação de múltiplos ambientes de visualização
- Controlos de interface de suporte à navegação
- Controlos modulares configuráveis
- Controlo com listagem de alarmes
- Criação de controlos de execução e de escrita de variável
- Criação de controlos embebidos de página
- Introdução de notas textuais em páginas

Abertura e representação de páginas XAML

A principal funcionalidade a suportar era a de permitir a abertura e representação de páginas XAML, geradas pelo módulo Designer do AS. Estas páginas podem ser sequenciais

(relação linear entre elas) ou não-sequenciais. Este contexto implica que possa haver navegação entre páginas de forma linear (ação de navegar para a página seguinte ou página anterior) ou de forma não-linear (ação de substituição de uma página por outra). As páginas geradas pelo AS são dinâmicas e baseadas no paradigma ZOIL. A visualização dos objetos depende do nível de *zoom* atual, logo foi necessário repensar a forma como se apresenta a informação. É também importante que seja possível visualizar e interagir (no caso em que isso é permitido) com os controlos disponíveis nas páginas XAML.

Visualização e interação com controlos de página

Uma funcionalidade proposta no decorrer do projeto foi a de criação de múltiplos ambientes em visualização permanente, em que cada um possa carregar uma página distinta. Este requisito pretende fornecer ao utilizador a possibilidade de monitorar informação de diferentes páginas, ao mesmo tempo, ou de comparar informação relacionada.

Controlos de interface de suporte à navegação

Para suportar a navegação e a experiência de utilização foi necessário criar controlos de interface cuja indicação de existência, seja visualmente percetível, em qualquer momento. É necessário permitir que o utilizador consiga navegar para a página anterior ou seguinte, através de botões representados em controlos (Figura 16). Na área da navegação foi também solicitada a existência de um controlo que listasse todas as páginas que tivessem relação linear e sequencial. A existência deste componente permite que o utilizador não se sinta perdido e que consiga navegar para a página que deseja, de forma rápida e eficaz.

Figura 16 – Protótipo inicial da *Toolbar*

Controlos modulares configuráveis

A solução a desenvolver pretendeu-se que fosse versátil e configurável por parte do utilizador. Esta característica levou à criação de uma área fixa na interface em que seja possível carregar uma página desenvolvida no AS, aumentando as possibilidades de interação. Esta página pode-se comportar como um menu, como uma lista de informação de alarmes ou como uma tabela de visualização de estados, por exemplo. Este componente deve estar sempre acima de todos os outros componentes existentes na interface, e deve ser possível de o definir numa das 4 margens (cima, baixo, esquerda ou direita). Após abertura da aplicação, a sua dimensão e localização não é alterável. Com o mesmo objetivo foi definida a necessidade de criação de um controlo que também permita o carregamento de páginas mas que não se encontre sempre visível e possa ser escondido numa margem da interface. Deve, ainda assim, manter-se, com uma indicação visual permanente da sua existência, podendo ser aberto em qualquer altura.

Controlo com listagem de alarmes

Quando um objeto numa página sofre uma alteração de estado, é disparado um alarme no sistema que deve ser possível visualizar. Para isso, é necessário criar uma área num controlo em que seja possível visualizar todos os alarmes existentes na solução, ou em cada página, podendo ser ordenados pela data em que foram criados, pelo nome do evento, ou pelo estado de aceite ou não aceite. Deve também ser possível selecionar múltiplos alarmes de cada vez e executar a ação de os aceitar. Este componente deve lançar um sinal sonoro e visual da existência de um novo alarme disponível, e deve ser possível remover o alarme visual sem ser obrigatório abrir a lista de alarmes. Esta informação deve estar inserida num controlo de visualização permanente.

Criação de controlos de execução e de escrita de variável

Os objetos gráficos nas páginas podem conter diversos tipos de ações associados em simultâneo. Estas ações devem ser listadas e deve ser permitido ao utilizador selecionar a que deseja executar. Quando executa uma destas ações (exceto a de navegação), é necessário permitir que o utilizador verifique a informação associada e que confirme a execução. No caso de ser uma ação do tipo escrita de variável (Figura 17), em que esta é do tipo numérico, há necessidade de criar um controlo apenas com teclado numérico para mais fácil inserção de valores. As ações de navegação podem ter uma particularidade que é a de uma página poder ser aberta num controlo que se encontra num nível superior ao da página, na qual é realizada a interação. Com o mesmo comportamento deve ser também possível abrir uma lista de alarmes num controlo deste tipo.

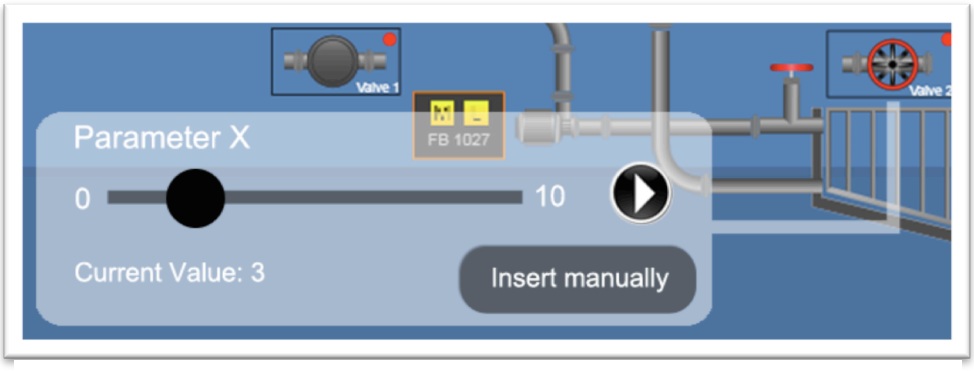


Figura 17 – Protótipo não funcional de um *Popup* de regulação de valor

Criação de controlos embebidos de página

Ainda no sentido de manter a aplicação versátil e de aumentar a liberdade de quem a configura, foi pensada a criação de comportamentos que possam ser atribuídos aos objetos desenhados nas páginas, de forma a torná-los dinâmicos e manipuláveis. Entre os comportamentos pensados destaca-se o de manipulação de variáveis numéricas discretas e contínuas em objetos lineares ou rotacionais. Os comportamentos referidos podem executar o comando de escrita apenas quando se interrompe o contacto (escreve quando o controlo é

largado), ou durante toda a manipulação. Para ajudar o utilizador podem ser criados botões com função de incremento ou decremento de valor de variável. Além destes, deve-se permitir também a atribuição de comportamentos em que o objeto possa estar em dois estados possíveis, comportando-se como uma variável booleana. Estes controlos devem poder solicitar ou não uma confirmação ao utilizador antes de realizarem a execução do comando associado.

Introdução de notas textuais em páginas

Uma funcionalidade avançada, que se pretendia adicionar, era poder criar uma nota textual, em qualquer localização de uma página já aberta. Desta forma, um operador pode adicionar um comentário a um objeto para visualização futura, ou, pode deixar uma nota de aviso para quem for utilizar a plataforma. Estas notas devem poder ser visualizadas, ou mesmo removidas, em qualquer altura.

3.3 Solução

Após análise e definição dos requisitos passou-se à fase de tomada de decisões, tanto em termos de arquitetura, como em termos de estrutura e desenvolvimento. Foi decidido dividir a implementação em quatro módulos principais: Interface, Media, Interações e Manipulações e finalmente o módulo dos User Controls (Figura 18).

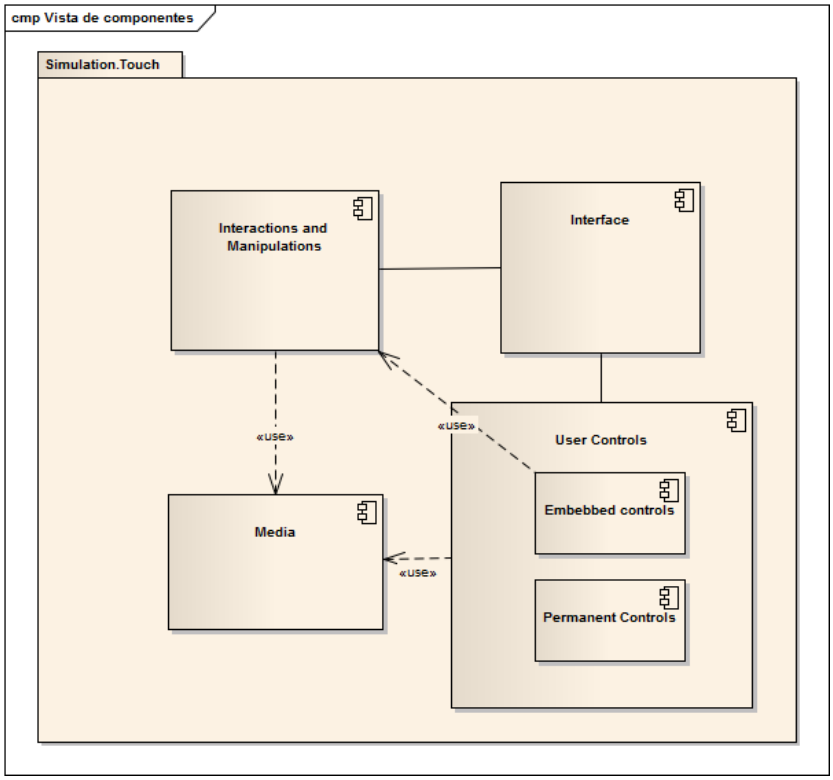


Figura 18 - Arquitetura da solução Simulation.Touch

3.3.1 Interface

O módulo Interface é responsável pela gestão das áreas de trabalho e da interação com as páginas. É através dele que são criadas múltiplas áreas de trabalho, sendo que cada uma contém um *Container* com ações e interações já definidas. Em qualquer momento só pode existir uma área de trabalho com foco, sendo representada com um bordo azul-claro. É neste controlo que são colocados os User Controls, havendo uma distinção em relação à ordem vertical, na qual são dispostos.

3.3.2 User Controls

Os User Controls que são inseridos na Interface são geridos num módulo que é responsável pela sua interação e pela execução das funcionalidades de cada um. Entre estes controlos podemos encontrar os que têm representação permanente e os embebidos em página.

Controlos de interface

Os controlos de interface possuem um botão que representa a sua localização que, quando clicado, abre o respetivo controlo, enquanto que os embebidos se encontram visíveis na respetiva página. Este botão também responde quando arrastado no sentido de abrir o controlo, efetuando a mesma ação. Entre estes controlos podemos encontrar:

- Alarm List – controlo que possui uma lista dos alarmes existentes na configuração atual, sendo possível de os aceitar após visualização. É também possível filtrar por página ou ordenar por nome, data ou estado. Quando existe num novo alarme disponível, é lançado um aviso sonoro e visual.
- Toolbar – controlo que também faz parte dos controlos de interface com representação visível da sua localização. Pretende ajudar o utilizador no processo de navegação e exploração da interface através de diversos botões de atalho.
- Page Drawer – permite a criação de uma área onde pode ser carregada uma página XAML com informação a monitorar, menus de navegação ou qualquer tipo de interação desejada. É configurável por parte do operador responsável pela criação da configuração do sistema.
- Page Navigator – listagem vertical das páginas com sequência linear, sendo que o clique numa determinada página, carrega-a para a área de trabalho que tiver com foco no momento. É possível manipular também as páginas recorrendo ao *drag* da imagem respetiva de cada uma.
- Inteface Bar – barra permanentemente visível na interface que se encontra no topo da hierarquia vertical (nenhum objeto a pode sobrepor). Este controlo é

semelhante ao *Page Drawer*, dado que permite carregar uma página XAML para a sua área de trabalho, embora não seja possível escondê-lo.

Controlos embebidos

Os controlos embebidos em página são um meio útil para configuração de valores numéricos ou para alteração de estado de uma variável ou controlo. São inseridos no momento de criação das páginas no Designer do AS, através da atribuição de um tipo a um objeto representado. Os controlos embebidos são os seguintes:

- Toggle – permite tornar qualquer objeto 2D XAML num manípulo booleano associado à execução de um controlo. O seu valor é alterado através do toque ou pelo arrastar do dedo do utilizador podendo ser solicitada uma confirmação.
- Slider – transforma um objeto 2D num controlo de configuração de valor. Pode ser do tipo contínuo ou discreto (com passos de *n* unidades) dependendo das definições atribuídas na configuração. O valor pode ser escrito no momento em que o utilizador está a deslizar o objeto ou apenas no momento em que o larga, podendo ser solicitada confirmação de execução.
- Page Handler – funciona como um botão responsável por abrir um *Popup* com uma área de trabalho embutida. Nesta área pode ser carregada uma página com informação, sendo que toda a interação funciona de forma natural, como se se tratasse de uma área de trabalho normal.
- Alarm/Event List – controlo que lista os alarmes disponíveis na configuração, sendo atualizada em tempo real. Possui ações de filtro e de confirmação, sendo semelhante ao controlo *Alarm List* da Interface. Difere no facto de poder ser inserido numa página em vez de se encontrar como controlo de Interface.

Controlos Mistos

Os controlos referidos nesta secção possuem características tanto de controlos de interface como de controlos embebidos, não podendo portanto ser enquadrados em nenhuma destas.

- Popup – o controlo *Popup* é um componente móvel na aplicação que é aberto através de uma ação associada a um objeto de uma página, quando este é clicado. Este controlo encontra-se num nível acima dos ambientes de página e abaixo dos controlos permanentes, podendo conter uma linha a indicar o objeto, a partir do qual este foi criado. Pode ser utilizado em diversos contextos:
  - Para confirmação de controlos de execução ou de escrita de variável;
  - Para carregar áreas de trabalho com páginas, permitindo as interações já inerentes à própria página;
  - Para carregar listas de alarmes ou listas de eventos;

  - Para inserir ou visualizar notas numa página, através de um campo de texto.

- Keypad – o controlo *Keypad* serve para inserir valores numéricos inteiros ou decimais, numa execução de um controlo de escrita de variável. Surge sempre associado a um *Popup* com o mesmo tipo de execução. Caracteriza-se por ter representado botões com algarismos de 0 a 9, para além de uma vírgula de separação.

3.3.3 Media

Este módulo é responsável pelas interações e animações utilizáveis nos objetos e nos controlos do Projeto. Tira partido das bibliotecas de manipulação e animação do WPF atuando como uma *framework* de comportamentos e eventos. Entre estes podemos identificar as rotações, translações e escalamentos, bem como os efeitos de elástico e animação associados à manipulação com o toque.

3.4 Resumo

No processo de planeamento do projeto foram definidos diversos requisitos que a aplicação deve suportar. A maior parte desta lista provém do módulo já existente, ao qual foram acrescentadas novas funcionalidades, tirando partido das vantagens que o multitoque oferece. Um exemplo claro é o uso de comportamentos de manipulação associados ao deslizar do toque, em vez de utilização do rato para deslizar um *Slider*.

Foi definida a versatilidade como característica fundamental e transversal a todo o projeto. Pretende-se que os componentes e comportamentos possam ser configuráveis e reutilizáveis, tanto pelo utilizador final bem como pela pessoa responsável pela configuração dos projetos. A navegação e exploração da interface é uma área crucial a que se deve dar bastante importância no desenvolvimento do projeto.

Foi decidido durante o planeamento que deviam ser desenvolvidos controlos de suporte ao utilizador para facilitar o processo de navegação e exploração, entre os quais o *Page Navigator*, *Toolbar*, *Page Drawer* e *Interface Bar*. A nível da solução foram também planeadas interatividades que se associam a objetos nas páginas, atuando como controlos embebidos. Desta forma, o projeto torna-se modular e configurável, aumentando as possibilidades de quem o configura e opera.

Capítulo 4

4. Implementação

4.1 Sistema

A solução desenvolvida é executada no contexto da plataforma AS. Nesta plataforma é possível criar uma configuração para vários *devices* (equipamentos), entre as quais, uma personalizada para este projeto. Esta configuração permite a criação de páginas XAML com objetos 2D e associação de interatividades. As interatividades associadas manipulam variáveis

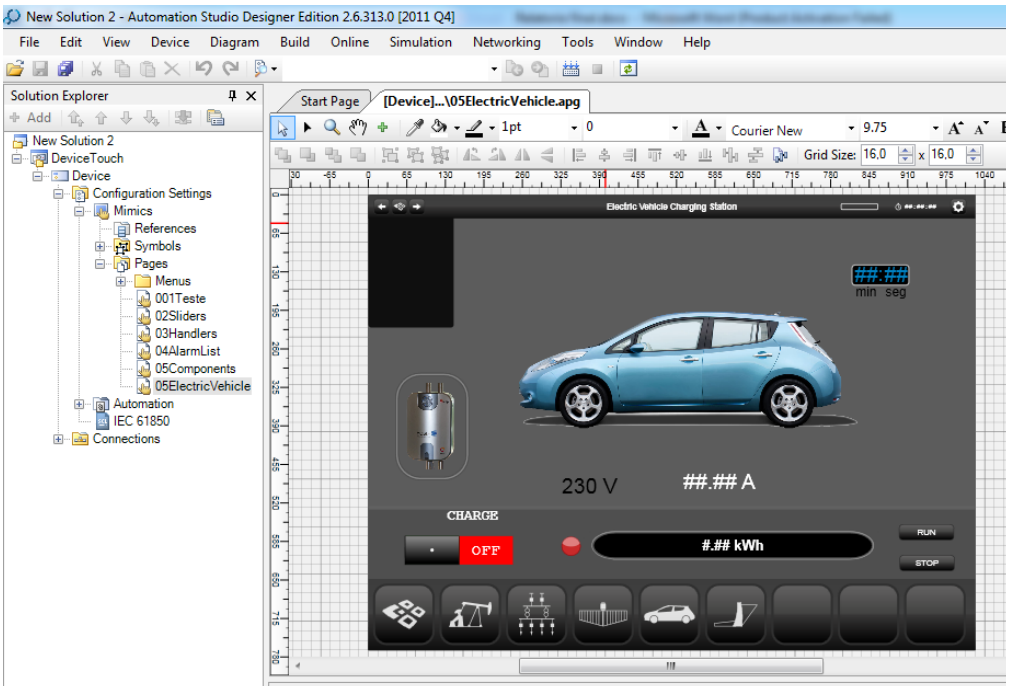


Figura 19 - Exemplo da criação de um Device Touch com páginas associadas

existentes numa base de dados de controlos de automação, que também é criada na plataforma. Podem ser do tipo de execução, incremento de valor, ou controlo.

Um processo normal de utilização do sistema envolve, numa primeira fase, a criação de um *device* no configurador (Figura 19). De seguida, essa configuração é compilada no AS e finalmente carregada no módulo Simulation.Touch. Caso seja necessário alterar algo na configuração, é necessário compilar novamente.

4.1.1 Arquitetura

Atores

No processo de especificação da aplicação foram identificados dois atores distintos que podem utilizar esta aplicação. O primeiro é o Quadro de Configuração responsável pelo desenho e configuração da solução e o segundo é o Operador que executa as tarefas na máquina.

O primeiro a ser referido é responsável por criar uma configuração base para o *device* respetivo, tratando de todo o aspeto visual e lógica inerente de execução de ações e controlos. É ainda responsável pela criação das variáveis e comandos de execução de controlos, bem como de procedimentos de automação. Utiliza normalmente a aplicação num contexto de teste em que entra no módulo de simulação, apenas para testar a configuração que desenvolveu, alterando-a e testando novamente, caso a resposta não seja a esperada.

O operador no terreno é responsável por execução das tarefas na máquina e pela utilização da aplicação para controlo. Observa os dados que vão sendo mostrados e caso seja necessário, altera algum valor ou controlo. Enquanto que o primeiro ator utiliza a plataforma num processo de teste do que está a desenvolver, este segundo ator utiliza-a num contexto crítico de monitorização.

Diagrama de classes

Por uma questão de organização a aplicação foi separada em pacotes e dentro de cada pacote existem diversas classes distintas (Figura 20).

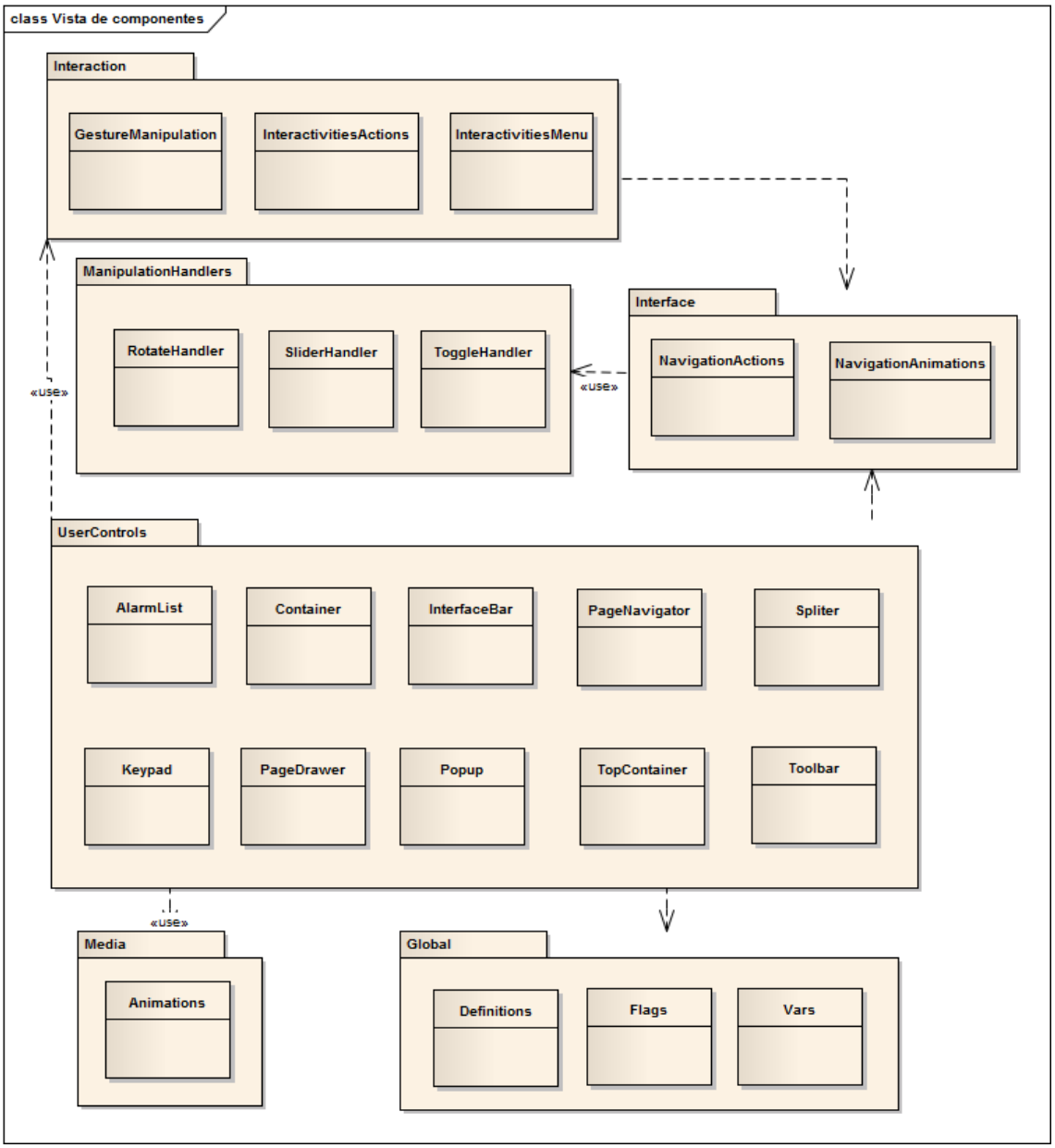


Figura 20 – Diagrama de classes do Simulation.Touch

As tabelas 1 a 6 identificam e descrevem cada uma das classes:

Pacote	Classe	Descrição
Global	Definitions	Classe estática onde se encontram as definições principais da aplicação (ex.: caminhos de solução, definição de tempos de fecho de controlos, etc.).
	Flags	Classe estática onde se podem encontrar as variáveis booleanas de controlo (ex.: Page Navigator aberto, Controlos todos fechados, etc.).
	Vars	Classe estática onde estão guardadas as variáveis estáticas partilhadas pela aplicação (ex.: lista de Popups abertos, lista de páginas carregadas em memória, etc.).

Tabela 1 – Classe Global

Pacote	Classe	Descrição
Interaction	GestureManipulation	Classe estática que é adicionada nos eventos de manipulação (ManipulationStarting, ManipulationDelta e ManipulationComplete) e que é responsável por reconhecimento de toque e gestos (ex.: tap, double-tap, swipe, etc.).
	InteractivitiesActions	Classe estática responsável pela implementação de controlos de execução e de escrita de variável. É responsável pela ligação ao servidor e escrita do novo valor.
	InteractivitiesMenu	Classe estática adicionada nos eventos de manipulação dos objetos com controlos de execução e escrita de variável. Cria um menu nos quais são listadas as ações disponíveis num objeto (execução de controlo, escrita de variável ou navegação).

Tabela 2 – Classe Interaction

Pacote	Classe	Descrição
Media	Animations	Classe estática responsável pela execução de animações (rotação, translação, escalamento, etc.) aplicáveis a objetos de página.

Tabela 3 – Classe Media

Pacote	Classe	Descrição
ManipulationHandlers	RotateHandler	Classe responsável pela implementação de comportamentos de rotação em objetos existentes nas páginas.
	SliderHandler	Classe responsável pela implementação de comportamentos de translação em objetos existentes nas páginas.
	ToggleHandler	Classe responsável pela implementação de comportamentos de manípulo binário em objetos existentes nas páginas. Semelhante ao Toggle Button de WPF.

Tabela 4 – Classe ManipulationHandlers

Pacote	Classe	Descrição
Interface	NavigationActions	Classe responsável pela implementação de ações de navegação entre páginas (ex.: navegar para página seguinte/anterior, navegar para determinada página, etc.).
	NavigationAnimations	Classe responsável pelas animações nas ações de navegação (ação de navegar para página seguinte/anterior ou de substituir página).

Tabela 5 – Classe Interface

Pacote	Classe	Descrição
UserControls	AlarmList	Controlo lista de alarmes.
	Container	Controlo *Container* responsável pelo carregamento das páginas.
	InterfaceBar	Controlo lateral configurável com possibilidade de carregamento de página.
	PageNavigator	Controlo lateral com lista de páginas.
	Splitter	Controlo responsável pela separação em vários ambientes de trabalho.
	Keypad	Teclado numérico.
	PageDrawer	Controlo lateral com possibilidade de carregar página.
	Popup	Controlo flutuante de confirmação de ações, criação de notas ou carregamento de páginas.
	TopContainer	Controlo de gestão da camada superior de manipulação onde se encontram os UserControls.
	Toolbar	Barra de apoio à navegação com opções de utilização avançadas.

Tabela 6 – Classe UserControls

4.1.2 Prototipagem e validação

O processo de prototipagem e validação decorreu de acordo com a metodologia de *interactive design*, no qual existiu constante acompanhamento e participação dos utilizadores. Foram estudadas as suas necessidades e desejos, tendo sido definidos requisitos funcionais e não funcionais para alinhavar o processo de prototipagem. As reuniões formais decorreram com regularidade, havendo apresentação de ideias e protótipos criados para o efeito. Os utilizadores davam a sua opinião sobre os protótipos, fazendo propostas sempre que se justificasse. Informalmente houve um constante acompanhamento da equipa do departamento de I&D e de Marketing no processo de desenvolvimento da interface através da sugestão de ideias e de emissão de opiniões.

A interface foi ganhando forma através de um processo iterativo de desenho. Numa

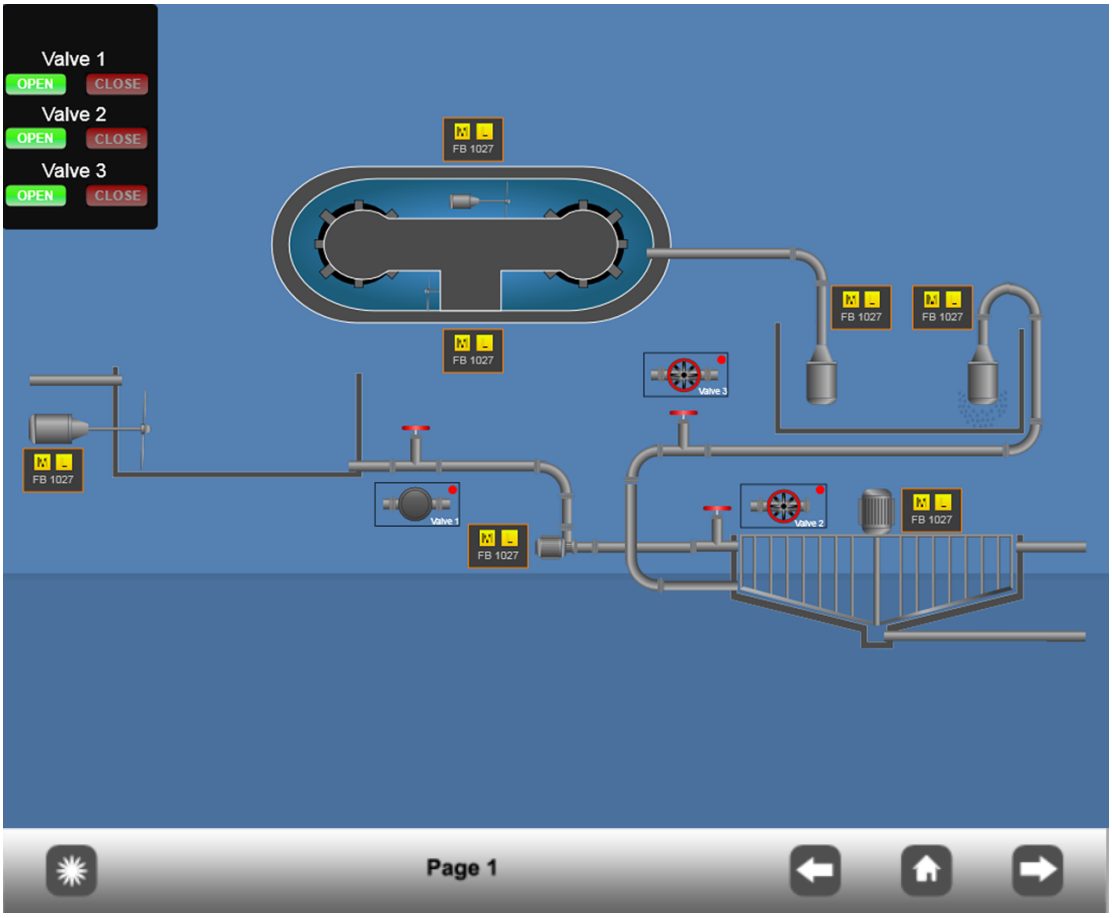


Figura 21 - Protótipo não-funcional

primeira fase foram desenvolvidos protótipos esquemáticos em papel, com o objetivo de transmitir os objetivos principais de cada componente e da interface. Estes protótipos de baixo nível (Figura 21) são fulcrais por envolverem pouco custo (monetário e em tempo), e por conseguirem apresentar a interface tal como ela foi pensada, em temos de dimensões e posições de componentes.

Numa segunda fase foram desenvolvidos protótipos em WPF já com interatividade e animações aproximando-se do comportamento final (Figura 22). Nesta fase não houve grandes preocupações a nível estético, tendo sido avaliado apenas o comportamento e a função de cada objeto desenvolvido. Esta fase foi proveitosa quanto à tomada de decisões finais no projeto, pois deu origem à descoberta de certos requisitos que não foram inicialmente identificados. Por exemplo, o controlo *Page Drawer* surgiu a partir de um protótipo funcional de um *Popup* que podia ser colado nas margens da interface e que permitia carregar páginas. Esta proposta proporcionou um novo requisito, que por sua vez, originou um novo controlo. Numa fase final, a solução foi aprimorada em relação à interação com toque e ao aspeto estético. Foi necessário criar uma solução consistente em termos de interação gestual e visual.

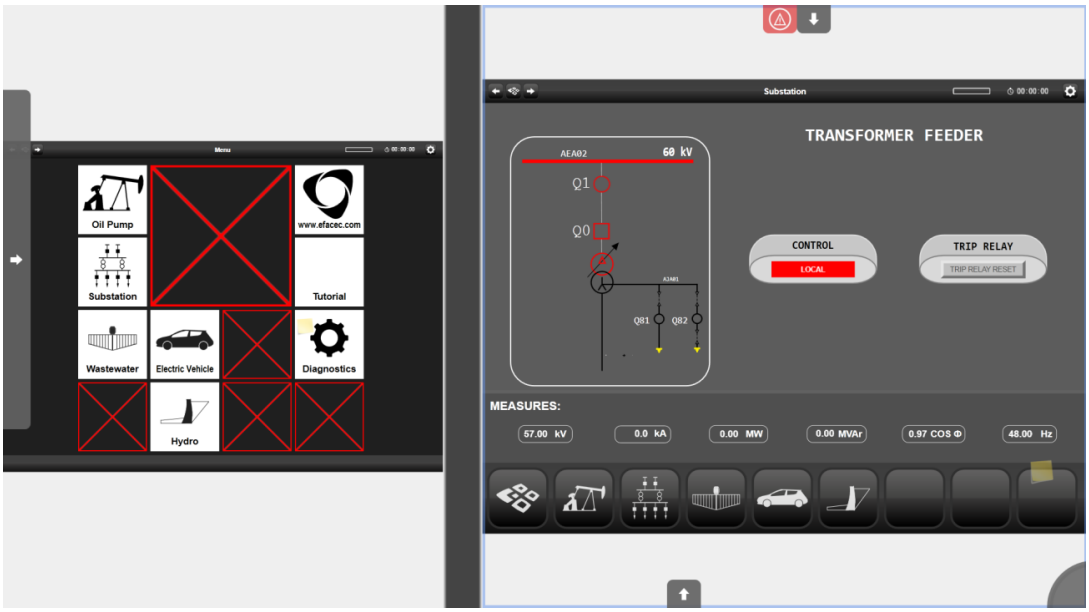


Figura 22 - Exemplo de protótipo funcional

4.1.3 Processo de desenvolvimento

O processo de desenvolvimento decorreu de acordo com a metodologia SCRUM, havendo uma separação em iterações dos componentes a integrar. Cada iteração caracterizou-se por uma fase de arquitetura, seguida de implementação e, finalmente, uma última de testes e correção e/ou validação. Separou-se o desenvolvimento em três iterações distintas que se descrevem de seguida:

- 1ª Iteração – navegação e exploração
  - Criação de áreas de trabalho com carregamento e exploração de páginas;
  - Desenvolvimento de criação de múltiplas áreas de trabalho;
  - Desenvolvimento das lógicas de interação;
  - Implementação do controlo por gestos e toque.
- 2ª Iteração – controlos de suporte ao utilizador
  - Desenvolvimento de controlos de apoio à navegação e à utilização do sistema
  - Criação de menus com listagem de interatividades (navegação, execução e escrita de variáveis)
  - Criação do módulo de Media e desenvolvimento de animações
- 3ª Iteração – controlos de execução e controlos de página
  - Execução de controlos e escrita de variável
  - Criação de controlos de página
  - Implementação de *handlers* de manipulação

No final de cada iteração os componentes desenvolvidos eram testados a nível de integração com a solução já existente. O facto de o desenvolvimento ter sido dividido em três fases distintas, ajudou a agilizar o processo e a mantê-lo consistente em relação ao inicialmente delineado.

4.2 Interface

A interface foi desenvolvida de forma a ser modular e facilmente compreensível pelo utilizador. É composta por diversas camadas sobrepostas, havendo uma lógica inerente a esta organização. Inicialmente, é apresentada a primeira página da configuração carregada (usualmente a página Menu, caso exista) com as etiquetas dos User Controls visíveis e focadas.

4.2.1 Diagrama vertical da interface

A interface está dividida em vários planos horizontais sendo que cada um tem diversos controlos e componentes associados. Na Figura 23 pode-se verificar que o plano inferior se

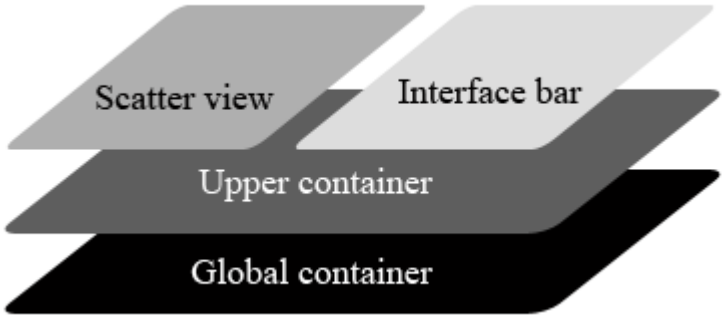


Figura 23 – Diagrama vertical da interface

chama *Global Container*. Neste índice estão concentrados os controlos do tipo *Container* e os *Splitviews* criados e associados. No *Upper container* estão concentrados os controlos de interface *Toolbar*, *AlarmList*, *PageNavigator*, *PageDrawer* e *Keypad*, estando por isso num nível superior aos *Containers*. No primeiro nível encontra-se um plano *Scatter view* onde são inseridos os *Popups* que são criados e um outro plano onde se encontra a *Interface Bar*. Esta divisão em planos permite sintetizar a ordem pela qual os conteúdos são disponibilizados e inferir uma importância relativa em relação à oclusão de componentes pelos outros.

4.2.2 Lógica da interface

A interface foi planeada de forma a que o foco estivesse direcionado para a visualização de informação, evitando a oclusão desta. Assim sendo, foi decidido apresentar de início um

Container a ocupar a área total da interface. Neste *Container* é carregada uma página XAML (em geral a página Menu caso exista), havendo margens laterais acinzentadas. De forma a não perturbar a visualização do estado do sistema, decidiu-se que os controlos inicialmente deveriam estar fechados e com opacidade mínima.

Para haver uma melhor forma de monitorizar a informação da solução decidiu-se permitir a criação de múltiplos ambientes de trabalho. Cada *Container* pode conter uma página necessariamente diferente, apesar de que, quando um novo é criado, não se encontra com nenhuma página.

Container

O *Container* (Figura 24) é o controlo que tem maior importância na aplicação, dado que é onde toda a interação decorre. Neste controlo são carregadas páginas dinâmicas XAML com as quais é possível interagir e manipular, sendo o resultado visível no imediato. Por forma a tornar a experiência de navegação mais coerente, recorreu-se ao paradigma de *zoomable interface*. Este paradigma promove a apresentação de informação tendo em conta o nível de zoom atual, estando o nível de detalhe dependente do zoom efetuado. Para navegação recorre-se à utilização do gesto *pan,* para mover na horizontal ou vertical e o *pinch-to-zoom* para fazer navegação por *zoom* e translação ao mesmo tempo. A escolha deste tipo de interação deveu-se à tentativa de tornar a interação o mais simples possível utilizando tecnologia multitoque, para além de seguir o tipo de interação já utilizada em sistemas semelhantes (ver capítulo 2.3.3). Pretende-se também fornecer ao utilizador o poder de manipulação sobre a informação de forma direta, não necessitando de utilizar controlos para explorar a interface.

Este controlo é modular, sendo portanto possível de ser inserido noutro controlo desenvolvido (ex.: *PageDrawer*, *Popup*, *BarMenu*, etc.). Inicialmente foi pensado permitir apenas a utilização de um objeto deste tipo. No processo de prototipagem foi introduzido um novo requisito para permitir a comparação entre páginas e nesse sentido foi criada uma forma de abrir vários controlos do tipo *Container*.

Quando existe mais do que um objeto deste tipo, e se as ações dos restantes controlos, incidem sobre objetos *Container*, refletem-se no que estiver com foco. Assume-se que um *Container* obtém foco quando recebe ação do toque do utilizador, sendo que não é possível haver mais do que um objeto deste tipo com foco. Esta decisão deveu-se ao facto de a utilização ser planeada e pensada para modo de interação monoutilizador. Este estado é representado por um bordo de cor azul clara a toda a volta do *Container*.

Ações de navegação de páginas

A ação de mudança de página pode ser efetuada recorrendo a dois paradigmas distintos. O primeiro recorre ao conceito de substituição de página, no qual uma página não possui relação direta com a que a vai substituir. Neste caso, é executada uma animação própria na qual a

Figura 24 - Exemplo de um *Container*

página através de *fade-in* é sobreposta na já existente. O segundo paradigma diz respeito a uma ação através da qual as páginas se encontram relacionadas por uma ordem linear. A animação, neste tipo de mudança de página, simula uma pilha na qual as páginas se encontram ordenadas. Esta diferença de comportamento pretende induzir no utilizador a perceção da existência de páginas que se encontram encadeadas e de outras que não têm qualquer tipo de ordenação.

A navegação para a página anterior ou seguinte, pode ser feita recorrendo a gestos de *swipe* que comecem na margem direita da interface, na horizontal (direita-esquerda), para navegar para a página seguinte; ou na margem esquerda (esquerda-direita), para navegar para a página anterior (ações também disponíveis como atalhos no controlo *Toolbar*). Estas interações foram introduzidas para permitir, ao utilizador, navegar para a página seguinte ou anterior, de forma rápida e de fácil execução. Houve também a intenção de transmitir ao utilizador um sentido de continuidade, no qual ele pudesse manipular a própria navegação das páginas, melhorando a experiência de utilização. Os gestos utilizados pretendem não envolver demasiado esforço cognitivo, nem serem difíceis de memorizar.

Splitview

Para se criarem vários *Containers* recorre-se à utilização de um *Splitview* (Figura 26). Este controlo é ativado através do ato de *press-and-hold* de dois dedos na margem direita da interface (ou clique no respetivo atalho na *Toolbar*) até aparecer uma barra vertical. Essa barra é manipulada através do deslizar do dedo. Caso se utilizem dois dedos na manipulação, surge uma barra vertical que permite a criação de *Splitviews* horizontais (Figura 25). Esse comportamento permite criar *Containers* verticais e horizontais com diversas páginas (cada *Splitview* adicionado

dá origem a um novo *Container*), aumentando a capacidade de análise e monitoria do utilizador. Caso se deseje remover algum *Container*, basta deslizar a barra até à extremidade da interface.

Este controlo pretende permitir que o utilizador possa comparar e monitorar diferentes páginas que podem ou não ter relação entre si. O *Splitview* foi desenvolvido de forma a ocupar o mínimo de espaço possível do écran, mantendo as dimensões mínimas para não haver ambiguidade no toque (cerca de 60px).

Escolheu-se um toque com um dedo para movimentar os *Splitviews* e dois dedos para criação de novos (horizontal ou vertical), de forma a manter a consistência na interação. O

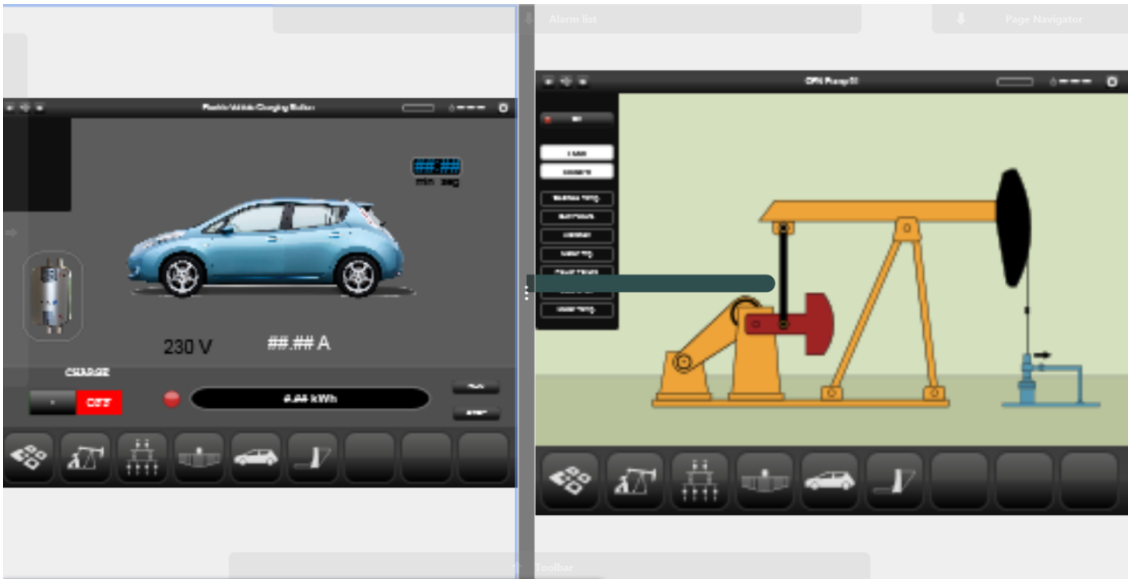


Figura 25 – Criação de um *Splitview* horizontal a partir de um vertical

utilizador, quando começa a movimentar um *Splitview*, este muda de tonalidade, passando a ter propriedades de transparência para o informar que este foi selecionado e está a ser alvo de interação. Com o mesmo objetivo, quando um *Splitview* é movimentado, os *Containers* que dependem dele são atualizados no que diz respeito ao tamanho que possuem. Esta forma de proceder fornece ao utilizador uma visão imediata do tamanho com que cada *Container* vai ficar no final do processo. Quando se cria um *Splitview* horizontal, existe a visualização de um *Splitview* virtual (transparência) que muda de cor, para uma tonalidade mais escura; quando é ultrapassada uma determinada dimensão, indica ao utilizador que, ao largá-lo, este vai ser criado. Esta decisão pretende cobrir os casos de um *Splitview* ser criado por engano, ou caso o utilizador tenha iniciado o gesto de criação mas o deseje cancelar. O mesmo acontece quando se pretende remover um *Splitview* arrastando para a extremidade da interface. A barra passa a ter propriedade de transparência quase total, para transmitir a ideia do desaparecimento da mesma, caso termine o contacto.

Optou-se por remover na extremidade em vez de um gesto de *press-and-hold* ou *double-tap*, pois permite uma ação mais rápida e menos ambígua para o utilizador. A criação de um *Splitview* horizontal pode ser também efetuada recorrendo a um atalho na *Toolbar*, caso o utilizador prefira utilizar uma forma mais explícita de criar um *Splitview*.

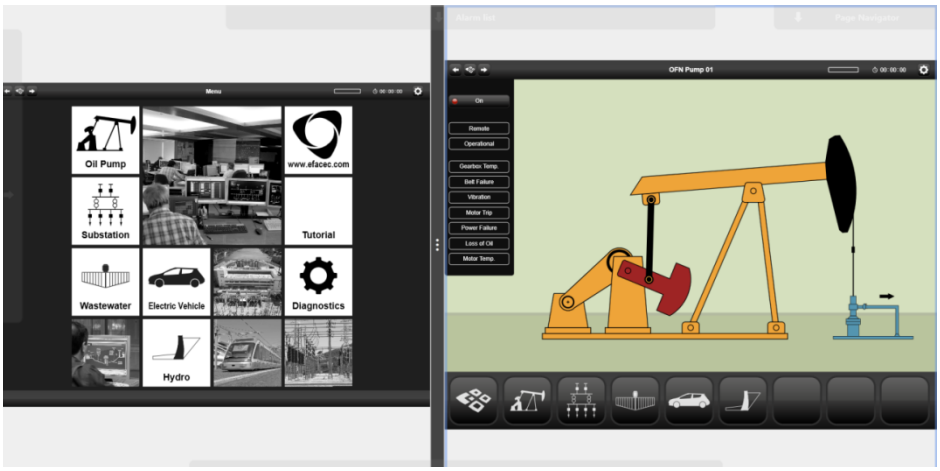


Figura 26 – *Splitview* vertical

4.2.3 Modelo de interação

Um dos requisitos da aplicação era o de utilizar os paradigmas do multitoque de forma a melhorar a experiência de utilização numa aplicação profissional. Com esse objetivo definiu-se como requisitos utilizar uma biblioteca de gestos que fosse simples de usar e de aprender. Foi também decidido utilizar gestos que já sejam utilizados em aplicações atuais (telemóveis, *tablets*, etc.) com objetivos semelhantes, tal como o *zoom* ou o *pan*.

Pretende-se que a interação seja coerente e consistente, reduzindo a carga cognitiva ao utilizador. Para isso foram definidos certas interações que se mantêm em toda a aplicação; por exemplo para carregar num botão, utiliza-se sempre o gesto de *tap* e para arrastar um objeto usa-se o *drag*.

Linguagem gestual

Pretendeu-se reduzir ao mínimo a complexidade de interação gestual na aplicação. Foi efetuado um estudo dos gestos utilizados por aplicações multitoque comerciais, tendo-se concluído que estas utilizam interação simples e com recurso a poucos dedos.

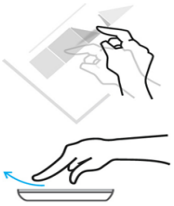
Gesto	Funcionamento	Utilizações frequentes
 Tap	Toque na superfície seguido de um levantar do dedo	• Seleção de botões • Seleção de ações • Abrir controlos de interface
 Double-tap	Dois toques consecutivos na superfície num curto espaço de tempo (inferior a 0.3 segundos)	• Restabelecer o nível de *zoom* inicial
 Press-and-hold	Gesto de pressionar a superfície durante mais do que um Segundo sem interromper o contacto.	• Abrir menu de interatividades
 Pan	Gesto de deslizar o dedo na superfície, semelhante a arrastar um objeto numa mesa	• Navegar na página • Abrir controlos • Arrastar *Popups*
 Flick	Gesto de arrastar o dedo numa superfície de forma rápida	• Navegar para página anterior • Navegar para página seguinte • Abrir *PageNavigator*
 Pinch and strech	Gesto de juntar ou afastar dois dedos em contacto com a superfície.	• Aumentar ou diminuir o nível de *zoom* de uma página

Tabela 7 – Gestos utilizados na solução implementada

O modelo de interação que foi seguido baseia-se numa interação limitada no máximo a dois toques em simultâneo.

A interação suportada pode-se separar em gestos com um e dois dedos. A seguir lista-se e identifica-se o tipo de interações suportadas:

4.2.4 Usabilidade e experiência do utilizador

Neste projeto a utilização da tecnologia e paradigmas multitoque introduz vantagens e desvantagens no que diz respeito à usabilidade e à experiência de utilização. Nesse sentido, foi necessário ter um cuidado especial com os aspeto menos positivos que a tecnologia introduz (tal como a oclusão ou a dificuldade em fazer uma seleção precisa) e tentar mascará-los.

Para corrigir a oclusão dos controlos na interface criou-se uma forma de estes poderem ser escondidos nas extremidades da interface, podendo ser reabertos com ação do utilizador. Caso estejam todos fechados e permaneçam inativos durante 5 segundos, as suas etiquetas passam para opacidade de 10%, permitindo que o utilizador visualize o que se encontra por baixo. Em relação à oclusão dos *Popups* foi criado um mecanismo que os move no momento em que são abertos de forma a evitarem sobrepor-se ao objeto que lhe deu origem. Regra geral, o objeto que é clicado deve manter-se visível, pois o seu aspeto pode variar em função de um estado alterável. Ainda no controlo *Popup* existe um componente que pode ser aberto para mostrar informação extra através do clique num botão. De modo a evitar a oclusão do controlo por parte do dedo, decidiu-se abrir esse componente na lateral direita do *Popup*, evitando assim que o dedo do utilizador tape a informação que se deseja visualizar.

Num sentido mais particular foi decidido permitir que o controlo *PageNavigator* abra do lado esquerdo ou do lado direito em função do local onde o gesto *flick* for realizado. Imaginando uma divisão na vertical da interface, caso o gesto seja efetuado na metade esquerda, então o controlo abre desse lado, caso contrário abre do outro. Este comportamento permite que um utilizador destro ou canhoto utilize o controlo sem qualquer distinção.

A interação na aplicação tenta aproveitar ao máximo o paradigma de manipulação direta permitindo, por exemplo, que o utilizador arraste objetos tal como se estivesse a manipular objetos físicos. Ao criar este paralelismo entre a realidade e o virtual, o tempo de aprendizagem e processamento do utilizador em relação à aplicação, torna-se menor.

O aspeto visual da aplicação foi baseado no visual Metro do Windows 8 e Windows Phone 7 de forma a manter a consistência numa futura inclusão numa destas plataformas. Para além disso, torna-se intuitivo para um utilizador que já tenha alguma experiência num destes sistemas, quando encontra ícones semelhantes e um aspeto visual em tudo parecido com o que já lhe é comum.

4.3 Controlos

Os controlos listados em baixo encontram-se permanentemente na interface, possuindo um botão utilizado para os abrir (Figura 27), que se encontra sempre visível. Este botão pode ser utilizado recorrendo ao gesto de *drag* na direção e sentido da seta representada, ou recorrendo ao toque na área do mesmo. O facto de se criar dois comportamentos possíveis para atingir o mesmo fim, pretende abranger utilizadores que se sentem mais confortáveis e habituados ao toque (menos acostumados ao paradigma multitoque) e também os que intuitivamente se apercebem que o controlo é arrastável. Seguindo a mesma lógica, os controlos podem ser fechados recorrendo ao *drag* na direção da seta (em qualquer ponto do controlo), ou pressionando novamente no botão utilizado para o abrir.

Para evitar a existência de demasiados controlos abertos em permanência, foi implementado um sistema de encerramento automático após um determinado período de tempo

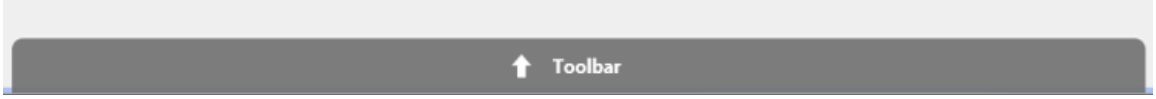


Figura 27 – Exemplo de etiqueta de controlo de interface

(*auto-close*). Este comportamento pretende evitar que o utilizador ocupe, em demasia, a interface principal, deslocando a sua atenção do que é mais importante, ou seja, a visualização de dados e do sistema. É também possível de configurar cada um dos controlos com a possibilidade de ter ou não função de *auto-close*.

4.3.1 Controlos de interface

Quando os controlos foram pensados foi necessário solucionar o problema da sua

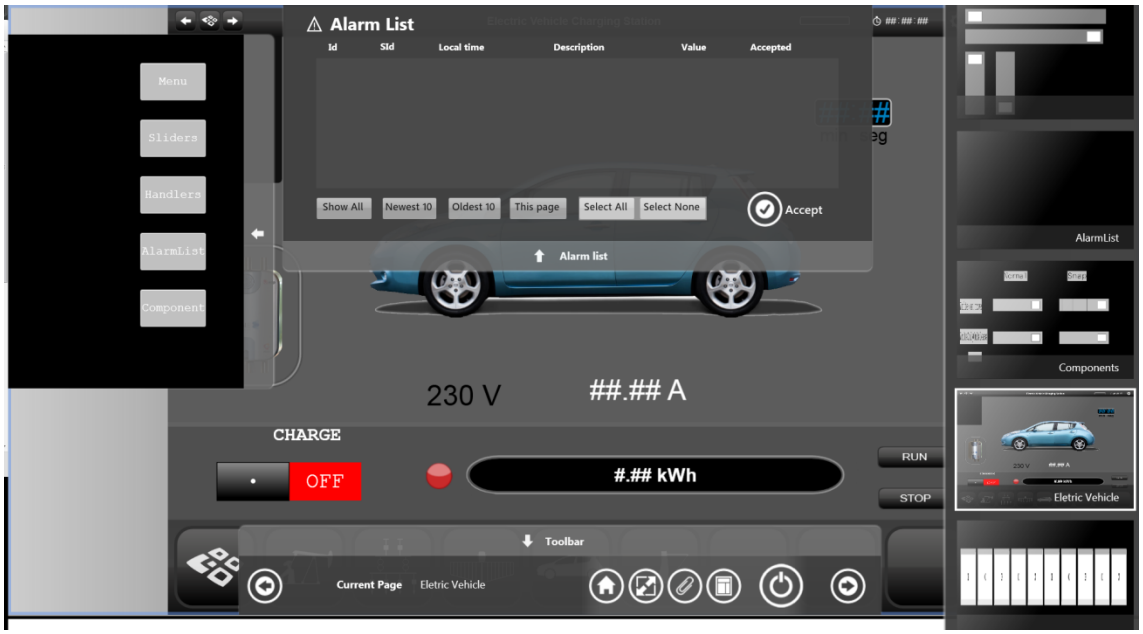


Figura 28 – Controlos de interface abertos

representação. Um dos requisitos era de serem de fácil acesso e de visualização permanente (Figura 28), contudo esta última característica traz consigo um problema inerente que é o da oclusão permanente da interface.

Tendo por base estes requisitos, decidiu-se localizar os controlos nas extremidades da interface. Cada controlo possui uma etiqueta (única parte do controlo que se encontra visível quando este está fechado) com o respetivo nome e que o permite abrir e fechar. O controlo é aberto recorrendo ao *tap* ou *drag* da etiqueta na direção da seta representada, havendo uma animação do controlo caso este seja largado antes da sua posição final. Caso não exista contacto do utilizador os controlos permanecem abertos durante 10 segundos. Após esse período de tempo o controlo é fechado e passados 5 segundos a sua etiqueta passa a ter visibilidade de 10%. Desta forma pretende-se evitar a oclusão do ambiente de trabalho.

Decidiu-se localizar os controlos nas extremidades para permitirem um fácil acesso ao utilizador. O comportamento de abrir e fechar um controlo assemelha-se ao deslizar de uma cortina física. Esta metáfora de interface permite ao utilizador assimilar mais facilmente o seu comportamento com pouco tempo de uso da aplicação e, acima de tudo, antecipar as consequências das suas ações.

Page Navigator

O controlo *Page Navigator* (Figura 29) pretende funcionar como uma forma de acesso rápido a páginas distintas da solução que não tenham sequência linear entre elas. Para esse efeito criou-se um controlo cuja ação de abertura fosse de fácil execução e que permitisse uma visualização lógica e imediata das páginas da solução. O controlo é composto por uma lista alfabeticamente ordenada do conjunto de páginas abertas na solução, cada uma representada por um *thumbnail*.

Figura 29 – Controlo *Page Navigator* aberto

Recorrendo ao paradigma de *floating palette* foi criado um protótipo inicial para este menu que utilizava as propriedades de transparência e de manipulação direta pelo utilizador. Devido à oclusão provocada por este tipo de menu, optou-se por uma outra abordagem na qual o controlo seria deslizante e apareceria sempre ocupando uma área lateral da interface. O menu está inicialmente representado na margem direita da interface, deslocando-se sempre na vertical. Para tornar o seu comportamento mais natural e real, optou-se por adicionar ação de inércia, associado à sua manipulação.

A abertura do controlo faz-se recorrendo a um *swipe* vertical nas extremidades laterais da interface, mas, por forma a não limitar a experiência do utilizador optou-se por permitir a abertura deste controlo do lado esquerdo ou direito do écran, consoante o local do *swipe*. A escolha por este comportamento geral do controlo permite

que haja menor oclusão da interface e que a distância do utilizador ao local que pretende manipular seja menor do que a solução de *floating palette*, para além de não diferenciar um utilizador destro de um canhoto. Para fechar o controlo basta carregar em qualquer sítio da interface ou esperar que este se feche automaticamente (*auto-close*). O controlo pode ser explorado através de um gesto de *drag* na vertical sobre o mesmo, dado que este desliza com conteúdos que se podem encontrar fora da interface. Este comportamento permite concentrar mais informação num menor espaço da interface, ao mesmo tempo que a deixa facilmente acessível e explorável.

O utilizador seleciona uma página clicando no *thumbnail* representativo da página que deseja carregar. A ação de mudança de página tem efeito sobre o *Container* que estiver com foco no momento. É ainda possível carregar uma página para um *Container* arrastando o *thumbnail* respetivo. A representação da página carregada num *Container* é feita recorrendo a um *border* de cor branca adicionado ao *thumbnail* representativo da página. Quando um dos *thumbnails* é arrastado, recorrendo ao *drag* horizontal, o mesmo fica com bordo de cor cinzenta para representar um estado diferente de utilização. À medida que o *thumbnail* é arrastado pela interface o *Container* com foco vai sendo alterado, sendo que o que possui foco no momento em que é largado o *thumbnail* recebe a ação de carregamento de página.

O comportamento do controlo pretende assemelhar-se a o de um *slot machine* na qual existe uma roda com vários objetos, em que só uma parte destes se encontra permanentemente visível. Para visualizar os restantes, o utilizador tem que arrastar o controlo na vertical.

Page drawer

O controlo *PageDrawer* (Figura 30) foi criado para permitir que o utilizador responsável por criar uma configuração para um *device* possa configurar a interface de navegação. O controlo encontra-se na lateral esquerda da interface, tendo o comportamento de um controlo de interface. A sua configuração envolve o desenho de uma página XAML que pode ser carregada para o seu interior. As interatividades da página ficam disponíveis, sendo que as que dizem respeito à navegação incidem sobre o *Container* com foco no momento. Ainda assim não é permitido fazer *zoom* ou *pan* sobre as páginas que são carregadas neste controlo. As dimensões do controlo são definidas na configuração da aplicação sendo que, mesmo que a página seja de dimensão superior prevalecem as dimensões do controlo.

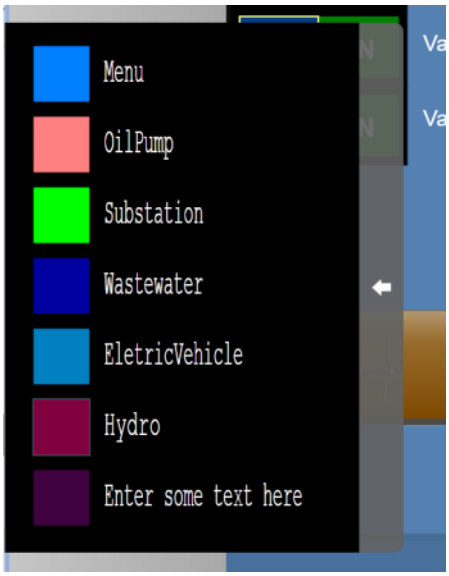


Figura 30 – Controlo *Page Drawer*

O facto do *Page Drawer* se encontrar facilmente

acessível permite que o utilizador aceda a informação que necessita de consultar com regularidade de forma rápida e eficiente. As suas dimensões são configuráveis pelo utilizador no momento de escolha da página a carregar no controlo.

Alarm list

O controlo em causa pretende servir como lista dinâmica de consulta de alarmes que existam na solução. Os alarmes são adicionados em tempo real, sendo possível consultar e aceitar alarmes da lista. Como o objetivo é disponibilizar uma forma de o utilizador consultar informação de forma fácil e rápida, foi decidido criar este controlo com representação permanente na interface (Figura 31) através de uma etiqueta representativa com o seu nome.

A lista de alarmes, representada na Figura 32, possui uma entidade associada, o Klaxon.

Figura 31 – Etiqueta da lista de alarmes

Este elemento representa o estado da lista de alarmes e é ativado quando existe um novo alarme disponível. É representado por um botão vermelho localizado na extremidade direita da etiqueta e possui uma animação intermitente quando é ativado (em paralelo é também disparado um aviso sonoro para alterar o utilizador). Para o desativar o utilizador necessita de o pressionar, o que automaticamente termina a animação

Cada item de alarme possui diversos campos associados: um Id, um Sequence Id, a data e hora em que foi criada, uma pequena descrição do alarme, um valor que e um estado de aceitação. A lista permite selecionar múltiplos alarmes de uma vez, sendo que a única ação permitida é aceita-los. A ação de aceitar todos os alarmes não desativa o Klaxon.

O controlo permite ainda que o utilizador faça um filtro dos alarmes pelos alarmes mais recentes, mais antigos ou pelos que dizem respeito à página aberta. É ainda possível selecionar

Figura 32 – Lista de alarmes aberta

todos os alarmes ou anular a seleção efetuada recorrendo a botões de atalho. Numa utilização

normal, o utilizador seleciona os alarmes que pretende da lista e carrega no botão de *aknowledge* (representado com um visto) para os aceitar. De forma a manter uma semelhança com as listas de dados utilizados no paradigma GUI, adaptou-se a ordenação dos valores através do toque no título do campo respetivo.

Toolbar

O controlo *Toolbar* foi pensado para abranger requisitos inicialmente definidos de suporte ao utilizador. Pretendia-se criar um controlo que permitisse complementar a navegação e a exploração das páginas, disponibilizando diversas opções num contexto de utilização avançada.

No seu desenvolvimento existiu um conceito base: facilitar a navegação na aplicação, impedindo que o utilizador se perdesse e acabasse por entrar num estado irreversível. Para prevenir que isso acontecesse foram criados diversos atalhos: voltar ao início da aplicação, navegar para a página seguinte/anterior e um campo com o nome da página que estaria aberta no

Figura 33 – Protótipo inicial da *Toolbar*

momento.

Esta proposta foi sofrendo alterações durante o processo de prototipagem. Em termos estéticos e de usabilidade a proposta evoluiu de uma barra fixa e estática que ocupava toda a dimensão horizontal do écran (Figura 33), para uma barra modular e com dimensão fixa inicial que podia ser expandida na horizontal e na vertical. Esta característica permitia a disponibilização de mais informação e opções na barra, sem contudo ocupar tanto espaço na interface. A nível de comportamento a barra seria inicialmente apresentada escondida, recorrendo-se a um gesto de *swipe* vertical, num objeto presente na interface, para a fazer aparecer.

Por fim, chegou-se a uma solução final (Figura 34), na qual a barra seria apenas possível de manipular na horizontal sendo que, para a abrir, se arrastaria uma etiqueta representativa da

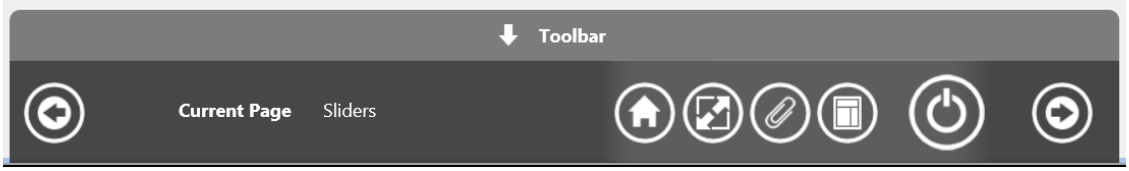


Figura 34 – Aspeto final do controlo *Toolbar*

sua presença. Como atalhos, para além dos de navegação já referidos, introduziram-se cinco referentes a funcionalidades presentes no sistema: a entrada no modo de criação/edição de notas, a criação de um *Splitview* vertical, a possibilidade de fazer *zoom-to-fit*, a possibilidade de recuar para a primeira página, e ainda, um atalho para terminar a aplicação.

Popup

O *Popup* é um controlo modular manipulável pelo utilizador, que pode ser utilizado em diferentes contextos e implica uma ação por parte do utilizador para ser aberto. É possível haver múltiplos *Popups* abertos, havendo uma ordenação vertical por ordem de criação. Esta gestão é efetuada por um controlo existente no SDK do Microsoft Surface denominado *ScatterView*. Este componente trata de toda a gestão dos *Popups*, seja em termos de ordenação, localização e mesmo de abertura e fecho dos mesmos.

Como já foi referido, este controlo pode ser utilizado em diversas situações, nomeadamente para confirmar a execução de um controlo (Figura 35), carregar uma página, uma lista de alarmes, ou mesmo adicionar uma nota a uma página. Este controlo é ativado através de uma ação associada a um evento um objeto de uma página. Quando esse evento é disparado (usualmente é um evento de clique), abre-se um *Popup* dando foco ao objeto que lhe

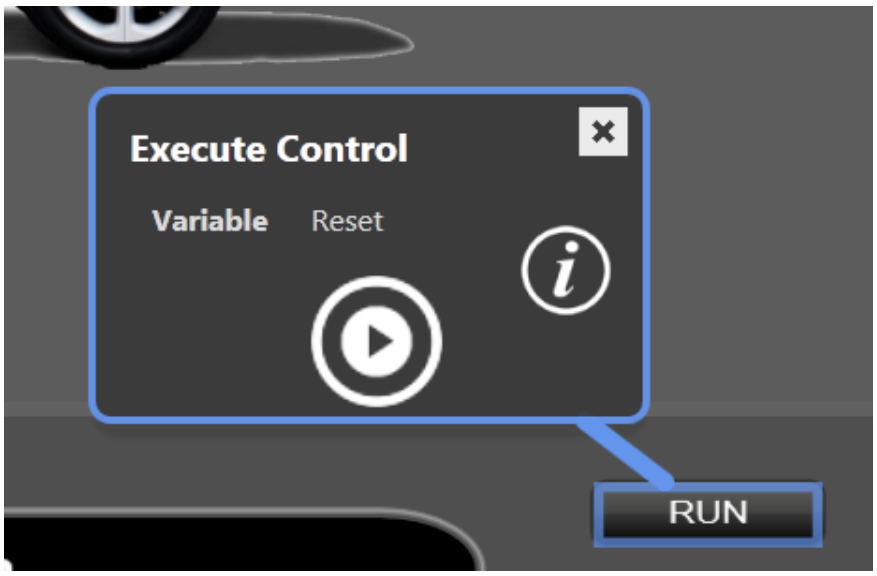


Figura 35 – *Popup* de execução de controlo com informação extra

deu origem, através da representação de um bordo, à sua volta. Do bordo, segue uma linha que o liga ao *Popup*, para criar uma perceção visual de pertença a um objeto. O rasto permite que o utilizador associe facilmente o objeto a que cada *Popup* se refere, pois, por vezes, existe necessidade de ter vários *Popups* abertos em simultâneo. Para fechar um *Popup*, existe um botão com um 'X' no canto superior direito, de modo a ser eficaz e consistente com os *Popups* existentes noutros paradigmas de interação (WIMP).

A prototipagem deste controlo seguiu sempre um modelo no qual o controlo se encontrava por cima da informação da página, e que teria uma linha a indicar o objeto do qual dependia, ou sobre o qual estaria a atuar (*line trace*). Este comportamento manteve-se, sendo que na ordenação vertical, os *Popups* encontram-se a um nível superior ao do plano das Páginas.

Os *Popups* são também utilizados na inserção de notas de apoio nas páginas da solução, apresentando-se com um bordo e linha de ligação amarelados, enquanto que os *Popups* de

execução e escrita de variáveis, se apresentam com cor azulada. Estes últimos possuem ainda uma lista de parâmetros acessíveis, através de um botão que faz deslizar um objeto na lateral do *Popup* quando clicado. Esta lista possui pares de parâmetro-valor para consulta, caso se justifique, antes da execução da ação. Inicialmente, foi proposto ser colocado e aberto a partir do fundo do *Popup*. Esta ideia foi abandonada, porque, ao clicar para abrir a lista, o dedo ou a mão do utilizador podia ocultá-la.

Os *Popups* de execução de controlos, de escrita de variáveis e de notas têm *trace* e *auto-close* automáticos e não existem opção de configuração. Já os *Popups* de abertura de página ou de lista de alarmes/eventos podem ter ou não *auto-close*, sendo esta opção definida na configuração da solução.

Os *Popups* de notas têm um comportamento diferente dos restantes, pois não possuem botão de confirmação. Como forma de tornar a interação mais simples e imediata, pretende-se transmitir ao utilizador que as ações que ele toma sejam executadas no imediato, portanto, mal este feche o *Popup*, o conteúdo fica gravado. Este comportamento é diferente da execução de controlos, dado que essas são entendidas como operações críticas, sendo para isso requisitada

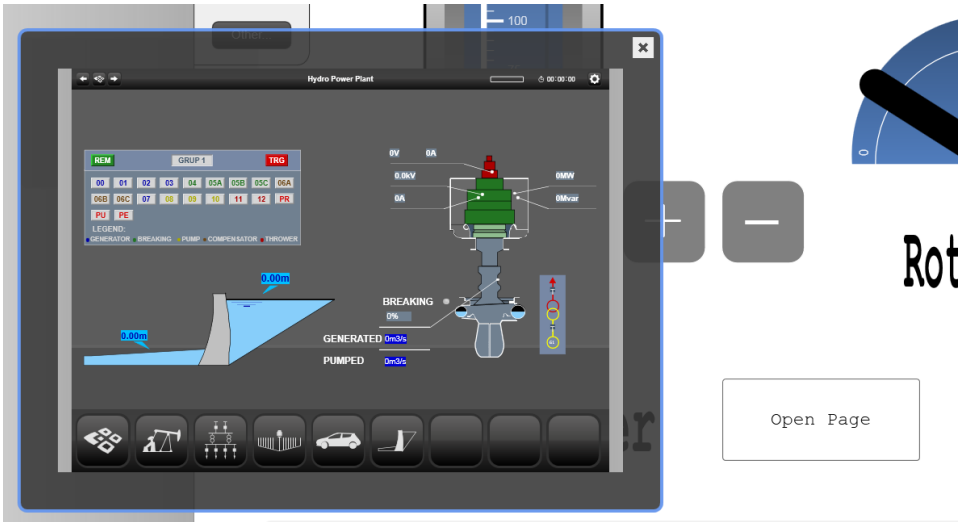


Figura 36 – *Popup* com página carregada e sem *line trace*

uma confirmação ao utilizador.

Os *Popups* também podem ser utilizados para carregar páginas desenhadas pelo utilizador. Pretende-se que este controlo permita a abertura e visualização de páginas não sequenciais (Figura 36) com informação que se pretende visualizar. Estas páginas podem ter ou não propriedades de manipulação (*zoom*, *pan)*, sendo que a interação com as ações em cada uma (execução, escrita ou navegação) está sempre habilitada. No que diz respeito às ações de navegação, estas incidem sobre o *Container* do próprio *Popup*, e não sobre o que tiver foco no momento.

Keypad

O *Keypad* é um controlo que aparece associado a um comando de escrita de variável numérica (Figura 37), simplificando o processo de introdução de dados. Foi decidido criar este controlo para permitir melhor integração dos componentes utilizados na interação com a aplicação em si, seja em termos de aspeto, seja em termos de comportamento. O facto de haver menos uma janela suspensa (a do teclado virtual do Windows 7) evita a oclusão da interface e simplifica o uso da aplicação. Este controlo aparece automaticamente quando o comando de escrita é aberto deslizando a partir da margem direita do écran. É composto por 10 botões que representam os algarismos de 0 a 9, um botão para introdução de números decimais, um botão para apagar o número mais à esquerda, um botão 'OK' para confirmar e, ainda, uma caixa de texto onde é representado o número atual.

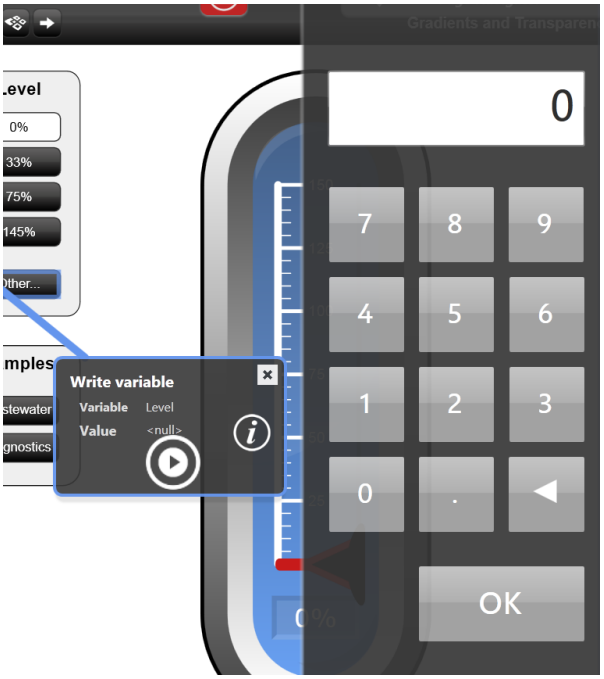


Figura 37 – Controlo *Keypad* com *Popup* associado

Numa utilização natural, o utilizador, após abertura do *Keypad*, pressiona os números que deseja introduzir e, finalmente, pressiona o botão 'OK'. Este botão automaticamente trata de escrever o novo valor da variável e de fechar o *Keypad* e o *Popup* que lhe está associado. No caso de se pretender cancelar a inserção de valor, pode-se proceder de duas maneiras: ou se espera 10 segundos até o *Popup* e o *Keypad* fecharem ou, então, fecha-se o *Popup* que automaticamente cancela a operação de inserção de valor.

Neste controlo não existiu grande preocupação em habilitar o uso do controlo para utilizadores destros e canhotos, pois o gesto requerido no uso do controlo é pouco complexo. Os botões numéricos do controlo possuem 100x100 pixéis de forma a aumentar a facilidade de inserção de dados sem enganos, sendo equivalente na resolução utilizada a cerca de 2 centímetros. O *Keypad* pode ser substituído pela utilização do teclado físico ou virtual, pelo pressionar do ícone que aparece junto à caixa de texto; nesse caso, é aberto um teclado virtual pertencente ao Windows 7.

Interface Bar

O controlo *Interface Bar* (Figura 38) surgiu com base num requisito inicialmente definido de permitir que uma margem lateral do écran pudesse ser reservada para carregamento de páginas. Estas poderiam servir para mostrar uma lista de alarmes, menu de navegação ou outra

informação importante para o utilizador. As ações de navegação neste caso incidem sobre o *Container* que se encontrar com foco. É possível dispor o controlo em qualquer das margens da interface (cima, baixo, esquerda ou direita), sendo que a sua dimensão é definida na configuração da aplicação. O facto de este controlo existir, implica que a área de exploração e navegação (onde se inserem os *Containers*) se torna mais reduzida, sendo, por isso, necessário ter em conta este facto, no momento de definição do tamanho do controlo *Interface Bar*. O posicionamento dos controlos fixos na interface é também alterado em função do local em que a barra é colocada. Todas as ações referentes a interação com os limites da interface passam a ser possíveis dentro deste espaço definido anteriormente. Caso o controlo se encontre desabilitado, a interação funciona relativa à totalidade da interface.

Ao tornar este controlo versátil, pretende-se fornecer ao utilizador a possibilidade de ele desenvolver uma página apropriada com a informação que pretende disponibilizar. Desta forma, aumenta-se as possibilidades de configuração e interação com a aplicação.

4.3.2 Controlos embebidos de página

Um dos requisitos iniciais do projeto era o de poder haver uma reutilização de componentes. Nesse sentido, para permitir uma maior flexibilidade na aplicação, foram desenvolvidas interatividades possíveis de serem atribuídas a objetos do desenho. Esta lógica de interação permite tornar componentes gráficos estáticos em objetos de controlo dinâmicos, comportando-se como objetos de definição de valores, execução de controlos ou mesmo de navegação. O objetivo principal é o de fornecer ao utilizador a noção de que tudo o que se apresenta perante si é manipulável, introduzindo um conceito de liberdade, melhorando a experiência de utilização.

Contudo, a criação de controlos costumizáveis e flexíveis pode introduzir um problema a nível de consistência. Como o aspeto gráfico está na sua totalidade do lado do utilizador que configura a solução (desenho das páginas), corre-se o risco de haver certos componentes que não são intuitivos ou cujo significado não seja percetível. Para colmatar esse problema, deve-se recorrer à definição de linhas de orientação de desenho (a definir num momento futuro) que não violem as regras de usabilidade e que favoreçam a facilidade de utilização da aplicação.

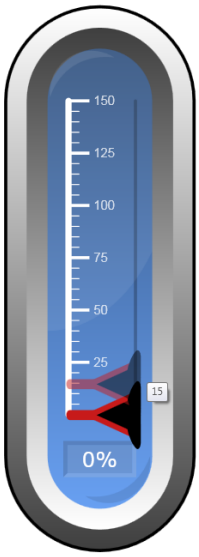


Figura 39 – Controlo *TouchSlider* (W*riteOnlyOnRelease* activada)

Touchslider

O controlo *Touchslider*, representado na Figura 39, permite definir um valor de uma variável associada através da manipulação de um objeto de uma página dentro de um determinado limite pré-definido. Inicialmente é definida a

variável a ser escrita, os valores iniciais e finais da mesma e os limites de manipulação do objeto (posição inicial e final). É ainda possível escolher se o valor a definir é discreto (com definição de intervalo) ou contínuo.

O controlo pode ter dois comportamentos distintos. Caso esteja definido com a opção *WriteOnlyOnRelease* descativada, é feita uma escrita de variável por cada alteração na manipulação detectada pela aplicação. Ou seja, cada vez que o objeto seja movido, a aplicação detecta esse evento e lança uma escrita do valor correspondente à manipulação efetuada. Caso esta opção esteja ativada, o utilizador, quando manipula o objeto, está de facto apenas a manipular uma cópia do mesmo com opacidade a 50% (Figura 39 – Controlo *TouchSlider* (WriteOnlyOnRelease activada) não havendo escrita do valor durante o decorrer da manipulação. Quando o contacto com a superfície é terminado é aberto um *Popup* de confirmação para o utilizador pode aceitar, dando origem à execução de escrita do novo valor. Caso o utilizador cancele, esta a cópia do controlo desaparece, mantendo-se o valor da variável igual ao valor de início da manipulação. Em ambos os comportamentos é aberta uma *tooltip* a indicar o valor corrente em qualquer momento da manipulação, sendo automaticamente fechada quando termina.

A lógica de interação pretende tirar partido de uma metáfora de interface usual nos controlos de regulação de valores físicos. Estes controlos podem- ser encontrados, por exemplo, nas mesas de mistura de som, onde um objeto é arrastado linearmente, num sentido ou no outro, alterando o volume do som, a quantidade de baixos, agudos, graves, etc.

RotateSlider

O componente *RotateSlider* (Figura 40) é um controlo que se comporta de forma idêntica ao *TouchSlider*. São associadas interatividades a um objeto representado numa página que passa a poder ser movido num sentido rotacional, de acordo com as propriedades definidas. Estas propriedades são atribuídas por um ponto que define o centro de rotação, um ângulo inicial, um ângulo final, o nome da variável associada e os valores limite. O *RotateSlider* possui também a possibilidade de funcionar apenas com valores discretos, recorrendo à propriedade de *snap*. O controlo permite, ainda, definir a direção de rotação (*clockwise* ou *anti-clockwise*).

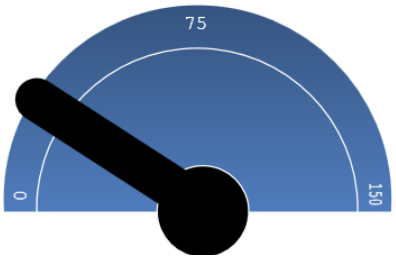


Figura 40 - Exemplo de *RotateSlider*

Botões de incremento/decremento

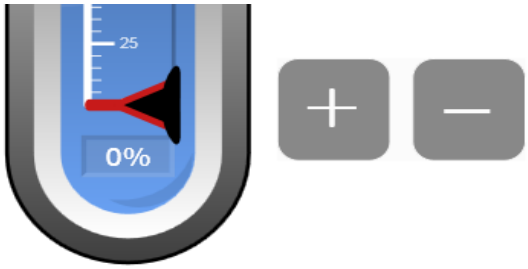


Figura 41 – Botões de incremento e decremento de valores

Para flexibilizar o processo de utilização de controlos embebidos de página foi decidido criar interatividades de incremento e decremento. Estas interatividades funcionam como um complemento à utilização normal de um controlo embebido. Podem ser associadas a botões numa página (Figura 41), permitindo incrementar ou decrementar o valor de uma variável numa certa quantidade. Ao haver esta atualização do valor da variável, automaticamente todos os controlos *RotateSlider* e *TouchSlider* sofrem uma atualização dos seus valores. Esta forma de interagir, torna mais flexível o processo, pois os botões não necessitam de ficar indexados a nenhum controlo em particular, mas sim à variável que lhes diz respeito.

Por fim, convém salientar, ainda, que o comando de escrita é executado quando o utilizador pressiona o botão, não havendo possibilidade de definir ação de confirmação. Decidiu-se optar por este comportamento, porque este controlo tem como base um requisito de incremento rápido e simples de valor, visto que a abertura de um *Popup* de confirmação violaria este requisito. Há que salientar ainda que, quando o utilizador pressiona um dos objetos, a variável associada é alterada no momento.

ToggleButton

Um controlo embebido *ToggleButton* é uma interatividade que pode ser associada a um objeto de uma página XAML. Foi criada para resolver o requisito de necessidade de existência de interatividade de página que permitisse alterar o valor de uma variável com dois estados possíveis (booleana ou multiestado). Esta alteração de estado é efetuada sobre um determinado controlo definido nas propriedades do *ToggleButton* e com um determinado valor associado a cada estado. Para alterar o estado do objeto, o utilizador pode pressionar o componente responsável por esta alteração (Figura 42), deslocando-se, este, na horizontal, ou pode arrastá-lo no sentido que deseja alterar. No caso da imagem é deslocado o retângulo com uma bola branca no seu centro para a direita, sobrepondo o retângulo avermelhado. Caso o objeto de controlo seja largado antes da sua posição final, é iniciada uma animação que o desloca até à posição que se encontrar mais perto.

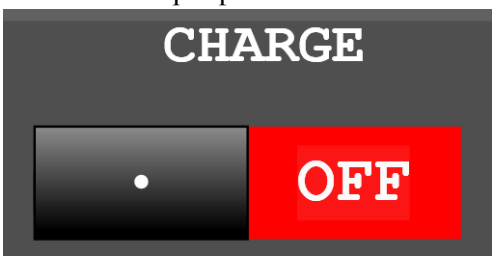


Figura 42 – Exemplo de um controlo embebido *ToggleButton*

Para definir um controlo deste tipo, é necessário fornecer a variável associada e o estado inicial em que ela se encontra. Assume-se que o deslocamento em pixéis do controlo na mudança de estado, é igual ao tamanho do objeto, logo não é algo que tenha necessidade de ser definido pelo utilizador.

Alarm e Event list

Um controlo embebido do tipo *AlarmList* ou *EventList* é um controlo semelhante ao *AlarmList* (já analisado em 0), apresentando-se de forma diferente, dado que se encontra inserido numa página XAML. O seu comportamento é em tudo igual ao *AlarmList*, para além de

possuir as mesmas opções de manipulação e de filtro. Foi criado para suprimir o requisito de poder ser adicionada uma lista de alarmes numa página, flexibilizando a experiência de utilização. O utilizador responsável pela configuração da aplicação pode, por exemplo, criar uma página com uma lista de alarmes e apresentá-la no *PageDrawer* ou num *Popup*. Desta forma, aumentam-se as possibilidades de criar configurações diversas para melhor resolverem os problemas de cada situação.

Na configuração deste controlo, é necessário definir, apenas, se é do tipo *ReadOnly*, sendo que, se este estado é verdadeiro, o controlo é do tipo *EventList*. O *AlarmList* difere do *EventList* na medida em que, este último, apenas serve como consulta, não dispondo da opção *Aknowledge*. As opções de filtro e ordenação são iguais em ambos os controlos, sendo ainda possível definir se o controlo permite, ou não, seleção múltipla de alarmes.

4.4 Resumo

O presente capítulo serviu para apresentar o sistema detalhadamente. Foi analisada a arquitetura da interface, do modo como a informação é apresentada, e na forma como esta foi composta. Fez-se um resumo sobre o processo de desenvolvimento e prototipagem, havendo uma justificação de todas as decisões tomadas no processo de desenho e implementação.

Foi explicada a lógica de interação global na aplicação, havendo uma demonstração dos gestos e tipo de toque suportado. Foram também descritas as principais preocupações relativas à melhoria da usabilidade e da experiência de utilização da aplicação.

Por fim, foram descritos os controlos implementados a nível de interface e de página, suportados nos requisitos inicialmente definidos. A sua lógica de interação e comportamento foi analisada e justificada com detalhe.

Capítulo 5

5. Testes e resultados

5.1 Testes de usabilidade

Os testes de usabilidade são uma ferramenta para avaliar o nível em que um produto se encontra em relação a determinados critérios de usabilidade (Jeff Rubin 2008)(Jeff Rubin 2008). São realizados com recurso a utilizadores-alvo, de acordo com um perfil traçado, que executam diversas tarefas que pretendem cobrir as funcionalidades essenciais da solução a testar. Este processo pode decorrer numa fase de prototipagem e modelação, na qual a interface é construída recorrendo à opinião do utilizador ou já numa fase mais avançada do projeto com o objetivo de validar algo já implementado.

O processo de teste é bastante custoso pois envolve um número elevado de pessoas com disponibilidades individuais que têm de ser conciliadas, sendo que, em alguns casos, não participam sem receberem uma comparticipação financeira. Acaba por ser também um processo bastante moroso dado que envolve a execução de diversas tarefas por várias pessoas. É necessário saber equilibrar a necessidade de teste com os recursos disponíveis para a execução do projeto.

5.2 Objetivo

De forma a proceder a uma validação da aplicação desenvolvida e a uma comparação com uma solução existente (HMI – Human Machine Interface), decidiu-se proceder à execução de testes de usabilidade. O objetivo principal dos testes era de verificar se existem melhorias na alteração do paradigma de interação de uma solução SCADA, no contexto de uma utilização profissional. Para obter melhores resultados, optou-se por separar os utilizadores em dois grupos

distintos, tendo um deles bastante experiência no HMI e outro pouca ou nenhuma. Com dois grupos distintos é possível comparar performances, confrontando determinados aspetos em causa.

Noutro sentido pretendia-se também validar a aplicação no que concerne à experiência de utilização. Pretendia-se verificar diversas características: facilidade de execução, ser ou não intuitivo e conforto na utilização. Além destes aspetos pretendia-se também estudar o nível de ambiguidade associada a cada aplicação.

Ainda assim, convém referir que estes testes têm meramente um carácter indicativo e preliminar, tendo em conta a amostra reduzida que foi possível reunir. O objetivo principal foi de obter um primeiro *feedback*, fazendo uma avaliação geral em relação a este tipo de interface e interação e não de fazer um estudo final de usabilidade. Para obter resultados mais conclusivos e sustentados seria necessário elaborar um estudo mais completo e com maior número de utilizadores.

5.3 Plano de testes

5.3.1 Caracterização da população

Grupo 1 – Grupo com pouco conhecimento da plataforma HMI

O grupo 1 foi constituído por 6 pessoas da equipa de desenvolvimento do Automation Studio, com pouca ou nenhuma experiência de utilização do HMI. É composta por 4 pessoas do sexo masculino e 2 do sexo feminino, sendo que as idades estavam compreendidas entre os 26 e os 32 anos (com média de 29.5 e desvio padrão 1.71). Dos seis, apenas um indivíduo já tinha tido contacto com tecnologias multitoque (excetuando *tablets* e telemóveis) e em relação ao HMI apenas dois não tinham tido qualquer contacto com o mesmo.

Grupo 2 – Grupo com conhecimento elevado da plataforma HMI

O grupo 2 foi composto por 6 pessoas entre operacionais, quadros de configuração (cujas competências são de desenho de páginas no Automation Studio) e por elementos da equipa de desenvolvimento do HMI. As idades dos elementos do grupo situavam-se entre os 26 e os 41 anos (com média de 33.5 e desvio padrão 4.59), sendo que 5 eram elementos do sexo masculino e um do sexo feminino. No grupo dois indivíduos já tinham tido contacto com tecnologia multitoque (descartando *tablets* e telemóveis). No que diz respeito à plataforma HMI estes eram utilizadores usuais, uns num contexto de trabalho e monitoria e outros na função de desenvolvimento de aplicações para clientes profissionais.

5.3.2 Execução

Os testes foram executados segundo um guião pré-definido (Anexo B e C). As sessões decorreram individualmente e em média duraram entre 15 a 20 minutos. Inicialmente foi feita uma pequena introdução sobre o projeto e a aplicação em geral, explicando o contexto e o objetivo da mesma. Esta abordagem pretender deixar o utilizador mais confortável, passando-lhe um sentimento de à-vontade, salientando-lhe que não está a ser alvo de avaliação. De seguida foi permitido que durante 5 minutos as pessoas se ambientassem a ambas as plataformas (Simulation.Touch e HMI). Após este intervalo de tempo foram executados três testes em sequência com recurso a um guião individual (um por teste), sem intervenção do moderador. O primeiro teste foi executado no Simulation.Touch e foi destinado a verificar e validar a aplicação em termos de usabilidade, navegação e exploração. O segundo e terceiro teste (um executado no Simulation.Touch e o outro no HMI) tiveram como objetivo comparar a execução de um exemplo em termos de eficácia, eficiência, prazer de utilização e fiabilidade. Por fim foi pedido o preenchimento de um pequeno questionário de oito perguntas (D) relativas à experiência de teste.

Cada sessão foi gravada utilizando uma câmara de vídeo para simplificar o processo de tratamento da informação gerada. Os testes foram executados num monitor de 23 polegadas com tecnologia multi-toque, tendo sido utilizada uma resolução de 1920x1600 (Figura 43). Em

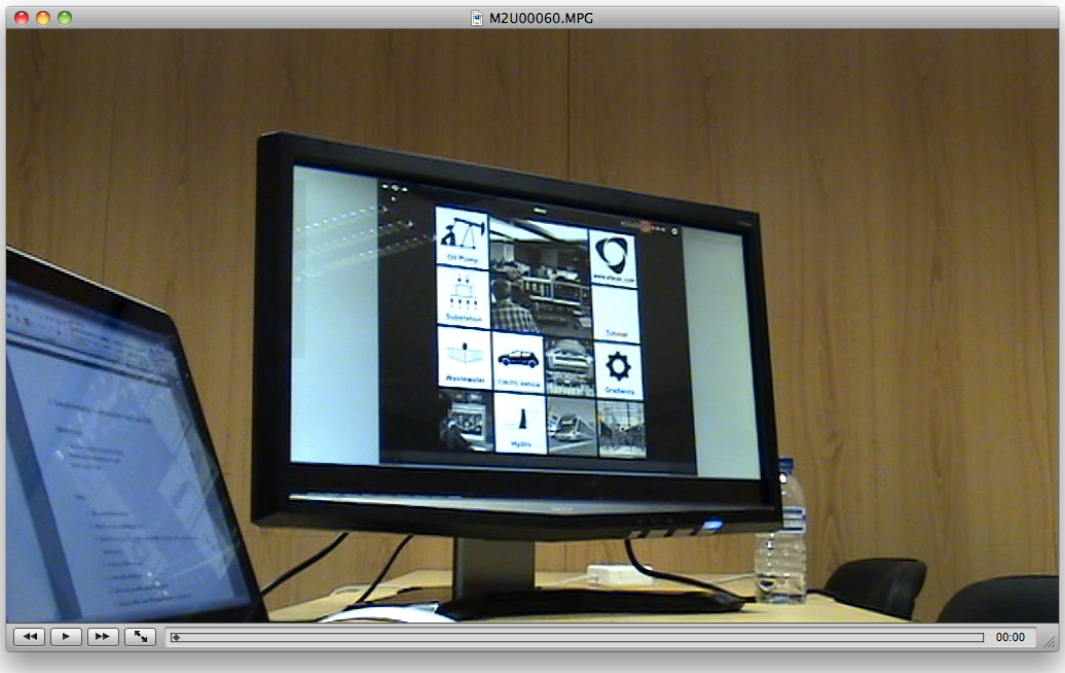


Figura 43- Exemplo do registo de vídeo recolhido

cada vídeo foram recolhidos diversos dados: tempo de execução, número de toques totais

efetuados, sucesso do número de toques e sucesso da execução da tarefa proposta. Estas medidas permitem caracterizar o processo de teste e a fiabilidade de utilização da aplicação desenvolvida.

Teste 1 – Verificação e validação

O Teste 1 foi realizado por ambos os grupos a partir de um guião pré-definido (B) de realização de uma tarefa exemplo no Simulation.Touch. Este teste é do tipo de Verificação e Validação que pretende validar se uma tarefa, quando executada, preenche determinados objetivos pré-estabelecidos. Para validar a aplicação desenvolvida predefiniram-se métricas na execução do primeiro teste, nomeadamente:

- Número de interações totais utilizados durante o teste (considera-se a utilização de dois dedos em simultâneo ou o deslizar do dedo como um toque apenas);
- Número de interações erradas ou sem sucesso (quando o utilizador pretende pressionar o écran e não recebe a ação esperada);
- Tempo demorado na realização do teste;
- Percentagem de sucesso na execução do teste;

Os objetivos para cada métrica encontram-se descritos na tabela seguinte:

Sigla	Métricas	Valor
TT	Número de interações totais	30
TSS	Percentagem de interações erradas	10%
TD	Tempo despendido	02:00
S	Percentagem de sucesso	70%

Tabela 8 – Métricas definidas para validar o teste número 1

Os valores especificados dizem respeito à execução da tarefa com um critério mínimo de fiabilidade (percentagem de sucesso) e eficiência (tempo de execução, número de interações totais e percentagem de interações erradas). Estas métricas foram definidas com a equipa de desenvolvimento dado que esta tem imensa experiência na plataforma o que permitiria detectar quando a aplicação deixasse de ser fiável para utilização profissional.

Há que salientar que os dados relativos às interações totais, interações sem sucesso e tempo de execução apenas dizem respeito aos testes que forem executados com sucesso. Considera-se um teste efetuado com sucesso caso o utilizador consiga executar todas as tarefas presentes no guião sem qualquer ajuda ou intervenção externa. Os testes que não obtiveram sucesso não foram contabilizados pois, nestes casos, os utilizadores ou saltavam tarefas do guião ou então

chegavam a um estado em que não conseguiam executar nenhuma tarefa sem ajuda do moderador do teste.

Teste 2 e 3 – Comparativo

Para medir a qualidade da solução Simulation.Touch decidiu-se executar um teste comparativo com a solução HMI. Este tipo de teste pretende servir como um comparativo entre soluções idênticas ou cuja função seja a mesma. A ideia base foi criar uma tarefa exemplificativa de uma utilização normal de uma plataforma SCADA, abordando a função de navegação, execução de controlos e monitorização.

Aos utilizadores foi distribuído um guião com diversos passos que teriam de executar, primeiro na plataforma Simulation.Touch e de seguida na plataforma HMI. Nesta tarefa foram registados o sucesso ou insucesso do teste, o número total de interações utilizadas, o número de interações erradas e o tempo total despendido. Entende-se um interação errada como uma ação de contacto do utilizador com a interface em que não existe resposta positiva por parte da aplicação, ou seja, à ação do utilizador não surge a resposta que este espera. Este dado pretende medir a fiabilidade de utilização de uma plataforma em relação à outra.

Tal como no teste 1, os dados recolhidos (interações totais, interações sem sucesso e tempo despendido) referem-se apenas aos casos em que o teste foi executado com sucesso.

Questionário

O questionário foi criado com o objetivo de captar aspetos menos objetivos relacionados com a experiência em causa por parte do utilizador. Numa primeira fase foi pedido para estes avaliarem determinados aspetos numa escala crescente de 1 a 5, sendo 5 o valor mais positivo e o 1 o menos positivo.

Numa segunda fase foi solicitado para escolherem preferências no que diz respeito à execução do último teste, salientando se o preferiram executar no Simulation.Touch, no HMI ou se não houve qualquer preferência. Os aspetos questionados foram os seguintes: eficiência, eficácia, facilidade de execução e prazer de utilização. Os dados foram compilados e separados nos dois grupos de indivíduos.

Por fim foi pedido aos utilizadores para escreverem algum comentário em relação aos testes (ou ao projeto no global), com o objetivo de fornecer alguma ideia suplementar, que complemente a execução dos testes.

5.4 Apresentação e discussão de resultados

5.4.1 Teste de validação

Pela análise dos dados da Tabela 9 é possível inferir que os resultados nas diferentes métricas diferem do grupo 1 para o Grupo 2. Os elementos do grupo 1 conseguiram realizar a

tarefa, em média, com mais rapidez (01:29m vs. 01:41), com recurso a menos interações totais (24.60 vs. 31) e menos interações sem sucesso (2.60 vs. 3.25). Saliente-se ainda que o sucesso geral de execução no grupo 2 é bastante inferior ao do Grupo 1 (67% vs. 83%).

	T1 TT	T1 TSS	T1 TD	Sucesso	% TSS
Média Grupo 1	24,60	2,60	01:29:24	83%	10,6%
Média Grupo 2	31,00	3,25	01:41:45	67%	10,5%
Desvio padrão Grupo 1	6,53	2,42	00:20:18		
Desvio padrão Grupo 2	5,48	1,26	00:05:34		
Média Total	27,80	2,93	01:35:34	75%	10,6%

Tabela 9 – Resultados do teste número 1

No global, analisando os valores médios, podemos observar que no que diz respeito ao número de interações totais estes são inferiores à métrica definida (27.80 vs. 30). Ainda assim, o número de interações totais do grupo 2 excede o valor definido em uma unidade (31). No que diz respeito à percentagem de interações sem sucesso, estes são em médias superiores ao estimado (10,6% vs. 10%), sendo que a diferença é mínima. Ambos os grupos apresentam valores semelhantes (10,6% vs. 10,5%). O valor mais díspar é o que diz respeito ao tempo de execução na qual existe uma diferença de cerca de 25 segundos, do valor médio em relação ao valor de referência (01:35 vs. 02:00). Finalmente, o sucesso médio de execução é de 75% sendo superior aos 70% definido. O Grupo 2 obteve um sucesso de execução de 67% não tendo ultrapassado o limite mínimo exigido (70%).

Se analisarmos o desvio padrão das amostras verificamos que alguns resultados podem ultrapassar os valores-limite impostos para a métrica correspondente. Este aspeto revela-se no caso das interações totais de ambos os grupos (24.60 + 6.53> 30 e 31.00 + 5.48> 30). Pode-se explicar este resultado pelo facto da amostra ser de reduzida dimensão, o que pode levar a que os resultados sejam mais díspares e menos consistentes.

Dos resultados apresentados pode-se retirar que as métricas apresentadas são satisfeitas no caso das interações totais, tempo despendido e sucesso de execução. As interações sem sucesso são superiores ao definido, sendo a diferença mínima em relação a este limite.

5.4.2 Teste comparativo – Simulation.Touch e HMI

Depois da avaliação da validade da aplicação procurou-se comparar a mesma com uma solução já existente e frequentemente utilizada. Desta forma pretende-se obter uma indicação de que a aplicação proposta é fiável num contexto profissional e eficiente na óptica do utilizador. Nestes testes foram registados valores para as mesmas métricas analisados no teste número um.

Em termos gerais existem diferenças registadas no tempo dispendido na execução do teste entre grupos (Figura 44). No que respeita ao grupo 1 não se verificam diferenças muito significativas (01:43:40 vs. 01:43:00). No que respeita ao grupo 2 verificou-se mais tempo despendido no Simulation.Touch do que no HMI (02:01 vs. 01:51). Em termos médios verificamos que existe uma pequena diferença do Simulation.Touch com mais tempo

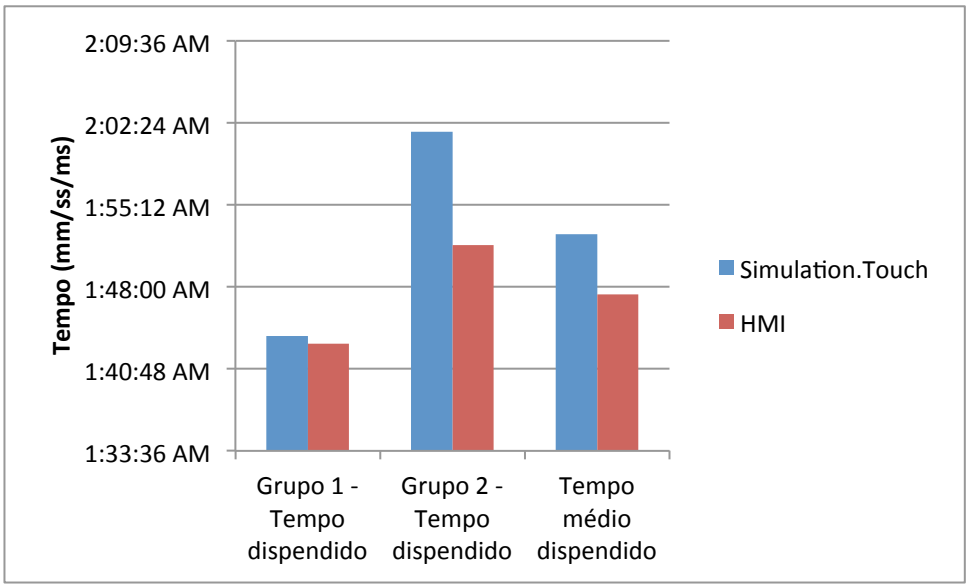


Figura 44 – Dados de execução do teste 2 e 3 referentes ao tempo demorado separado por grupo

despendido (01:52 vs. 01:47).

Através da análise do tempo despendido por cada grupo na execução das tarefas consegue-se () verificar que o Simulation.Touch em média regista um tempo de execução mais elevado do que o mesmo teste no contexto do HMI (01:52 vs. 01:47). Esta diferença pode ser explicada pela necessidade de habituação a um paradigma diferente de interação dado que a grande maioria dos utilizadores nunca tinha tido contacto com tecnologia multitoque. Contudo, se se apenas analisar o grupo 1, nota-se que a diferença entre os dois testes é mínima (ao contrário do grupo 2), levando a concluir que o tempo de execução para quem não conhece ambas as plataformas é semelhante. Este aspeto é relevante e indicia que as plataformas se comportam de forma bastante semelhante, não havendo grande diferença de tempo despendido na execução da mesma tarefa nas diferentes soluções.

Analisando o desempenho do grupo 1 e do grupo 2 no HMI verifica-se que o primeiro executa a tarefa mais facilmente, mesmo apesar dos elementos do segundo grupo serem bastante experientes no uso da plataforma. Este resultado pode ser justificado pelo facto dos indivíduos do grupo 2 efetuarem as tarefas com automatismos e vícios já adquiridos durante o uso prolongado do HMI, que por vezes não se aplicavam no teste em causa. Estes vícios eram mais notados na questão da navegação, na qual o HMI está completamente dependente do conteúdo

das páginas, não existindo controlos que permitam navegar para uma determinada página, tendo outra distinta como ponto de partida. Por exemplo, o utilizador para ir da primeira página para a quarta página (considerando que existem 10) ou utiliza um atalho de navegação em alguma página ou então necessita carregar 3 vezes no botão "Próxima página". Os utilizadores do grupo 2, provavelmente por estarem habituados a ter constantemente atalhos nas páginas, perdiam algum tempo a navegar para a página indicada. Já os utilizadores do grupo guiavam-se constantemente pela página inicial que interiorizavam como a de Menu. Para melhor compreender e interpretar estes resultados seria necessário um estudo futuro mais aprofundado e com maior número de utilizadores.

Em relação à percentagem de sucesso de execução o valor é igual para o grupo 1 (100%) em ambos os testes, sendo que os elementos do grupo 2 tiveram mais dificuldade em executar com sucesso as mesmas tarefas no Simulation.Touch. Nesta tarefa foi registado um sucesso de 83%, contrastando com 100% de eficácia na execução do teste no HMI (Tabela 10 e Tabela 11).

	T2 TT	T2 TSS	T2 TD	% TSS	Sucesso
Média Grupo 1	24,83	2,83	01:43	11%	100%
Média Grupo 2	29,40	3,40	02:01	12%	83%
Desvio Padrão Grupo 1	3,18	1,86	00:13		
Desvio Padrão Grupo 2	6,50	1,50	00:27		
Média total	27,12	3,12	01:52	11%	92%

Tabela 10 – Dados relativos à execução do teste na aplicação Simulation.Touch

	T3 TT	T3 TSS	T3 TD	% TSS	Sucesso
Média Grupo 1	27,33	3,50	01:43	13%	100%
Média Grupo 2	29,83	5,17	01:51	17%	100%
Desvio padrão Grupo 1	5,12	3,40	00:24		
Desvio padrão Grupo 2	4,22	3,08	00:17		
Média Total	28,58	4,33	01:47	15%	100%

Tabela 11 – Dados relativos à execução do teste na aplicação HMI

Outro dado recolhido foi o referente ao número total de interações na interface e a percentagem destas que foi inválida ou sem sucesso (Figura 45). Os dados revelam que, em ambos os grupos, houve um registo de menor número de interações totais no Simulation.Touch do que no HMI, sendo que o mesmo se verifica nos toques sem sucesso. No que diz respeito ao grupo 1 houve uma diferença observável no número de interações totais (24.83 vs. 29.40) sendo a percentagem de interações sem sucesso também favorável ao Simulation.Touch (11% vs. 13%). Em relação ao grupo 2 verifica-se uma diferença pouco substancial no número de

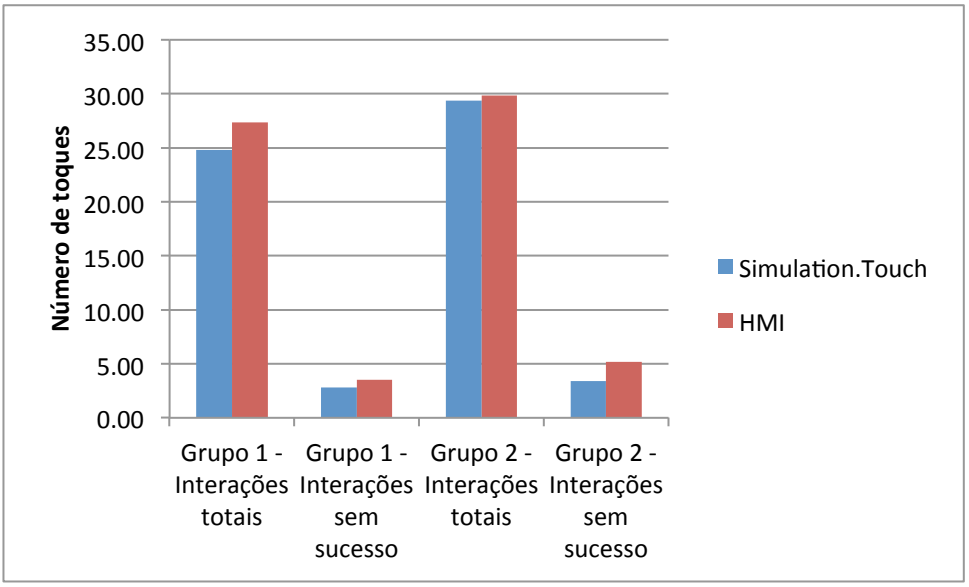


Figura 45 – Registo do número de toques totais e toques sem sucesso em cada teste nas duas aplicações

interações totais (29.40 vs. 29.83), sendo que a diferença na percentagem de interações sem sucesso é bastante elevada (12% vs. 17%). Analisando os valores médios (Tabela 11) também se nota a mesma tendência de haver menor diferença no número de interações totais (27.12 vs. 28.58) do que na percentagem de interações erradas (11% vs. 15%).

A análise dos dados leva a concluir que na solução Simulation.Touch os utilizadores de ambos os grupos conseguem realizar a mesma tarefa recorrendo a menos interações, sendo que a percentagem destas que são sem sucesso é bastante superior no HMI. Há que salientar que a redução da percentagem de interações sem sucesso no Simulation.Touch indica uma experiência de utilização mais eficiente. Para além disso, o grupo 1, que não tinha conhecimento de ambas as plataformas executou as tarefas praticamente com o mesmo tempo despendido e com a mesma percentagem de sucesso.

Se juntarmos todos estes aspetos analisados, temos fortes indícios de uma utilização mais eficiente da aplicação Simulation.Touch do que da plataforma HMI.

5.4.3 Questionário

No que respeita à aplicação do questionário sobre a dimensão "Facilidade de utilização e de execução das tarefas" (Figura 47) os resultados demonstram que os indivíduos consideraram

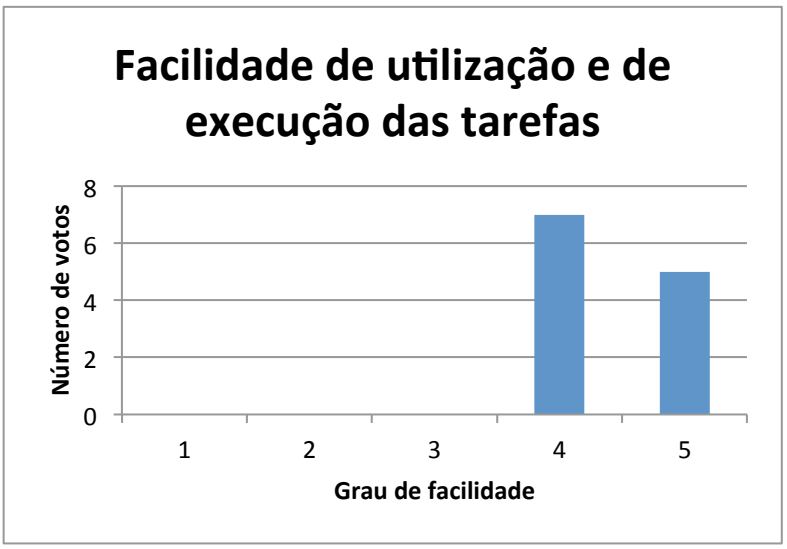


Figura 46 – Gráfico da preferência dos utilizadores relativo à facilidade de execução e utilização da aplicação

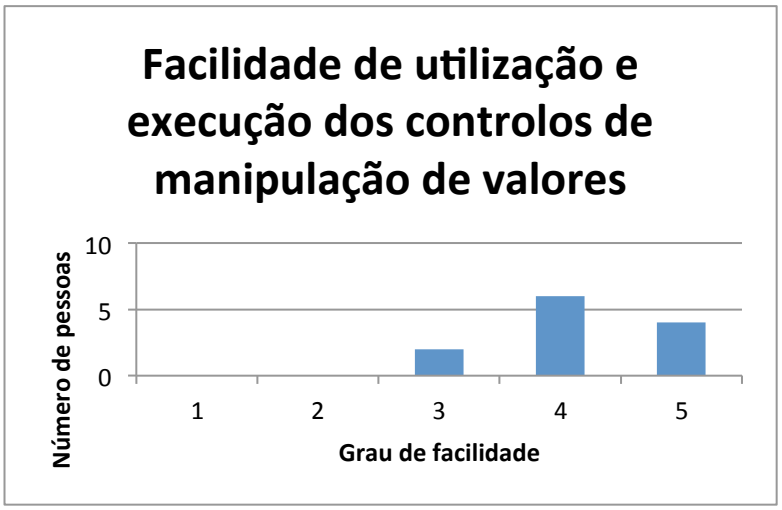


Figura 47- Gráfico da preferência dos utilizadores relativo à facilidade de utilização de controlos embebidos de página

a aplicação Simulation.Touch fácil de utilizar e as tarefas propostas de simples execução (7 votos de valor 4 e 5 de valor 5). Os utilizadores foram também questionados relativamente à facilidade de utilização dos controlos embebidos de página (*TouchSlider*, *RotateSlider*, *ToggleButton*, etc.) (Figura 46). Em relação a este aspeto o consenso não foi tão alargado, registando-se ainda assim um valor bastante positivo (3 votos no valor 3, 6 no 4 e 4 votos no valor 5).

No que concerne à avaliação dos controlos de interface (suporte a navegação e exploração) foi questionado se no decorrer do teste tinham utilizado algum destes componentes. Todos os indivíduos de ambos os grupos responderam afirmativamente, podendo-se concluir portanto que os controlos funcionaram como um suporte positivo e essencial na execução dos testes.

Analisando os dados relativos à escolha de plataforma pelos elementos do grupo 1 (Figura 48) verifica-se que a maioria das preferências se situa do lado do Simulation.Touch. No que diz

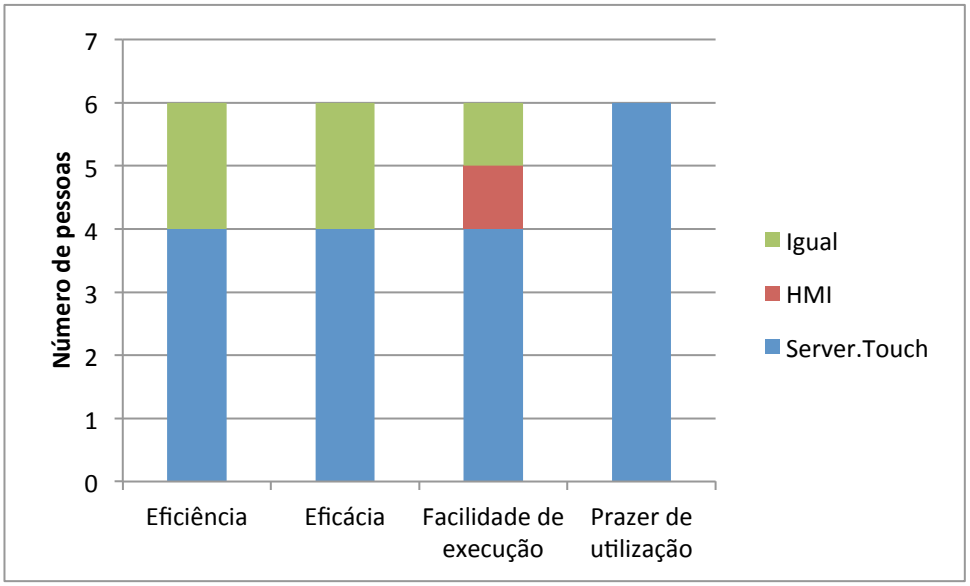


Figura 48 - Respostas dos utilizadores do grupo 1 em relação a aspetos subjetivos de utilização

respeito à eficiência, 4 indivíduos preferiram o Simulation.Touch e 2 acharam-na equivalente ao HMI. Em relação à eficácia verificou-se a mesma situação. No aspeto da facilidade de execução, o Simulation.Touch continua a ser o mais escolhido (4 utilizadores), apesar de também já haver alguma preferência pelo HMI (1 utilizador). Ainda relativo a este aspeto houve um utilizador que achou que a execução era semelhante nas duas plataformas. No capítulo do prazer de utilização a escolha do Simulation.Touch foi unânime.

Se analisarmos os dados referentes aos mesmos aspetos respondidos pelo grupo 2 (Figura 50), verificamos que a tendência não é tão acentuada, verificando-se ainda assim uma preferência maioritária pelo Simulation.Touch. Analisando o aspeto da eficiência verifica-se que 4 pessoas escolheram o Simulation.Touch e 2 preferiram o HMI. Em relação à eficácia houve 3 pessoas que gostaram mais do Simulation.Touch, 2 do HMI e uma achou que se comportam de forma semelhante. No que diz respeito à facilidade de execução o Simulation.Touch foi o que teve menor expressão (1 escolha), seguido pelo HMI (2 escolhas), tendo a maioria dos utilizadores achado que as soluções são semelhantes neste aspeto. Por fim, em relação ao prazer

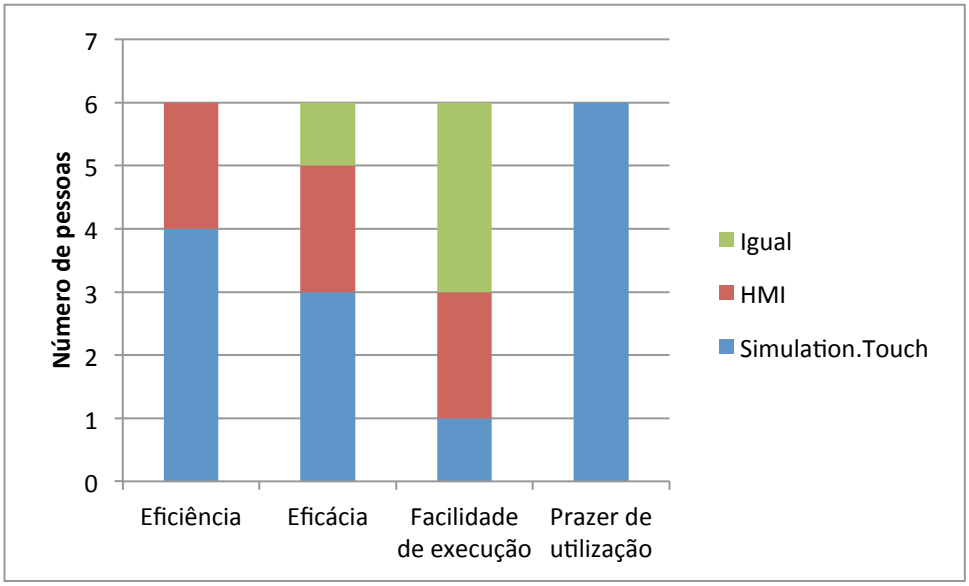


Figura 50 – Respostas dos utilizadores do grupo 2 em relação a aspetos subjetivos de utilização

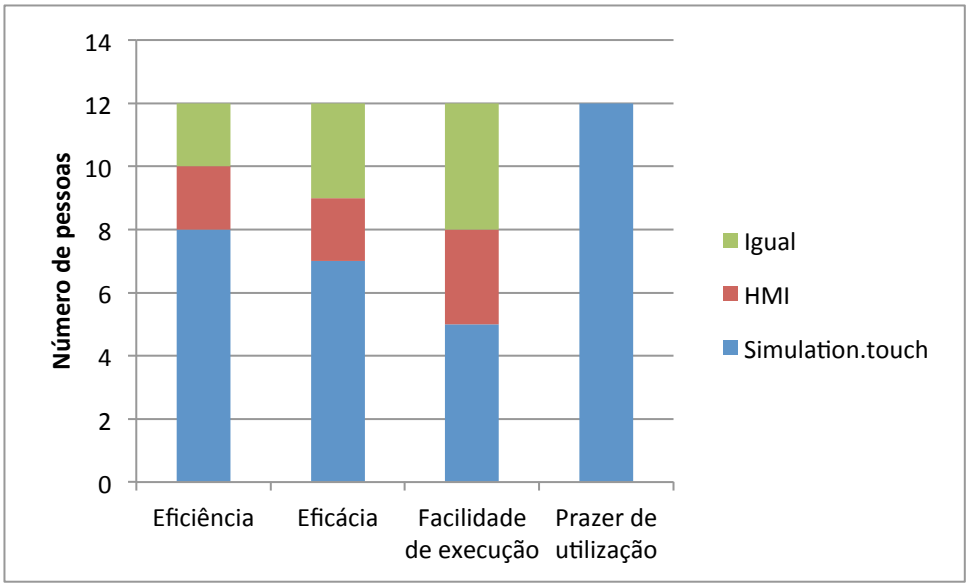


Figura 49 – Respostas dos utilizadores de ambos os grupos em relação aos aspetos subjetivos de utilização

da utilização a resposta foi semelhante à do grupo 1 tendo o Simulation.Touch recolhido a totalidade das escolhas.

Após a análise dos dados (Figura 48, Figura 50 e Figura 49) pode-se concluir que os indivíduos que não têm grande conhecimento das duas plataformas (grupo 1) preferiram

maioritariamente a solução Simulation.Touch, tendo a solução HMI recebido apenas uma escolha de um utilizador na totalidade dos 5 aspetos em causa. Esta análise demonstra que a aplicação é, do ponto de vista do utilizador, mais eficaz e eficiente, sendo mais fácil de executar e havendo mais prazer na sua utilização do que no HMI.

Em relação ao grupo 2 as escolhas continuaram a estar do lado do Simulation.Touch não havendo ainda assim uma maioria tão expressiva. O facto de os utilizadores, que estão bastante habituados ao HMI, terem achado a solução Simulation.Touch mais eficiente e eficaz é um aspeto que deve ser realçado. Além disso, o prazer de utilização foi unânime nos dois grupos o que valida o paradigma multitoque como um tipo de interação que fornece uma experiência de utilização mais enriquecedora. O facto de a facilidade de execução não reunir maioria do lado do Simulation.Touch pode ser justificada pelos automatismos naturais adquiridos pelos utilizadores do grupo 2 após anos de uso da plataforma HMI.

Capítulo 6

6. Conclusões

O trabalho descrito nesta dissertação consistiu no desenvolvimento de uma aplicação multitoque no contexto da automação industrial. Teve como objetivo principal expandir um módulo existente numa suite de software da Efacec, o Automation Studio, aplicando os paradigmas de interface e de interação apropriados. No decorrer do trabalho foram realizados testes de usabilidade de forma a verificar e validar a usabilidade e fiabilidade da solução, bem como medir a eficácia e eficiência comparando com uma solução real profissional. A aplicação foi desenvolvida recorrendo à tecnologia WPF (versão 4.0) e utilizando o SDK do Microsoft Surface (versão 2.0).

6.1 Conclusões do trabalho

Os objetivos propostos foram satisfeitos na totalidade, tendo sido implementado um protótipo funcional demonstrável de uma solução de automação baseada em interação multitoque. Este protótipo foi implementado recorrendo à utilização de paradigmas de interface apropriados a uma interação com o toque.

Para favorecer a interação e a experiência de utilização foram desenvolvidos controlos de interface e controlos embebidos nas próprias páginas. Estes controlos são um suporte importante na execução das tarefas, funcionando como um complemento à exploração e navegação.

A aplicação foi desenvolvida de forma a ser flexível e costumizável, aumentando o poder de configuração da plataforma e tornando o utilizador um elemento importante no processo de desenvolvimento.

Este projeto foi sujeito a testes com utilizadores-alvo, tendo recolhido opiniões maioritariamente positivas. Analisando os resultados verifica-se que, no grupo de pessoas que têm pouco ou nenhum conhecimento das duas plataformas, o tempo de execução da tarefa foi praticamente o mesmo, sendo o sucesso de execução igual no Simulation.Touch e no HMI. A este fator junta-se o facto de, em ambos os grupos, o número de toques totais e percentagem de toques sem sucesso ser inferior no Simulation.Touch. Destes aspetos podemos concluir que existem indícios de que a plataforma Simulation.Touch é mais eficiente do que a solução HMI, havendo uma diferença mínima no tempo de execução. Esta conclusão valida a tese de que uma solução multitoque demonstra ser mais eficiente que uma solução baseada em rato e teclado no contexto da automação.

No que diz respeito às métricas definidas inicialmente os testes revelaram que dos quatro aspetos analisados três passaram com sucesso, sendo que na métrica restante a diferença em relação ao estabelecido é mínima.

Quanto à experiência de utilização, as preferências estiveram maioritariamente do lado do Simulation.Touch, mesmo no grupo dos utilizadores frequentes do HMI. O facto destes utilizadores terem achado a solução Simulation.Touch mais eficiente e eficaz é um aspeto que deve ser devidamente realçado. Além disso, o prazer de utilização foi unânime nos dois grupos o que confirma que o multitoque oferece uma experiência de utilização mais enriquecedora.

O trabalho desenvolvido contribui para a área do multitoque e da automação com a descrição de um caso experimental e do resultado do mesmo. Conclui-se que a aplicação destes paradigmas de interação na área da automação industrial é perfeitamente viável existindo melhoria na produtividade, eficiência e na experiência de utilização.

6.2 Perspetivas de desenvolvimento futuro

A aplicação desenvolvida teve sempre como objetivo constituir-se como um protótipo demonstrável funcional. Nesse sentido existem determinados aspetos a serem aperfeiçoados no futuro, no sentido de tornar a aplicação funcional a nível comercial.

Do nosso estudo salientamos a necessidade de resolução da ambiguidade detectada na seleção e exploração da interface por parte dos utilizadores. Por exemplo, o facto de os controlos passarem a ter 10% de visibilidade após dez segundos de inatividade torna-os de difícil deteção caso a página que se encontra no fundo tenha uma cor clara.

Seria ainda interessante melhorar a usabilidade da aplicação no que diz respeito ao *feedback* de certos controlos ou botões com o objetivo de aumentar a intuição e a facilidade de compreensão.

Pretende-se que este projeto evolua para outras plataformas tal como *tablets*, telemóveis num contexto de monoutilizador ou mesmo num paradigma de multiutilizador. Esta última

solução obrigaria a uma reformulação de interação no sentido de a aplicação estar disponível numa mesa na horizontal, de forma a os utilizadores interagirem em seu redor.

Referências

Abacus. (2011). "Abacus - Automation Innovation." Retrieved 2011-07-10, from http://www.abacus-automation.co.za/products.php?productID=6&catagoryID=4.

Ahlberg, C., C. Williamson, et al. (1992). Dynamic queries for information exploration: An implementation and evaluation. Proceedings of the SIGCHI conference on Human factors in computing systems**:** 619-626.

AI4US. (2011). "Chatbot Anna." Retrieved 2011-06-17, 2011, from http://www.chatbots.org/virtual_assistant/anna_spain/.

Albinsson, P. (2003). High precision touch screen interaction. Proceedings of the SIGCHI conference on ….

Bachl, S., M. Tomitsch, et al. (2010). Challenges for Designing the User Experience of Multi-touch Interfaces.

Bailly, G., A. Demeure, et al. (2008). MultiTouch menu (MTM). Proceedings of the 20th International Conference of the Association Francophone d'Interaction Homme-Machine**:** 165-168.

Bailly, G., E. Lecolinet, et al. (2007). Wave menus: improving the novice mode of hierarchical marking menus. Proceedings of the 11th IFIP TC 13 international conference on Human-computer interaction**:** 475-488.

Bederson, B., J. Hollan, et al. (1996). Pad++: A zoomable graphical sketchpad for exploring alternate interface physics. Journal of Visual ….

Benko, H. and A. Wilson (2006). Precise selection techniques for multi-touch screens. Proceedings of the SIGCHI ….

Besacier, G., G. Rey, et al. (2007). Paper metaphor for tabletop interaction design. Human-Computer Interaction. Interaction Platforms and Techniques**:** 758-767.

Bier, E., M. Stone, et al. (1993). Toolglass and magic lenses: the see-through interface. Proceedings of the 20th ….

Blackwell, A. (2006). The reification of metaphor as a design tool. ACM Transactions on Computer-Human Interaction (….

Brandl, P., J. Leitner, et al. (2009). Occlusion-aware menu design for digital tabletops. Proceedings of the ….

BumpTop. (2011). "BumpTop." Retrieved 2011-06-14, 2011, from http://en.wikipedia.org/wiki/BumpTop.

Buxton, B. (2007). Multi-touch systems that i have known and loved. Microsoft Research.

Callahan, J., D. Hopkins, et al. (1988). An empirical comparison of pie vs. linear menus. Proceedings of the ….

Carroll, J. M. (1997). Human-computer interaction: psychology as a science of design. Annual review of psychology. **48**.

Ciolfi, L. (2004). Understanding spaces as places: extending interaction design paradigms. Cognition.

Dina Goldin, S. A. S., Peter Wegner (2006). Interactive Computation - The new paradigm, Springer.

Dragicevic, P. (2004). Combining crossing-based and paper-based interaction paradigms for dragging and dropping between overlapping windows. Proceedings of the 17th annual ACM symposium on User interface software and technology**:** 193-196.

Dwyer, T., D. Fisher, et al. (2011). Understanding Multi-touch Manipulation for Surface Computing. … Notes in Computer ….

Fallman, D. (2003). Design-oriented human-computer interaction. Proceedings of the SIGCHI conference on Human ….

GE. (2011). "Proficy CIMPLICITY." Retrieved 2011-07-10, from http://www.ge-ip.com/products/2819.

Generoso. (2011). "MIRIA SDK - Multi device Input UI controls for Silverlight." Retrieved June, 2011, from http://miria.codeplex.com/.

Goldin, D. and S. Smolka (2006). Interactive computation: The new paradigm. books.google.com.

Gould, J. (1985). Designing for usability: key principles and what designers think. Communications of the ACM.

Gunawardana, A., T. Paek, et al. (2010). Usability guided key-target resizing for soft keyboards. Proceeding of the 14th international conference on Intelligent user interfaces**:** 111-118.

Hancock, M. (2004). Improving menu placement strategies for pen input. Proceedings of Graphics Interface 2004.

Hilliges, O., S. Izadi, et al. (2009). Interactions in the air: adding further depth to interactive tabletops. … symposium on User ….

Hinrichs, U., M. Hancock, et al. (2007). Examination of text-entry methods for tabletop displays. Horizontal Interactive Human-Computer Systems, 2007. TABLETOP'07. Second Annual IEEE International Workshop on**:** 105-112.

Hofmeester, K. (2010). Using metaphors to create a natural user interface for microsoft surface. Proceedings of the 28th of the ….

Hong, J. (2011). Matters of design. Communications of the ACM. **54**.

Hopkins, D. (1991). The Design and Implementation of Pie Menus**:** 1-4.

Hurtienne, J., C. Stößel, et al. (2010). Physical gestures for abstract concepts: Inclusive design with primary metaphors.

Hutchins, E. L., J. D. Hollan, et al. (1985). Direct manipulation interfaces. Human-Computer Interaction. **1:** 311-338.

ICONICS. (2011). "HMI/SCADA Genesis." Retrieved 2011-06-25, 2011, from http://www.iconics.com/Home/Products/HMI-and-SCADA.aspx.

Ishak, E. (2004). Interacting with hidden content using content-aware free-space transparency. Proceedings of the 17th annual ACM ….

Jeff Rubin, D. C. (2008). Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests, John Wiley & Sons Inc.

Jetter, H. C., A. Engl, et al. (2008). Zooming not zapping: Demonstrating the zoil user interface paradigm for itv applications. Adjunct Proceedings of European Interactive TV Conference.

Khoury, G. (2003). Elastic metaphors: expanding the philosophy of interface design. … conference on Computers and philosophy- ….

Kim, S., J. Kim, et al. (2006). Multi-touch interaction for table-top display. Advances in Artificial Reality

Kiriaty, Y. (2009). "MultiTouch Capabilities in Windows 7." Retrieved June, 2011, from http://msdn.microsoft.com/en-us/magazine/dd861346.aspx.

Kiriaty, Y. (2010). Introducing windows 7 for developers. diazdesantos.es.

Kleid, N. (1996). Handwriting Recognition and Soft Keyboard Study. Human Factors and Ergonomics.

Kurtenbach, G. (1993). The limits of expert performance using hierarchic marking menus. Proceedings of the INTERACT'93

L'Abbate (1998). The Metaphor Definition Tool. Citeseer.

Lafon, M. (2000). Instrumental interaction: an interaction model for designing post-WIMP user interfaces. citeulike.org.

Landay, J. (2001). Sketching interfaces: Toward more human interface design. Computer.

Lao, S., X. Heng, et al. (2009). A gestural interaction design model for multi-touch displays.

LCDS. (2011). "LAquis 3.7." Retrieved 2011-07-10, from http://www.lcds.com.br/laquis.asp.

Mackay, W. (2002). Which interaction technique works when?: floating palettes, marking menus and toolglasses support different task strategies. Proceedings of the Working Conference.

Marcus, A. (1998). Metaphor design for user interfaces. CHI 98 conference summary on Human factors.

Marinos, D., C. Geiger, et al. (2010). Multitouch navigation in zoomable user interfaces for large diagrams. ITS '10: International Conference on Interactive Tabletops and Surfaces.

Matejka, J., T. Grossman, et al. (2009). The design and evaluation of multi-finger mouse emulation techniques.

Microsoft. (2010). "Multi-Touch on Microsoft Surface and Windows 7 for .NET Developers." from http://www.microsoftpdc.com/2009/CL27.

Microsoft. (2011). "About Windows Touch." Retrieved June, 2011, from http://msdn.microsoft.com/en-us/library/dd371406(v=vs.85).aspx.

Microsoft. (2011). "Microsoft Expression Blend." from http://www.microsoft.com/expression/products/Blend_Top10Features.aspx.

Microsoft. (2011). "Microsoft Visual Studio." from http://www.microsoft.com/australia/visualstudio.

Morris, M. and J. Wobbrock (2010). Understanding users' preferences for surface gestures. Proceedings of Graphics.

Moscovich, T. (2006). Multi-finger cursor techniques. Proceedings of Graphics Interface 2006.

Moscovich, T. (2008). Indirect mappings of multi-touch input using one and two hands.

Norman, D. A. (1983). Design rules based on analyses of human error. Communications of the ACM. **26:** 254-258.

Olwal, A. and S. Feiner (2008). Rubbing and tapping for precise and rapid selection on touch-screen displays.

Potter, R. L., L. J. Weldon, et al. (1988). Improving the accuracy of touch screens: an experimental evaluation of three strategies. Proceedings of the SIGCHI conference on Human factors in computing systems**:** 27-32.

Rogers, Y. (2006). Moving on from weiser's vision of calm computing: Engaging ubicomp experiences. UbiComp 2006: Ubiquitous Computing.

Roudaut, A., G. Bailly, et al. (2009). Leaf Menus: Linear Menus with Stroke Shortcuts for Small Handheld Devices. Human-Computer Interaction–INTERACT 2009**:** 616-619.

Ryall, K., C. Forlines, et al. (2006). Experiences with and observations of direct-touch tabletops.

Seow, S. (2005). Information Theoretic Models of HCI: A Comparison of the Hick-Hyman Law and Fitts' Law. Human-Computer Interaction. **20:** 315-352.

Shackel, B. (2009). Human-computer interaction-Whence and whither? Interacting with computers.

Shackel, B. (2009). Usability – Context, framework, definition, design and evaluation. Interacting with computers. **21:** 339-346.

Sharp, H., Y. Rogers, et al. (2007). Interaction design: beyond human-computer interaction. Book.

Shneiderman, B. (1981). Direct manipulation: A step beyond programming languages (abstract only). Proceedings of the joint conference on Easier and more productive use of computer systems.(Part-II): Human interface and the user interface-Volume 1981**:** 143.

Soloway, E. and M. Guzdial (1994). Learner-centered design: The challenge for HCI in the 21st century. interactions.

Star, X. (2007). "Xerox Star." from http://www.thocp.net/hardware/xerox_star.htm.

Systems, O. (2011). "OPC HMI and SCADA Software for .NET Applications." Retrieved 2011-07-10, from http://www.opcsystems.com/.

Took, R. (1990). Surface interaction: a paradigm and model for separating application and interface. Proceedings of the SIGCHI conference on Human Computer Interaction.

Tse, E., C. Shen, et al. (2006). Enabling interaction with single user applications through speech and gestures on a multi-user tabletop.

Turk, M. (2000). Perceptual user interfaces (introduction). Communications of the ACM.

Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. Proceedings of the SIGCHI conference on Human Computer Interaction.

Valli, A. (2008). The design of natural interaction. Multimedia Tools and Applications.

Van Dam, A. (1997). Post-WIMP user interfaces. Communications of the ACM.
Van Merriënboer, J. J. G., P. A. Kirschner, et al. (2003). Taking the load off a learner's mind: Instructional design for complex learning. Educational Psychologist. **38:** 5-13.
Vogel, D. (2010). Occlusion-aware interfaces.
Vogel, D. and P. Baudisch (2007). Shift: a technique for operating pen-based interfaces using touch. Proceedings of the SIGCHI conference on Human factors in computing systems**:** 657-666.
Wang, F., X. Cao, et al. (2009). Detecting and leveraging finger orientation for interaction with direct-touch surfaces.
Wang, F. and X. Ren (2009). Empirical evaluation for finger input properties in multi-touch interaction. Proceedings of the 27th international conference on Human factors in computing systems**:** 1063-1072.
Weinschenk, S. and P. Jamar (1997). GUI design essentials: for Windows 95, Windows 3.1, World Wide Web. portal.acm.org.
Weiser, M. (1999). The computer for the 21 st century. ACM SIGMOBILE Mobile Computing
Weiser, M. and J. S. Brown (1996). The coming age of calm technology [1]. Xerox PARC. Retrieved July. **8:** 2007.
Wigdor, D., G. Penn, et al. (2007). Living with a tabletop: Analysis and observations of long term office use of a multi-touch table.
Wigdor, D., S. Williams, et al. (2009). Ripples: utilizing per-contact visualizations to improve user interaction with touch displays.
Wixon, D. (1985). Engineering for usability (panel session): lessons from the user derived interface. ACM SIGCHI Bulletin.
Wobbrock, J. and M. Morris (2009). User-defined gestures for surface computing.
Wu, M. (2003). Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays.
Wu, M., C. Shen, et al. (2006). Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces.
Yatani, K., K. Partridge, et al. (2008). Escape: a target selection technique using visually-cued gestures.
Zhao, S. and M. Agrawala (2006). Zone and polygon menus: using relative position to increase the breadth of multi-stroke marking menus.

Anexos

A. User-defined gestures

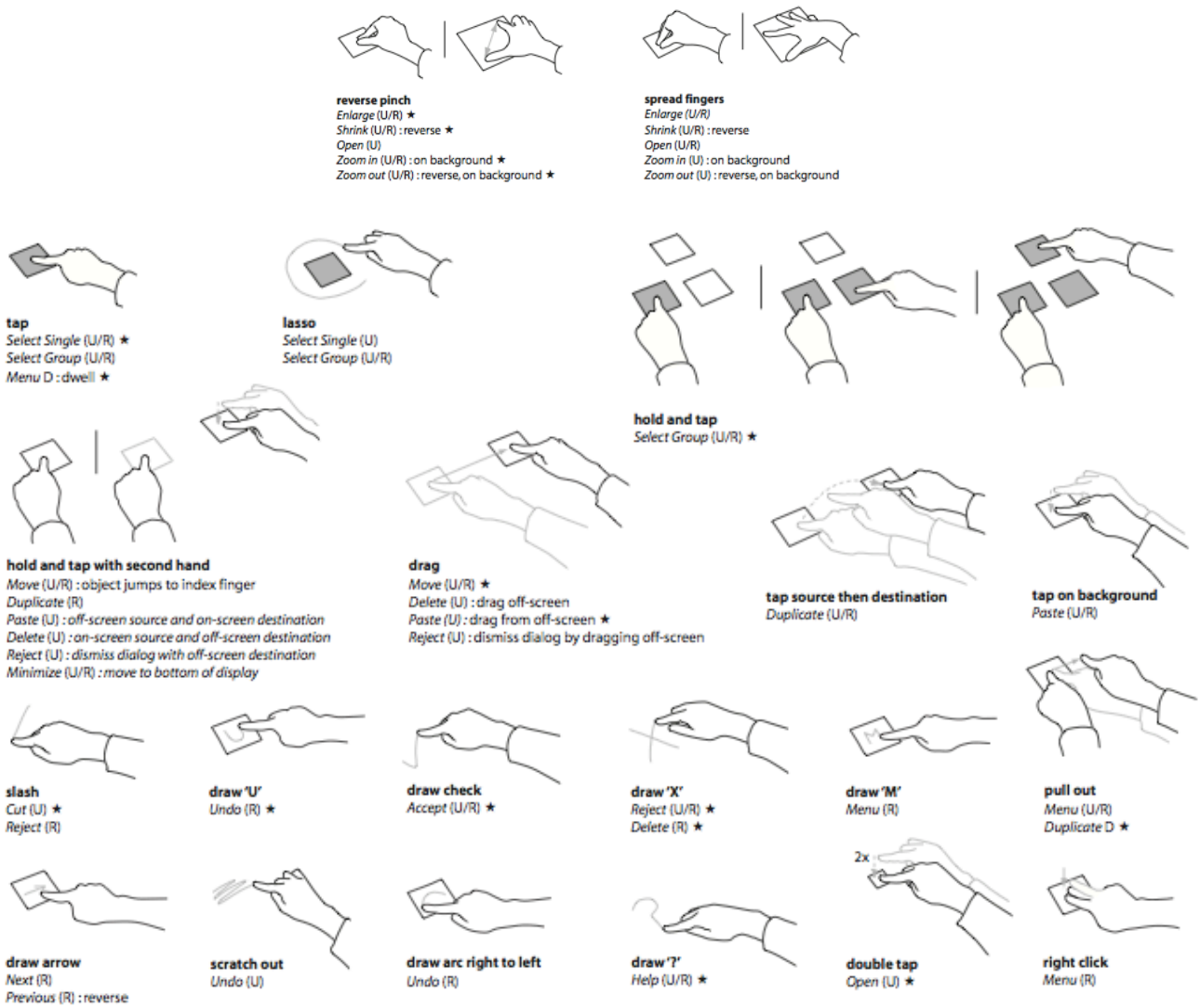


Figura 52 - User-defined gestures (lista completa em Wobbrock and Morris… 2009)

B. Guião Teste 1

NOTA: Irá surgir um som de alarme no decorrer do teste. Quando isso acontecer:

1. Abra a lista de alarmes
2. Aceite os alarmes com estado **Accepted** a **False**
3. Feche a lista de alarmes

Passos:

1. Navegar até página **Wastewater**
2. Fazer zoom na **Valve 2**
3. Centrar a **Valve 3** na interface.
4. Fazer ***Reset*** ao zoom
5. Navegar para página anterior (**Substation)**
6. Usando o controlo **Page Navigator,** Navegar para a última página (**Execute Control**)
7. Pressionar botão **OFF.**
8. Voltar à página inicial (**Menu**)

C. Guião Teste 2

Passos:

1. Abrir página **Eletric Vehicle**
2. Alterar o estado de **CHARGE** para **ON**
3. Esperar 5 segundos e de seguida pressionar **RUN,** confirmando a execução do controlo.
4. Pressionar **STOP** e confirmar execução do controlo.
5. Navegar para página **Gradients**
6. Alterar valor para **145** usando o **TouchSlider**
7. Pressionar **Other**. Inserir **95** (**usando Keypad**) e pressione **OK**.
8. Alterar valor para **0** usando **Rotate Slider**
9. Sair da aplicação

Executar a mesma tarefa em ambiente HMI:

1. Abrir página **Eletric Vehicle**
2. Pressionar **ON** e executar.
3. Esperar 5 segundos e de seguida pressionar **RUN.** Executar de seguida.
4. Pressionar **STOP** e executar.
5. Ir para página **Gradients**
6. Alterar valor para **145%**
7. Pressionar **Other**. Inserir **95**.
8. Alterar valor para **0 %.**
9. Sair da aplicação

D. Questionário

Grupo de utilizador

- ○ Grupo 1
- ○ Grupo 2

Já alguma vez tinha tido contacto com tecnologias multi-toque (excepto telemóveis ou tablets) ?

- ○ Sim
- ○ Não

Já tinha tido contacto com o HMI 500?

- ○ Sim
- ◉ Não

Facilidade de utilização e de execução das tarefas?

1	2	3	4	5
○	○	○	○	○

Facilidade de utilização e execução dos controlos de manipulação de valores (Touch Slider linear, Rotate Slider, etc.)?

1	2	3	4	5
○	○	○	○	○

Utilizou algum controlo de suporte à navegação (Toolbar, PageNavigator, etc.)?

- (●) Sim
- ( ) Não

Em relação à execução do último teste preferiu executar as tarefas no:

	Server.Touch	HMI	Igual
Em termos de eficácia	( )	( )	( )
Em termos de eficiência	( )	( )	( )
Prazer de utilização	( )	( )	( )
Facilidade de execução	( )	( )	( )

Comentários adicionais?