



LICENCIATURA SISTEMAS E TECNOLOGIAS DE INFORMAÇÃO

**PROCESSO DE MIGRAÇÃO DO SISTEMA OPERATIVO WINDOWS NA
STAPLES PORTUGAL**

Dissertação de Projeto Final de Licenciatura

Elaborado por Nuno Fernando Moreira da Silva

Discente nº 20131767

Orientador Professor Doutor José Braga de Vasconcelos

Barcarena

Junho de 2016

Universidade Atlântica

LICENCIATURA SISTEMAS E TECNOLOGIAS DE INFORMAÇÃO

**PROCESSO DE MIGRAÇÃO DO SISTEMA OPERATIVO WINDOWS NA
STAPLES PORTUGAL**

Dissertação de Projeto Final de Licenciatura

Elaborado por Nuno Fernando Moreira da Silva

Discente nº 20131767

Orientador Professor Doutor José Braga de Vasconcelos

Barcarena

Junho de 2016

Agradecimentos

O meu primeiro agradecimento é dirigido à equipa com quem trabalho todos os dias pela ajuda e suporte na realização deste projeto, nomeadamente o meu colega **Alfredo Almeida** e ao meu chefe e responsável do departamento **Paulo Marujo**.

Também e não menos importante, queria agradecer ao meu colega **Carlos Woodcook**, por me ter acompanhado neste desafio ao longo de 3 anos, não só pela partilha e trabalho desenvolvido, mas também pela relação de amizade criada fora do âmbito universitário.

Quero também agradecer ao meu colega **Paulo Carreteiro** pela paciência e por todo conhecimento que foi transmitido, não só a nível académico, mas também a nível profissional e pessoal.

Agradeço também ao meu orientador e professor **Dr. José Braga Vasconcelos**, por toda a sabedoria e indicações ao longo de todo o percurso académico e também no desenvolvimento da presente dissertação.

Para finalizar agradeço a todos os outros professores e colegas que de certa forma contribuíram direta ou indiretamente para o meu sucesso académico e tudo o resto.

O meu muito obrigado!

Resumo

Os sistemas de informação (SI) organizacionais têm vindo a ganhar importância na generalidade dos negócios, independentemente do tipo, tamanho ou localização dos SI. No setor dos serviços a indisponibilidade dos SI pode gerar uma quebra de receita muito significativa bem como uma diminuição da satisfação do cliente.

Para que não existam quebras do serviço prestado ao cliente final, é necessário implementar medidas técnicas, por forma a evitar falhas, erros e avarias no funcionamento dos SI que levam à interrupção do serviço prestado ao cliente final e à consequente perda de confiança no funcionamento do sistema de informação.

Adicionalmente, deve-se tomar medidas para garantir que o sistema esteja disponível, seja fiável, íntegro e de fácil manutenção. Todavia, as medidas consideradas não devem impedir que o SI seja flexível quanto à sua utilização, sob pena de o tornar tão restrito que possa não vir a servir o seu propósito.

Este trabalho propõe um artefacto que visa resolver o problema referido com base na utilização da metodologia de investigação-desenvolvimento (DSR), constituindo assim um contributo para se aumentar a confiança no funcionamento do sistema operativo Windows 7 instalado na generalidade dos computadores que dão suporte à prestação de serviços aos clientes das lojas da Staples em Portugal.

Palavras-chave: Confiança no Funcionamento, Tolerância a Falhas, Disponibilidade, Segurança, Engenharia de Software

Abstract

Organizational Information Systems have been gaining importance in most businesses, regardless of type, size or location of the SI. In the service sector the unavailability of SI can generate a significant revenue breaks and a decrease in customer satisfaction.

To avoid service breaks to the customer, its necessary implement technical measures in order to prevent failures, errors and malfunctions in the operation of SI that lead to the interruption of service to the end customer service and the consequent loss of confidence in the functioning of Information system.

In addition, you should take steps to ensure that the system is available, is reliable, integrate and easy to maintain. However, the proposed measures should not prevent the SI is flexible in its use, failing to make it so restricted that it cannot come to serve its purpose.

This paper proposes an artifact that aims to solve the problem based on the use of research-development methodology (DSR), as well as a contribution to increase confidence in the functioning of the Windows 7 operating system installed in most computers that support provide services to customers of Staples stores in Portugal.

Keywords: Dependability, Fault Tolerance, Availability, Security, Software Engineering

Índice

Declaração	3
Agradecimentos	4
Resumo	5
Abstract	6
Índice	7
Índice de Figuras	9
Índice de Tabelas	10
Lista de Acrónimos	11
1. Introdução	12
1.1. Enquadramento	12
1.2. Relevância do Tema	13
1.3. Objetivos e Contributos	14
1.4. Metodologia de Investigação	15
1.4.1. Design Science Research	15
1.4.2. Fases da Investigação	16
1.4.3. Identificação e Motivação do Problema	17
1.4.4. Definição de objetivos para uma solução	17
1.4.5. Conceção e Desenvolvimento	17
1.4.6. Demonstração	18
1.4.7. Avaliação	18
1.4.8. Comunicação	18
1.5. Organização da Dissertação	18
2. Estado da Arte	18
2.1. Confiança no funcionamento	19
2.1.1. Definições	19
2.1.2. Desafios	20
2.1.3. Como obter Confiança no Funcionamento	22
2.1.4. Atributos Confiança no Funcionamento	24
2.1.5. Como medir a confiança no funcionamento	25
2.2. Tipos de abordagens	25
2.2.1. Restritiva	26
2.2.2. Permissiva	26
2.3. Cópias de Segurança	26

2.4. Virtualização	26
2.4.1. VHD	27
2.5. Recuperação	28
2.5.1. Recuperação para trás	28
2.5.2. Recuperação para a frente	28
2.6. Engenharia de Software	29
2.6.1. SWEBOK	29
3. Análise de Requisitos	33
3.1. Requisitos Funcionais	33
3.2. Requisitos Não Funcionais	33
4. Modelação	34
4.1. Casos de Uso	34
5. Arquitetura	36
6. Desenvolvimento	37
6.1. Introdução	37
6.2. Metodologia	37
6.3. Tecnologias	38
6.3.1. Visual Studio	39
6.3.2. C#	39
6.3.3. VB .NET	39
6.3.4. Shell script	39
6.3.5. Windows Automated Installation Kit	40
6.3.6. Microsoft Diagnostics and Recovery Toolset	40
6.4. Interface Homem-Máquina	40
6.5. Código fonte	42
7. Testes e Resultados	46
7.1. Introdução	46
7.2. Implementação e testes	46
7.3. Resultados	46
8. Conclusão	48
9. Bibliografia	49

Índice de Figuras

Figura 1 - Framework IS Research (Hevner, March, Park, & Ram, 2004)	16
Figura 2 - Modelo de processos da metodologia DSR (Peppers, Tuunanen, Rothenberger, & Chatterjee, 2007).....	17
Figura 3 - Conceito de confiança no funcionamento. (Lopes, 2004).....	19
Figura 4 - Classes elementares de falhas (Lopes, 2004)	21
Figura 5 - Modos de avaria (Lopes, 2004)	22
Figura 6 - Transição entre estados do Sistema - Adaptado (Lopes, 2004)	25
Figura 7 - Recuperação para trás (Pullum, 2001)	28
Figura 8 - Recuperação para frente utilizando processos redundantes (Pullum, 2001)..	29
Figura 9 - Tópicos da Área de Conhecimento - Requisitos de Software (Bourque & Fairley, 2014).....	30
Figura 10 - Tópicos da Área de Conhecimento - Desenho de Software (Bourque & Fairley, 2014).....	31
Figura 11 - Tópicos da Área de Conhecimento - Construção de Software (Bourque & Fairley, 2014).....	31
Figura 12 - Tópicos da Área de Conhecimento – Testes do Software (Bourque & Fairley, 2014).....	32
Figura 13 - Tópicos da Área de Conhecimento – Manutenção de Software (Bourque & Fairley, 2014).....	32
Figura 14 - Caso de Uso Consola de Gestão	34
Figura 15 - Caso de Uso CD/DVD de Manutenção.....	35
Figura 16 - Caso de Uso Administração do Sistema	35
Figura 17 - Arquitetura do Sistema.....	36
Figura 18 - Modelo em Cascata.....	37
Figura 19 - Modelo Incremental (Pressman, 2010)	38
Figura 20 - Consola de Administração	40
Figura 21 - Menu Principal	41
Figura 22 - Menu de Contexto.....	41
Figura 23 - Configuração atualizações.....	41
Figura 24 - Tabela resultados	41
Figura 25 - Interface CD/DVD arranque	42
Figura 26 - Estado proteção disco.....	42
Figura 27 - Método runRemoteCommand	43
Figura 28 - Método serviceControl.....	43
Figura 29 - Método readRemoteRegistryAV	43
Figura 30 - Excerto código fonte serviço.....	44
Figura 31 - Excerto código fonte script CD/DVD.....	44
Figura 32 - Excerto script DISKPART_IMAGE_APPLY.CMD.....	45
Figura 33 - Excerto script REBUIL_SCRIPT.CMD	45

Índice de Tabelas

Tabela 1 - Incidências nos Computadores das Lojas	47
---	----

Lista de Acrónimos

SI – Sistemas de Informação

DSR - Design Science Research

SWEBOK - Software Engineering Body of Knowledge

RAID - Redundant Array of Inexpensive Disks

SLED - Slow Large Expensive Disk

VHD - Virtual Hard Disk

IEEE - Institute of Electrical and Electronics Engineers

AIK - Automated Installation Kit

SO – Sistema Operativo

IDE - Integrated Development Environment

DISM - Deployment Image Servicing and Management

WinPE - Windows Preinstallation Environmen

PE - Preinstallation Environmen

DART - Diagnostics and Recovery Toolset

MDOP - Microsoft Desktop Optimization Pack

IP - Internet Protocol

1. Introdução

Com a crescente dependência dos Sistemas de Informação para as Organizações e o facto de estes contribuírem diretamente para a sua missão, cada vez mais, é necessário garantir que não existam falhas a todos os níveis que possam influenciar os objetivos e os resultados cada vez mais exigentes num mundo globalizado em que a concorrência é enorme.

Com a utilização massiva da Internet é cada vez mais fácil para os indivíduos mal-intencionados atingirem os seus objetivos e por essa razão é necessário proteger os Sistemas de Informação a possíveis ataques internos ou externos, bem como a rápida recuperação em caso de sucesso destes ataques ou mesmo de uma possível utilização incorreta dos colaboradores da organização. Os atributos disponibilidade, fiabilidade, segurança, integridade e capacidade de manutenção são essenciais para atingir este objetivo.

1.1. Enquadramento

A Staples Portugal iniciou a sua atividade em 1996, com a designação social de OFCEP – Office Centre Portugal. O Office Centre fazia parte do projeto de expansão que a Makro Auto Serviço Grossista, SA começou a desenvolver em meados de 1995. Nessa altura, o Grupo Makro pertencia maioritariamente ao Grupo Holandês SHV e também ao Grupo Alemão Metro.

Depois de analisados vários projetos, a Makro Portugal entendeu apresentar à SHV uma proposta no sentido de iniciar as atividades em Portugal de um negócio dedicado à distribuição de papelaria e material de escritório.

Assim, nasceu o Office Centre em Setembro de 1996, tendo no seu quadro inicial de Recursos Humanos colaboradores que pertenciam ao Grupo Makro, liderados por Carlos Maia, Diretor Geral desde a sua criação.

Esta loja veio trazer para Portugal o novo conceito “tudo num só espaço para o escritório aos melhores preços”. Num só espaço, os clientes profissionais passaram a encontrar uma vasta oferta de artigos de papelaria, informática e tecnologia, mobiliário, equipamento e material de escritório e serviços de cópia e impressão.

Atualmente a Staples conta com 34 lojas de atendimento ao cliente, 1 escritório central e 1 armazém na plataforma da Azambuja.

Todas as 34 lojas dispõe de serviços de atendimento ao cliente especializados, mais especificamente Impressão e Centro de Cópias.

Esta área do negócio em específico é capaz de responder às necessidades do cliente, proporcionando a este uma larga oferta de produtos e soluções de impressão, cópias, digitalização, carimbos, brindes personalizáveis, etc.

Alguns dos serviços supra mencionados pressupõem que sejam utilizados equipamentos internos tais como computadores e impressoras, e também dispositivos de armazenamento externos à organização, neste caso podem ser discos USB ou outro tipo de media pertencentes ao cliente e onde estão por exemplo os trabalhos que pretende imprimir ou outros que sejam necessários para a prestação do serviço.

Este elemento externo à organização poderá ser uma ameaça aos sistemas internos por várias razões, sendo que a principal é o facto de não termos qualquer tipo de controlo sobre a origem e o conteúdo destes dispositivos, podendo com isto provocar algum tipo de falha nos sistemas internos.

Para além dos elementos supra referidos, também existem preocupações com uma possível utilização menos cuidada por parte dos colaboradores da organização, estas alterações podem ser voluntárias ou involuntárias. Apesar destes terminais e a própria rede local estarem protegidas contra as possíveis ameaças, é possível aceder á internet para consulta da caixa de correio eletrónico da organização e também a um conjunto de sítios web permitidos pela política da organização. Mais uma vez isto pode levar a que pessoas ou grupos mal-intencionados consigam com sucesso provocar danos ou até mesmo colocar em causa a segurança dos dados quer da organização (internos) ou mesmo dos clientes (externos).

Existe também um elevado número de pedidos de suporte devido ao facto de algumas funcionalidades não estarem a funcionar como o esperado, isto pode ou acontece porque um terminal é acedido por vários colaboradores que por sua vez, com intenção ou não, podem alterar estas para um estado diferente do configurado pelos Administradores de Sistemas.

Outro aspeto fundamental é proporcionar a melhor experiência possível aos colaboradores e aos clientes da organização, para poder concretizar este objetivo é necessário que os sistemas estejam sempre operacionais e que o seu comportamento seja o esperado sem nunca quebrar as expectativas e com isto prestar um serviço eficaz e de qualidade, marcando assim a diferença pela positiva num mercado globalizado e cada vez mais capaz.

1.2. Relevância do Tema

No que concerne á relevância do tema em questão podemos afirmar que cada vez mais as organizações procuram que os sistemas de informação sejam eficientes e que não coloquem em causa o negócio, sendo que o objetivo é exatamente o contrário, estes devem ser a ponte para o sucesso.

Uma das principais preocupações é garantir a continuidade dos serviços e as respetivas medidas que garantam este objetivo.

No caso em específico da Staples Portugal, foi possível perceber que esta área é muito importante, não só devido ao número de incidentes provocados por falhas nos sistemas, mas também pelo facto de muitas vezes não ser possível satisfazer os pedidos dos clientes

em tempo útil, baixando assim a qualidade do serviço prestado, bem como a expectativa do cliente em relação á organização.

Por este ser um paradigma que afeta todas as organizações e até mesmo instituições de educação ou a nível pessoal, este tema pode ser de extrema relevância. Tem-se vindo a observar um crescente nível de ameaças informáticas que afetam os sistemas direta ou indiretamente, por esta razão é muito importante estudar e procurar uma solução que permita a continuidade e não quebra do serviço prestado ou a utilidade do sistema.

Este tema também pode ser aplicado ao nível das instituições de ensino, quantas vezes após a utilização de um determinado sistema e alterações deste, levam a que o utilizador seguinte tenha que pedir suporte informático pois a alteração efetuada afeta diretamente o comportamento esperado ou até mesmo uma quebra de serviço no caso de a alteração implicar uma reinstalação de um determinado Software ou Sistema Operativo.

1.3. Objetivos e Contributos

A principal motivação para este trabalho de investigação é poder resolver os problemas que advêm do facto dos sistemas estarem expostos a intervenientes externos á organização e ao mesmo tempo aumentar a qualidade de serviço prestada ao cliente interno e externo.

No fundo o que se pretende face ao paradigma é aumentar a fiabilidade dos sistemas bem como as expectativas dos seus utilizadores, ao mesmo tempo contribuindo para a segurança dos dados e a continuidade dos serviços.

É possível também que o trabalho desenvolvido possa ser aplicado em outros cenários, sendo que todas as organizações querem ao máximo reduzir o nível de incidências de um ou vários sistemas e ao mesmo tempo aumentar ainda mais o nível de fiabilidade destes.

No que concerne á Staples Portugal, o principal objetivo é a migração do Sistemas Operativos para uma versão que o fabricante Microsoft continue a suportar, atualmente não é possível realizar este objetivo pois a aplicação “Windows Steady State” utilizada para garantir a recuperação em caso de falha não é suportada pelo novo Sistema Operativo Windows 7.

O objetivo será criar um artefacto e a sua respetiva aplicação, recorrendo ao atual estado da arte e às ferramentas disponibilizadas pelo fabricante ou outras, tendo por foco a concretização dos objetivos supra referidos, contudo evitando custos desnecessários mas mantendo sempre a fiabilidade e capacidade de manutenção do implementado.

Deverá ser tido em consideração a larga escala da infraestrutura e uma possível implementação noutros países em que a organização está presente.

O principal objetivo é que os Sistemas estejam operacionais e que não exista lugar a quebras do serviço sejam quais forem as razões.

Espera-se também com este trabalho de investigação poder contribuir diretamente para os resultados da organização bem como para a sua missão perante os clientes. Para além disto espera-se reduzir o número de incidências no que concerne aos terminais em questão, contribuindo assim para uma melhor eficiência e produtividade dos seus utilizadores e ao mesmo tempo reduzindo custos no que concerne aos serviços afetados ou até mesmo aos recursos humanos ou número de horas necessários em caso de uma intervenção.

A atualização do Sistema Operativo irá dar a possibilidade à organização de usufruir das mais recentes novidades e compatibilidade com novas aplicações que deixaram de ser suportadas.

Também e se possível contribuir para outras com interesse na metodologia realizada e com objetivos iguais ou semelhantes, sendo o principal foco a alta disponibilidade dos Sistemas Operativos e a aumento da respetiva tolerância a falhas e com isso a confiança no seu funcionamento.

1.4. Metodologia de Investigação

1.4.1. Design Science Research

Como este trabalho final de curso é de natureza científica, surge a necessidade da utilização de uma metodologia que credibilize todo o trabalho realizado bem como os resultados apresentados. A abordagem metodológica que irei seguir por esta se adaptar melhor ao objetivo do trabalho é a *Design Science Research* (DSR).

Segundo (Hevner, March, Park, & Ram, 2004) podemos dizer que esta abordagem metodológica deve ser utilizada quando se pretende resolver problemas organizacionais identificados através da criação e avaliação de artefactos. Estes podem ser constructos (vocabulário e símbolos), modelos (abstrações e representações), métodos (algoritmos e práticas), instanciações (protótipos) ou teorias conceção melhoradas.

A DSR é caracterizada pela utilização de um método rigoroso na conceção de artefactos que resolvam os problemas observados pelo investigador e pela forma como se avaliam e comunicam os resultados. Já um método é por si só, o meio, o procedimento ou a técnica utilizada para realizar um processo de forma lógica, ordenada e sistemática.

No fundo podemos afirmar que a principal preocupação da DSR é a utilidade do artefacto produzido. Por esta razão devemos conhecer e compreender o problema, mitigando este com a construção e aplicação de um artefacto, depois de aplicado deve ser efetuada uma avaliação rigorosa.

No que concerne ao artefacto em si a inovação é crucial, isto porque o objetivo é resolver um problema até agora sem solução ou resolver este de forma mais eficaz e eficiente. Por esta razão devem ser apresentadas evidências que demonstrem qual a utilidade do novo artefacto, caso contrário este não fornece nenhum contributo para a resolução do problema organizacional.

Para assegurar que os resultados apresentados são reais e confiáveis irá ser utilizada uma *framework* que ajude a definir claramente o âmbito de uma investigação nos SI. A figura abaixo apresenta a *framework* conceptual para a execução e avaliação de uma investigação em Sistemas de Informação.

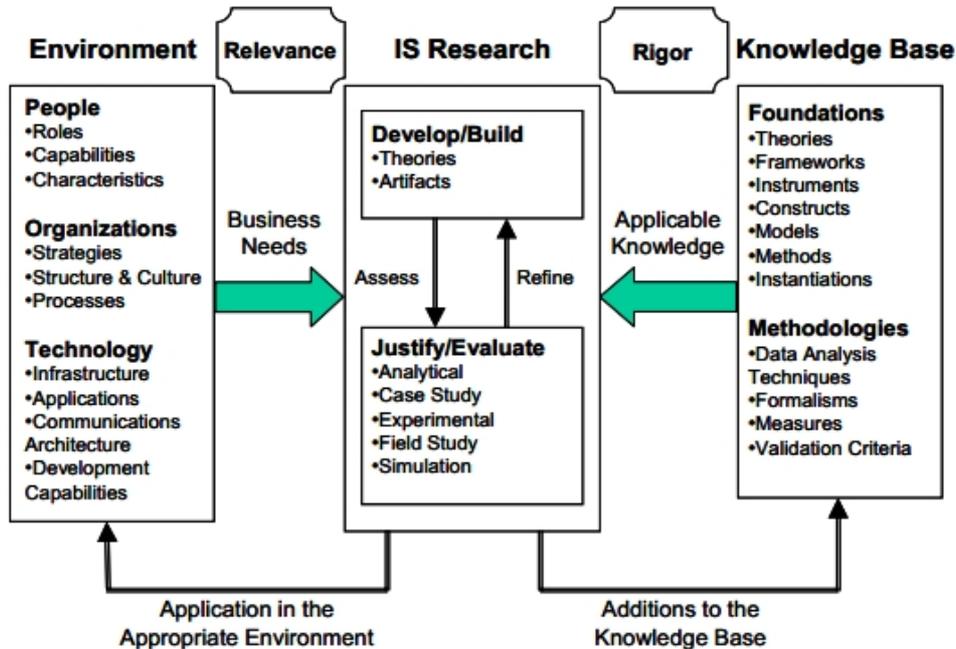


Figura 1 - Framework IS Research (Hevner, March, Park, & Ram, 2004)

Como se pode observar o ambiente é composto por pessoas, organizações e tecnologias, criando as necessidades de negócio originando o problema a resolver dando assim origem à investigação.

No processo de investigação são criados teorias ou artefactos que irão ser avaliados através de processos analíticos, experimentais, estudos de caso, simulações, etc. Este pode ser um processo iterativo dando origem a melhorias dos artefactos ou teorias criados previamente.

Para demonstrar rigor os processos devem ser sempre suportados por instrumentos de uma base de dados de conhecimento em qualquer uma das suas fases, validação ou avaliação, podendo estes ser fundamentos e ou metodologias. Na conclusão o processo para além de resolver o problema em questão, poderá também contribuir para o enriquecimento da base de dados, incrementando o conhecimento existente.

1.4.2. Fases da Investigação

(Peffer, Tuunanen, Rothenberger, & Chatterjee, 2007) propõe um modelo para o processo de DSR composto por seis atividades ou fases: Identificação e Motivação do

Problema, Definição de Objetivos para uma Solução, Conceção e Desenvolvimento, Demonstração, Avaliação e Comunicação.

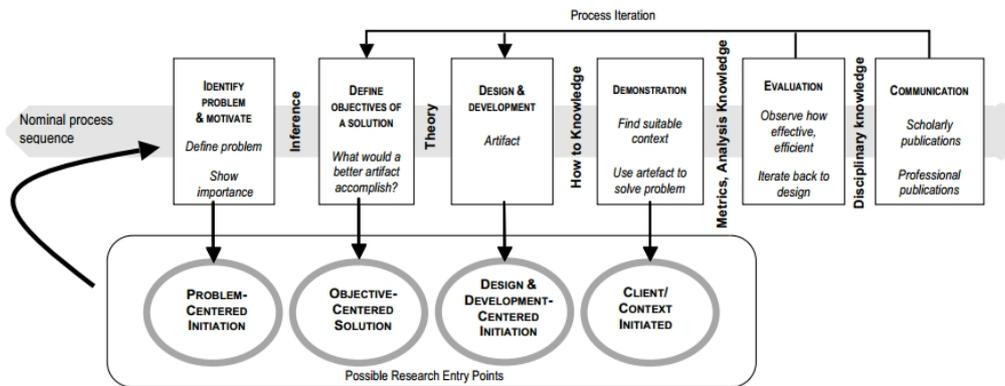


Figura 2 - Modelo de processos da metodologia DSR (Peppers, Tuunanen, Rothenberger, & Chatterjee, 2007)

1.4.3. Identificação e Motivação do Problema

Nesta fase deve-se definir o problema específico de investigação e justificar o valor de uma solução. Esta definição é utilizada para o desenvolvimento de um ou mais artefactos que poderão efetivamente fornecer uma solução. A justificação do valor da solução e da sua relevância para a área em questão requer um bom conhecimento do estado da arte na área do problema e da importância da sua solução. (Peppers, Tuunanen, Rothenberger, & Chatterjee, 2007)

1.4.4. Definição de objetivos para uma solução

Após a identificação do problema e da revisão da literatura e apuramento do estado da arte na área, devem-se explicitar os objetivos para a solução. Os objetivos podem ser quantitativos, tais como os termos em que a solução desejável seria melhor do que as atuais, ou qualitativos, como a descrição de como um novo artefacto deverá suportar soluções para problemas até então não abordados. Os objetivos devem ser inferidos racionalmente a partir do trabalho realizado na fase anterior, do conhecimento do estado da arte na área do problema e das soluções atuais, bem como da sua eficiência para a resolução do problema. (Peppers, Tuunanen, Rothenberger, & Chatterjee, 2007)

1.4.5. Conceção e Desenvolvimento

Nesta fase, concebe-se o artefacto. Esta atividade inclui a definição das funcionalidades desejadas para o artefacto e da sua arquitetura, e, em seguida, a criação do artefacto. (Peppers, Tuunanen, Rothenberger, & Chatterjee, 2007)

1.4.6. Demonstração

Neste momento faz-se a demonstração da utilização do artefacto para resolver uma ou mais instâncias do problema. A demonstração pode envolver a utilização do artefacto em experiências, simulações, casos de estudo, entre outras. (Peffer, Tuunanen, Rothenberger, & Chatterjee, 2007)

1.4.7. Avaliação

Nesta fase, avalia-se, observando e medindo, se o artefacto desenvolvido fornece uma solução para o problema. Esta atividade envolve a comparação entre os objetivos definidos e os resultados reais produzidos pelo artefacto na demonstração. Nesta fase, é necessário conhecer as métricas relevantes e as técnicas de análise. (Peffer, Tuunanen, Rothenberger, & Chatterjee, 2007)

1.4.8. Comunicação

Por último devemos comunicar o problema e sua importância, o artefacto, a sua utilidade e novidade, o rigor da sua conceção, e a sua eficácia na resolução do problema identificado na primeira atividade. (Peffer, Tuunanen, Rothenberger, & Chatterjee, 2007)

1.5. Organização da Dissertação

Por forma a facilitar a compreensão e leitura desta dissertação a mesma foi organizada em capítulos, sendo que cada um está relacionado com a metodologia de investigação adotada.

2. Estado da Arte

No que concerne à revisão da literatura e apuramento do estado da arte foram encontradas investigações, grupos de trabalho, conferências, etc, sobre o tema ou âmbito desta investigação.

Como já foi referido o objetivo deste trabalho é poder tornar um Sistema Operativo de computador mais fiável, seguro e tolerante a falhas, nunca reduzindo as suas capacidades perante o utilizador, no fundo o que se pretende é perceber qual o estado da arte referente a estas características.

2.1. Confiança no funcionamento

2.1.1. Definições

Um dos obstáculos encontrados no decorrer da revisão da literatura foi o facto de encontrar uma terminologia adequada para o termo original “*dependability*”. No entanto a adotada foi a mais consistente e adequada da terminologia associada.

A confiança no funcionamento ou “*dependability*” de um sistema é a propriedade desse que permite que um utilizador possa depositar uma confiança justificada no serviço que este disponibiliza. (Laprie, 1985)

Este conceito foi introduzido por existir a necessidade da análise de um sistema como um todo e não de forma individual, como por exemplo a análise com base nas características: disponibilidade, fiabilidade, segurança, etc.

No fundo a confiança no funcionamento de um determinado sistema de computação engloba todos estes aspetos e mais alguns, dos quais a prevenção, tolerância e tolerância a falhas como pode ser analisado na figura seguinte.

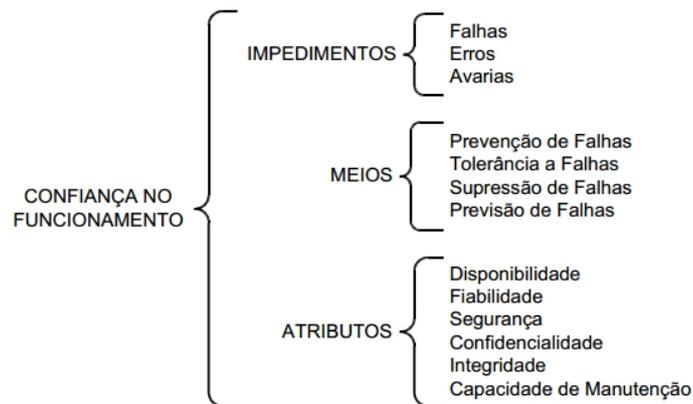


Figura 3 - Conceito de confiança no funcionamento. (Lopes, 2004)

Com base em (Lopes, 2004) a “confiança no funcionamento” de um sistema de computação pode ser analisada a partir de três grupos de fatores: o grupo dos Impedimentos, o grupo dos Meios e o grupo dos Atributos.

No que concerne aos Impedimentos, podemos dizer que estes constituem as causas que levam à falta de confiança no funcionamento do sistema, nomeadamente através da existência de falhas, erros ou avarias.

Os Meios são utilizados para aumentar o nível de confiança no funcionamento do sistema, através da prevenção, tolerância, supressão e previsão de falhas.

Os Atributos, tais como a disponibilidade, fiabilidade, segurança, confidencialidade, integridade e capacidade de manutenção, correspondem a fatores que permitem avaliar e caracterizar um determinado sistema.

Mais recentemente estes foram adicionadas a estes atributos o desempenho e a sobrevivência. No fundo o desempenho é o grau em que o sistema ou componente realiza as suas funções designadas dentro de determinados condicionalismos, tais como velocidade, precisão e uso de memória. Já a sobrevivência é a capacidade de um sistema para terminar uma determinada tarefa, em tempo útil, na presença de ataques, falhas ou acidentes. (Trivedi, Kim, Roy, & Medhi, 2009)

Segundo (Laprie, 1985), também é importante definir perante o âmbito os termos seguintes:

- O serviço prestado por um sistema é o seu comportamento tal como é percebido pelo seu utilizador;
- Um utilizador é um outro sistema (físico, humano) que interage com na primeira interface do serviço;
- A função de um sistema é que o próprio se destina a fazer, e é descrita pela especificação funcional;
- Serviço correto é entregue quando o este implementa a função de sistema;
- A falha do sistema é um evento que ocorre quando o serviço prestado se desvia do serviço correto;
- Uma falha é uma transição de serviço correto para serviço incorreto, ou seja, a não execução da função do sistema;
- A entrega de serviço incorreto é uma interrupção do sistema;
- A transição de serviço incorreto para serviço correto é a restauração do serviço.

Um sistema consiste num conjunto de componentes que interagem entre si, portanto, o estado do sistema é o conjunto do estado dos seus componentes. (Laprie, Avizienis, & Randell, 2000)

2.1.2. Desafios

Os desafios à confiança no funcionamento incluem a falhas, erros e avarias de *hardware* e/ou *software* pelo qual o sistema é composto.

2.1.2.1. Falhas

No que concerne às falhas estas podem ser classificadas perante os critérios representados na figura 2.

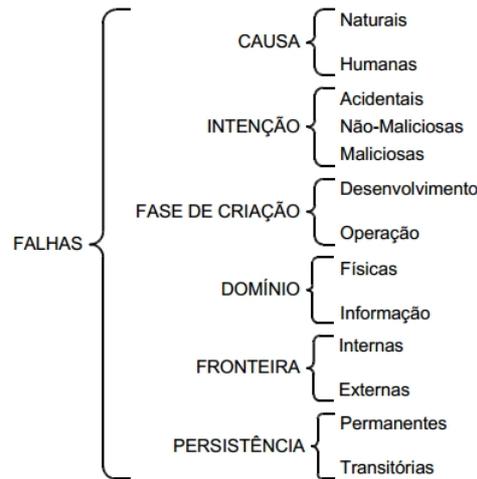


Figura 4 - Classes elementares de falhas (Lopes, 2004)

Sendo que quanto à causa estas podem ser naturais, quando resultam de fenómenos físicos ou humanas, quando resultam de erros humanos.

Quanto á intenção estas podem ser acidentais, quando aparecem ou são causadas sem qualquer intenção primária, não-maliciosas, quando são causadas com intenção mas em intuito de provocar danos no sistema, já as maliciosas são exatamente o oposto.

No que concerne à fase de criação as falhas podem ocorrer no período de desenvolvimento ou no decorrer da operação. Podem ser de domínio físico quando têm origem em fenómenos físicos ou de informação quando são originadas pelo processamento de informação. Já na fronteira estas podem ser classificadas como internas se têm origem interna ao sistema ou externas quando são provocadas por exemplo pelo utilizador através da interação ou interferência com o ambiente.

As falhas podem também ser de persistência permanente ou transitória. Permanentes quando não ocorrem de condições pontuais, sejam elas internas ou externas, já o contrário temos transitórias que têm uma duração limitada.

Apesar da classificação supra, não significa que uma falha tenha uma única classificação, sendo que na maioria dos casos uma falha é classificada em várias classes. Por exemplo uma falha de causa humana pode ser acidental e não ser persistente.

2.1.2.2. Erros

Um erro é um estado do sistema que é suscetível de conduzir a uma falha se não for corrigido em tempo útil. Estes podem ser permanentes, que é quando persistem até que seja tomada alguma ação, ou transitórios, estes são temporários e desaparecem sem ser necessário efetuar alguma medida.

Este ainda pode ter dois estados, latente, quando ainda não foi detetado ou detetado quando o mesmo foi intercetado e reconhecido como tal, podendo esta ação ser manual

ou automática. Também de referir que um erro pode gerar outros erros e propagar-se durante o seu ciclo de vida sem ser detetado na sua primeira fase.

2.1.2.3. Avarias

Podemos afirmar que não existem sistemas que nunca avariam, sendo por isso necessário classificar estes quanto ao modo de avarias e a sua respetiva gravidade.

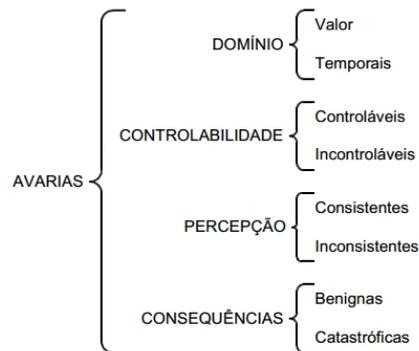


Figura 5 - Modos de avaria (Lopes, 2004)

Como se pode observar na figura 3, as avarias podem ser classificadas pelo seu domínio, controlabilidade, perceção e consequências. No que concerne ao domínio, este pode ser de valor, quando o valor lógico do serviço é o não esperado, e pode ser temporal, quando o serviço é prestado fora do intervalo esperado. Em termos de controlabilidade, temos as avarias controláveis que acontecem e têm um comportamento de forma esperada, e as incontroláveis, em que o acontecimento não era esperado e não está especificado. A perceção por parte do utilizador pode ser consistente ou inconsistente. Na consistente, todos os utilizadores têm a mesma perceção da avaria, o inconsistente é quando estes têm diferentes perceções da avaria. Já as consequências podem ser benignas, quando não existe prejuízo e catastróficas que é quando existe prejuízo maior do que os benefícios do correto funcionamento do sistema.

2.1.3. Como obter Confiança no Funcionamento

2.1.3.1. Prevenção

Para se prever uma falha e a consequente avaria de um sistema é necessário na sua fase de desenvolvimento garantir que todas as causas que a possam originar sejam removidas, isto pode ser concretizado através de um conjunto de técnicas ou tecnologias que evitem uma possível falha no sistema.

No que concerne ao desenvolvimento de *software* e a engenharia de software o guia mais utilizado e reconhecido é o SWEBOK. Este guia foi desenvolvido com os conhecimentos

adquiridos ao longo de 4 décadas e revisto por inúmeros profissionais e investigadores da área da Engenharia de Software, estabelecendo cinco objetivos principais.

Ao nível do hardware devem ser utilizados componentes de alta fiabilidade e se possível mecanismos que permitam a continuidade do serviço em caso de falha, por exemplo no caso do armazenamento de dados devem ser utilizado um conjunto redundante de discos, mais conhecido por RAID.

O RAID baseado em tecnologia de discos magnéticos desenvolvido para computadores pessoais, oferece uma alternativa atrativa do SLED, prometendo melhorias no que concerne à performance, fiabilidade, consumo de energia e escalabilidade. (Patterson, Gibson, & Katz, 1987)

2.1.3.2. Tolerância

A capacidade de tolerância a falhas pode ser em várias fases e tem por objetivo o aumento da confiança no funcionamento de um sistema.

As fases que o constituem são o processamento de erros, em que o objetivo é a deteção e remoção de erros antes de estes originarem um ou mais falhas. Já na fase de tratamento de falhas, o objetivo é impedir que estas reincidam.

No que concerne à implementação da tolerância a falhas estas podem ser classificadas como estruturais, temporais e informacionais. A estrutural está relacionada com a replicação da estrutura física ou lógica do sistema.

A temporal, com o repetido uso de um componente ou de uma sequência de ações. A informacional está relacionada com replicação de estruturas de informação (dados).

2.1.3.3. Supressão

A supressão de falhas é realizada no decorrer da conceção dos sistemas ou na operação deste. As atividades realizadas podem ser decompostas na verificação, diagnóstico e correção.

A supressão de falhas durante a fase operacional do sistema é realizada por intermédio de manutenção corretiva, que por sua vez pode ser decomposta em manutenção curativa e preventiva. (Lopes, 2004)

2.1.3.4. Previsão

A previsão das falhas que podem vir a ocorrer podem ser avaliadas qualitativamente ou quantitativamente.

Segundo (Lopes, 2004), esta avaliação pode ser realizada de duas formas:

- **Qualitativamente:** identificar, classificar e ordenar os modos de avarias do sistema; identificar as combinações de eventos (avarias de componentes ou condições ambientais) que podem levar a eventos indesejados;
- **Quantitativamente:** avaliar, em termos probabilísticos, alguns dos atributos da confiança no funcionamento, que por sua vez podem ser vistos como medidas da confiança no funcionamento.

As duas principais abordagens à previsão de falhas probabilística, tendo como objetivo a obtenção de estimativas (probabilísticas) da confiança no funcionamento, são a modelação e os testes de avaliação (ex. injeção de falhas).

2.1.4. Atributos Confiança no Funcionamento

No que concerne atributos da confiança no funcionamento estes podem ser os seguintes:

- **Disponibilidade:** prontidão para prestar um serviço correto;
- **Fiabilidade:** continuidade da prestação de um serviço correto;
- **Segurança:** ausência de consequências catastróficas para os utilizadores e o ambiente;
- **Confidencialidade:** ausência de divulgação não autorizada de informação;
- **Integridade:** ausência de alterações impróprias do estado do sistema;
- **Capacidade de manutenção:** aptidão para suportar reparações e modificações.

Alguns dos atributos anteriormente referidos podem quantificados e vistos como medidas da confiança no funcionamento.

No que concerne à segurança podemos dizer que os seus pilares são disponibilidade, confidencialidade e integridade. Estes estão listados separadamente pois o conjunto dos mesmos pode ser atingido ou não.

A única definição para a segurança poderia ser: a falta de acesso não autorizado, ou manipulação do estado do sistema. (Laprie, Avizienis, & Randell, 2000)

Temos também a integridade, em que só a podemos obter através da disponibilidade, fiabilidade e segurança.

A capacidade de manutenção pode ter diferentes abordagens, sendo que esta pode ser curativa, preventiva, corretiva, adaptativa e perfectiva. Na manutenção curativa o objetivo é mitigar as falhas que geraram os erros ou avarias. A preventiva é no fundo prever quais as falhas que poderão gerar erros e remover estas antes sequer de acontecerem. A corretiva tem por objetivo preservar ou melhorar a capacidade do sistema para fornecer o respetivo serviço. A adaptativa tem a função de se adaptar ao ambiente mediante as alterações. E por último a perfectiva tem por objetivo ir melhorando o sistema mediante *inputs* recebidos não só pelo cliente, mas também por quem concebe o sistema.

2.1.5. Como medir a confiança no funcionamento

Segundo (Laprie, Avizienis, & Randell, 2000) para medir a confiança no funcionamento de um sistema podemos utilizar os atributos disponibilidade, fiabilidade, segurança e capacidade de manutenção, sendo que estes atributos podem ser avaliados pontualmente ou num determinado espaço de tempo.

Para quantificar os atributos da confiança no funcionamento podemos segundo (Alves & Vasques, 1998) medir a fiabilidade como a probabilidade de o sistema estar a funcionar corretamente durante um determinado intervalo de tempo. A segurança como a probabilidade de que um sistema não sofrer uma avaria catastrófica dentro de um intervalo de tempo. A disponibilidade como a probabilidade de um sistema estar a funcionar corretamente e de estar disponível para executar a sua função no instante de tempo. A capacidade de manutenção é medida pela facilidade com que se efetua a manutenção de um sistema, no fundo o tempo que o sistema leva a ser restaurado em caso de avaria.

De uma forma abstrata segundo (Alves & Vasques, 1998) a vida de um sistema é percebida pelo seu utilizador como uma alternância entre dois estados do serviço prestado:

- **Serviço Correto** - onde o serviço prestado está conforme com a especificação;
- **Serviço Incorreto** - onde o serviço prestado não está conforme a especificação.

A transição entre o estado SC para SI pode ser provocada por falhas, erros e ou avarias, já o contrário é denominado de restauro do serviço onde pode ser aplicada a capacidade de manutenção.

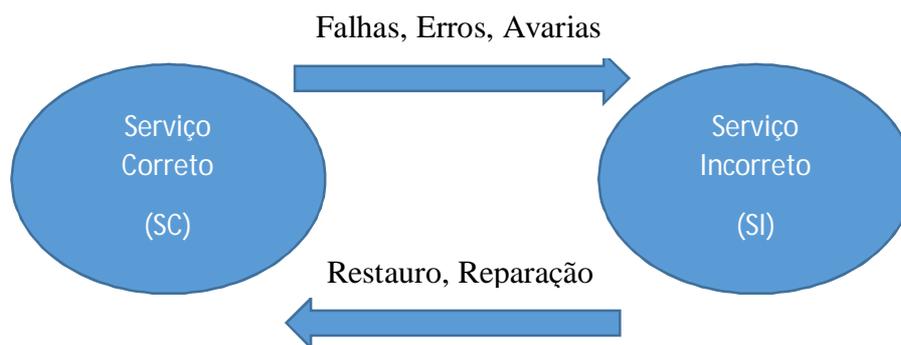


Figura 6 - Transição entre estados do Sistema - Adaptado (Lopes, 2004)

2.2. Tipos de abordagens

Existem dois tipos de abordagens mais comuns que os administradores de sistemas podem adotar no que concerne à gestão de estações de trabalho.

2.2.1. Restritiva

Nesta abordagem são utilizadas ferramentas que de alguma forma inibam alterações que possam prejudicar o bom funcionamento do sistema, contudo estas inibições podem alterar também o funcionamento expectável com ou sem consentimento do utilizador. Com este tipo de abordagem normalmente a manutenção é reduzida.

2.2.2. Permissiva

Numa abordagem permissiva os sistemas são mais flexíveis e menos restritos, no entanto a probabilidade de acontecerem erros que levem a falhas do serviço é maior. Para a recuperação são utilizados mecanismos de reinstalação e reconfiguração dos sistemas, podendo existir alguma indisponibilidade no decorrer destes processos.

2.3. Cópias de Segurança

A cópia de segurança (*backup*) protege os sistemas de ficheiros de erros do utilizador, disco ou outras falhas de *hardware*, erros de software que podem corromper o sistema de ficheiros e desastres naturais. A utilização mais comum das cópias de segurança são no restauro de ficheiros apagados acidentalmente por utilizadores ou para recuperar em caso de falha do disco rígido. (Chervenak, Vellanki, & Kurmas, 1998)

Apesar da cópia de segurança contribuir para a confiança no funcionamento de um sistema de informação, com o passar do tempo se não forem tomadas medidas que promovam a eficácia e eficiência, este processo pode tornar-se moroso e dispendioso no que concerne a recursos. No entanto existem diferentes abordagens e deve ser escolhida a que melhor se adapta ao ambiente.

2.4. Virtualização

É possível através da virtualização dos computadores reduzir drasticamente os custos de manutenção e melhorar a segurança utilizando as várias técnicas disponíveis em lugar de os ambientes tradicionais. (Hui, Seok, & Ki, 2015)

A virtualização de computadores pode ser classificada como cliente ou servidor, em que na primeira temos uma execução local e não partilhada, enquanto na segunda a execução das máquinas virtuais é remota e pode ser partilhada por várias instâncias que correm no mesmo *hardware*.

2.4.1. VHD

No âmbito desta investigação a virtualização pode ser utilizada para aumentar a confiança no funcionamento de um sistema, através da utilização de discos virtuais e a sua reposição em caso de desastre ou até mesmo a eliminação de alterações efetuadas pelos utilizadores, criando assim uma forma subtil de eliminar qualquer alteração que seja efetuada de forma voluntária ou involuntária.

Disco virtual ou *virtual hard disk* (VHD) é uma especificação de formato de imagem disponível publicamente que especifica um disco rígido virtual encapsulado num ficheiro único, capaz de hospedar sistemas de ficheiros nativos, apoiando operações padrão de disco e de ficheiros. (Microsoft Corporation, 2015)

Com a introdução deste conceito e através da capacidade dos sistemas operativos mais recentes, é possível que uma máquina física possa arrancar diretamente através de um sistema instalado num disco virtual, sendo possível usufruir de todas as vantagens de um ambiente virtualizado. Outras vantagens são:

- Implementação rápida de novas máquinas;
- Coerência na implementação de configurações e sistema base;
- Criar um sistema com a flexibilidade de um ambiente virtual utilizando o *hardware* local;
- Possibilidade da utilização de discos diferenciais, permitindo com isto eliminar todas as alterações efetuadas, voltando a um ponto inicial.

No fundo o que se pretende com a virtualização do disco rígido é que de forma rápida se possa voltar a um ponto anterior sem que seja necessário a reinstalação de um computador, poupando tempo, recursos e garantindo a continuidade do serviço em caso de falhas de *software*.

No entanto irá ser necessário garantir que seja possível efetuar alterações permanentes nos discos base, ou seja, a capacidade de manutenção deverá ser garantida e por essa razão o sistema de virtualização de discos deverá permitir que seja alterado o disco inicial e os dados nele contidos.

Os mecanismos de reversão e recuperação são essenciais para um sistema manter a sua propriedade de tolerância a falhas. O gestor de recuperação do sistema é capaz de o trazer de volta a um estado consistente onde todos os seus processos podem começar de novo. (Shah, 2002)

No fundo estes mecanismos podem ser utilizados para aumentar a confiança no funcionamento de um sistema, sendo que pelas diversas razões já referidas é possível voltar a um ponto atrás em caso de um comportamento anormal ou não esperado do sistema.

2.5. Recuperação

A recuperação de erros faz parte do processo de tolerância a falhas de *software*, sendo que o objetivo deste é extinguir os erros bem como os seus efeitos a partir de um estado computacional antes de ocorrer a falha.

O processo é decomposto na deteção de erros, diagnóstico, isolamento e recuperação, esta pode ser realizada utilizando uma recuperação para trás¹ ou para a frente².

2.5.1. Recuperação para trás

Quando um programa sofre um erro este torna-se instável podendo levar a uma falha, neste caso podem ser utilizadas técnicas de recuperação que permitam voltar para trás ou seja voltar a um ponto onde não existiam problemas através do restauro do sistema para um estado previamente guardado.

Estes estados são chamados de checkpoints e devem ser guardados em locais onde uma falha não os possa atingir. No fundo quando um erro é detetado podemos voltar para um estado anterior e continuar a processar a informação.

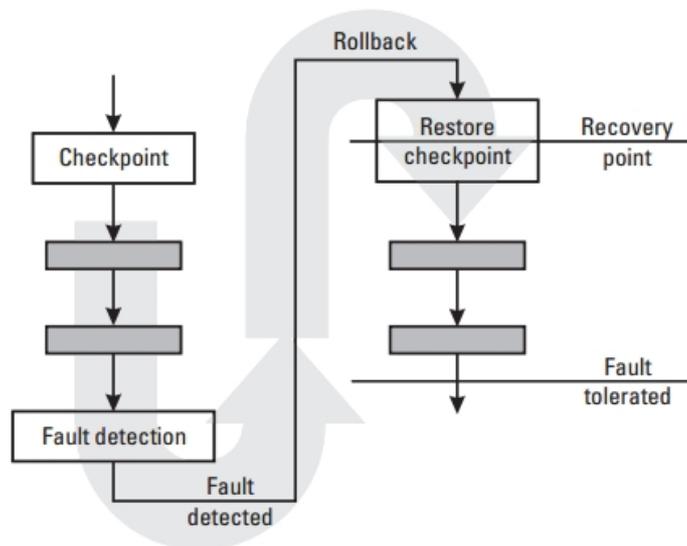


Figura 7 - Recuperação para trás (Pullum, 2001)

2.5.2. Recuperação para a frente

Ao contrário do ponto anterior, a recuperação para a frente tenta encontrar um novo estado de onde o sistema possa continuar após a deteção do erro. Podem ser utilizadas técnicas

¹ Backward Recovery

² Forward Recovery

ou algoritmos de compensação capazes de selecionar qual a melhor abordagem a seguir. Esta técnica é utilizada quando não é possível voltar a um ponto anterior.

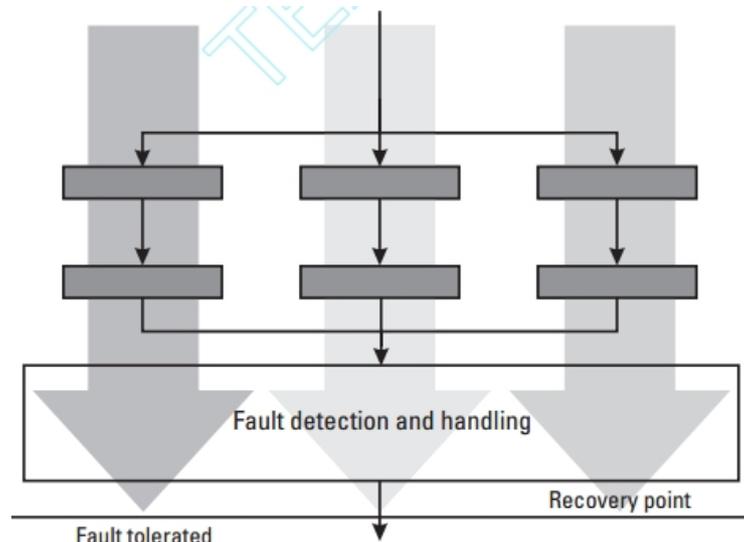


Figura 8 - Recuperação para frente utilizando processos redundantes (Pullum, 2001)

2.6. Engenharia de Software

Como já foi referido as metodologias para desenvolvimento de *software* de qualidade contribuem diretamente na confiança do funcionamento de um sistema, sendo que para isso um determinado software deve ser livre de falhas e vulnerabilidades.

Para atingir este objetivo devem ser utilizadas técnicas de validação ou seja metodologias de testes e avaliação de *software*.

2.6.1. SWEBOK

O SWEBOK³ é um guia criado pela IEEE⁴ e serve de referência em assuntos gerais mas pertinentes para a Engenharia de Software. Este apresenta uma classificação hierárquica de tópicos tratados pela Engenharia de Software, onde o nível mais alto são as Áreas do Conhecimento. Os cinco principais objetivos desta guia são:

- Promover uma visão consistente da engenharia de *software* no mundo;
- Esclarecer e delimitar as fronteiras entre a engenharia de *software* e as outras disciplinas relacionadas;
- Caracterizar o conteúdo da disciplina de engenharia de *software*;
- Classificar em tópicos a área de conhecimento da engenharia de *software*;

³ Software Engineering Body of Knowledge

⁴ Institute of Electrical and Electronics Engineers

- Promover uma fundação para o desenvolvimento do currículo, para certificação individual e para licenciamento de material.

Até 2014 as áreas de conhecimento eram 10, no entanto no guia mais recente estão presentes as 15 seguintes:

- Requisitos de Software;
- Desenho do Software;
- Construção do Software;
- Testes do Software;
- Manutenção do Software;
- Gestão da Configuração do Software;
- Gestão da Engenharia de Software;
- Processo da Engenharia de Software;
- Modelos e Métodos da Engenharia de Software;
- Qualidade do Software;
- Prática Profissional da Engenharia de Software;
- Economia da Engenharia de Software;
- Fundamentos de Computação;
- Fundamentos de Matemática;
- Fundamentos de Engenharia.

A área do conhecimento de requisitos de *software* é responsável pelo levantamento, análise, especificação e validação dos requisitos de software. Nesta fase devem ser envolvidos os *stakeholders*.

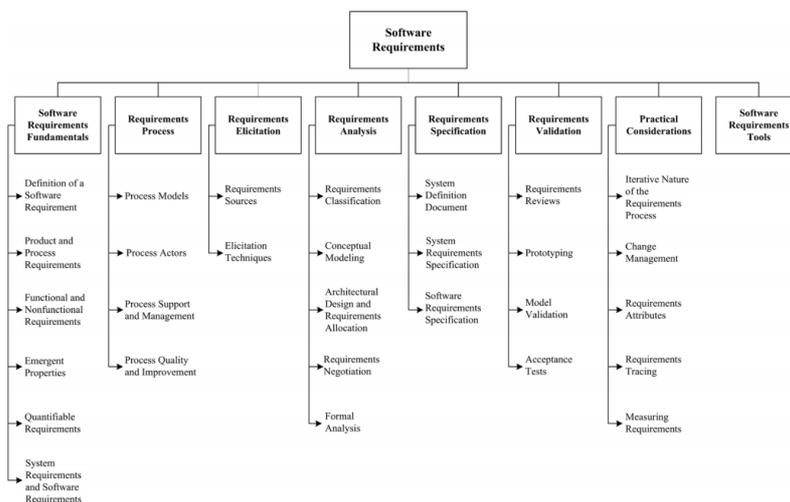


Figura 9 - Tópicos da Área de Conhecimento - Requisitos de Software (Bourque & Fairley, 2014)

Já na área de conhecimento desenho de *software* é definida a arquitetura, componentes, interfaces e outras características do sistema ou componente, bem como o resultado que

deve ser obtido ou esperado destes. Nesta fase normalmente são utilizadas ferramentas de desenvolvimento de *software* que têm por objetivo facilitar o trabalho do programador.

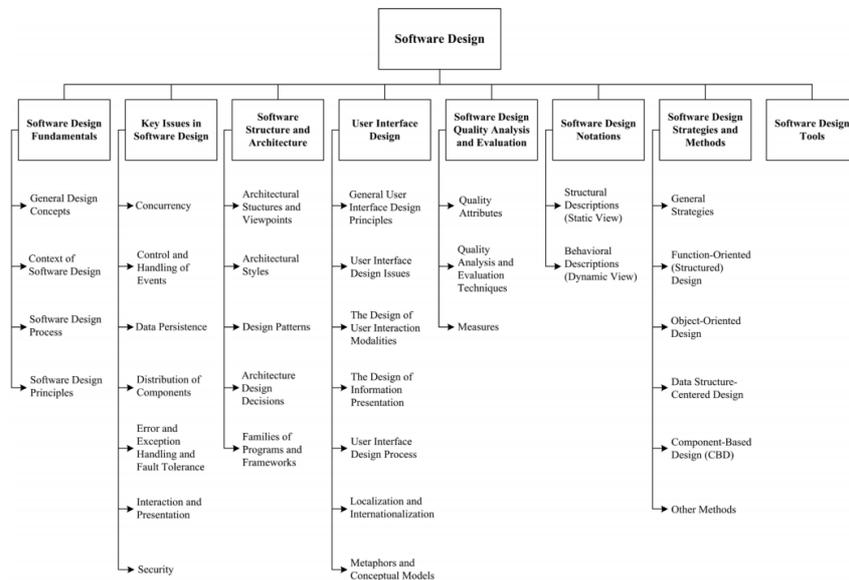


Figura 10 - Tópicos da Área de Conhecimento - Desenho de Software (Bourque & Fairley, 2014)

A área do conhecimento de construção de *software* esta associada à criação do software funcional através da combinação da codificação, verificação, testes, etc.

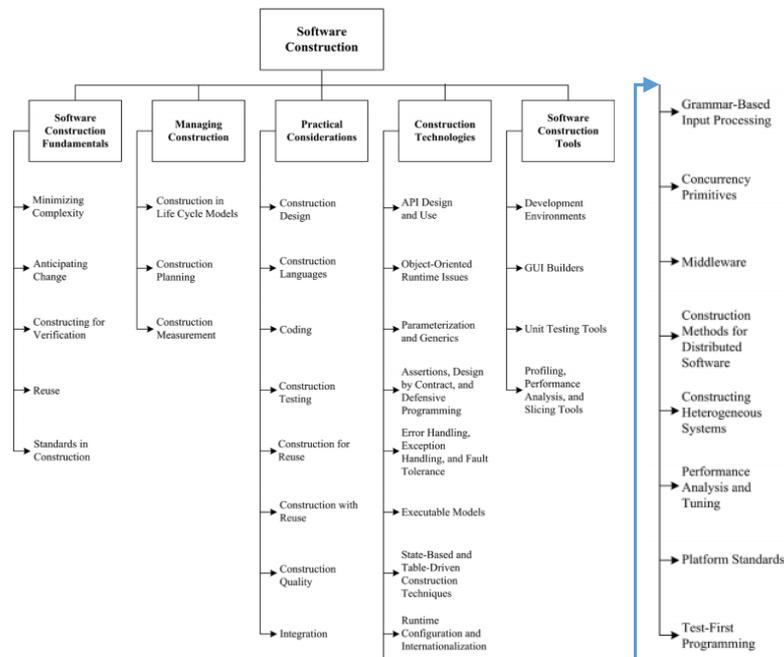


Figura 11 - Tópicos da Área de Conhecimento - Construção de Software (Bourque & Fairley, 2014)

A área de conhecimento de testes do *software* consiste na verificação dinâmica do comportamento de um programa num conjunto finito de casos de teste contra o comportamento esperado.

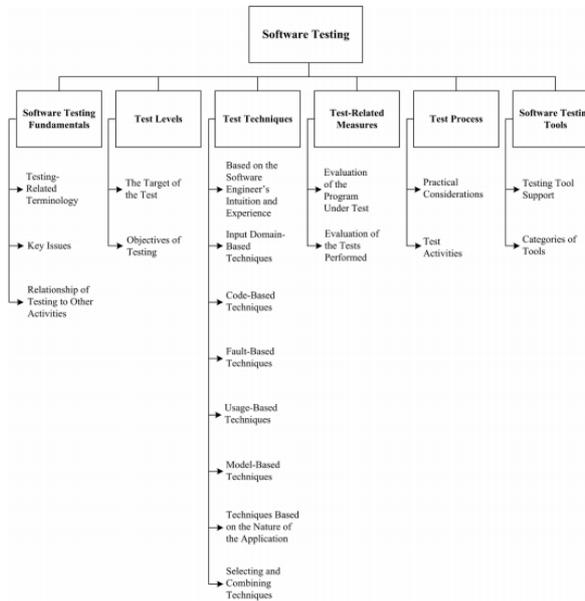


Figura 12 - Tópicos da Área de Conhecimento – Testes do Software (Bourque & Fairley, 2014)

No guia SWEBOK a manutenção de *software* é definida como a totalidade das atividades necessárias para fornecer suporte de baixo custo para o software. (Bourque & Fairley, 2014)

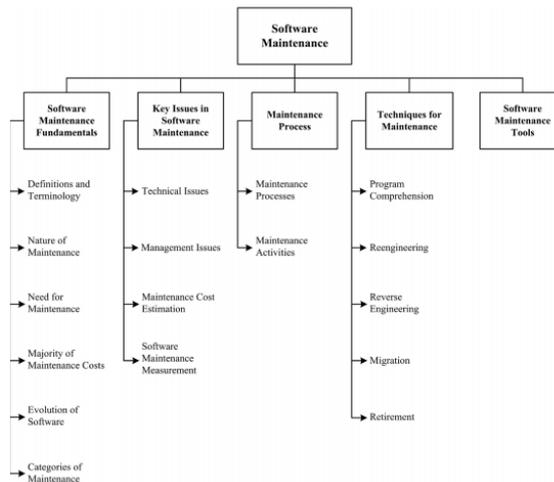


Figura 13 - Tópicos da Área de Conhecimento – Manutenção de Software (Bourque & Fairley, 2014)

As áreas de conhecimento referidas anteriormente com mais detalhe são das 15 as que mais relevância têm para esta investigação. Sendo que as restantes serão sempre utilizadas como acessório em caso de necessidade e por esse motivo não são aprofundadas neste trabalho.

3. Análise de Requisitos

Os requisitos apresentados nos pontos seguintes foram recolhidos através da experiência adquirida ao longo dos anos de trabalho no departamento e também das funcionalidades disponibilizadas pela solução utilizada anteriormente.

3.1. Requisitos Funcionais

Os requisitos funcionais da aplicação devem descrever explicitamente as funcionalidades e serviços do sistema, documentando a forma como este reage aos *inputs*, bem como o que deve fazer ou não deve fazer. No fundo é aquilo que se espera que o sistema produza. Por esse motivo os requisitos para esta aplicação são os seguintes:

- **Administração** - deverá ser possível de uma forma central a gestão do parque informático, bem como a sua monitorização ou qualquer outra tarefa comum, por forma a facilitar o trabalho diário aos Administradores de Sistemas. As tarefas passam por exemplo a instalação de novos sistemas, aplicações ou outras que possam ser incluídas futuramente.
- **Clientes** - deve ser possível a instalação de novos clientes remotamente a partir de um estado “Bare Metal”. Também deve ser possível criar uma cópia de um sistema padrão e a sua replicação por forma a tornar o processo mais rápido e eficiente. Devem ser garantidos todos os meios para que seja garantida a confiança no funcionamento.
- **Atualizações** – deve ser possível agendar uma data e hora para que sejam instaladas atualizações no sistema.
- **Diagnóstico** – se ocorrer algum problema deve ser possível efetuar um diagnóstico mesmo em caso de falha do sistema.
- **Segurança** – para além de todas as configurações de segurança que o sistema operativo disponibiliza, os clientes ao reiniciarem o sistema todas as alterações deverão ser descartadas.

3.2. Requisitos Não Funcionais

No que concerne aos requisitos não-funcionais, estes não implicam diretamente nas funcionalidades do sistema ou do que se espera do resultado deste. Os RNF são:

- **Sistema Operativo** – a aplicação deverá ser compatível com o Windows 7 e as ferramentas disponibilizadas pela Microsoft, mais especificamente Windows Automated Installation Kit.

- **Interoperabilidade** – a consola de gestão central deverá poder ser executada nos vários Sistemas Operativos utilizados pelos Administradores de Sistemas, mais nomeadamente Windows 7 e 8.1.
- **Escalabilidade** – a aplicação deverá ser desenhada para que seja possível a qualquer altura crescer sem afetar a performance de forma negativa a consola de gestão ou os próprios clientes.
- **Segurança** – deverão ser implementadas as melhores práticas no que concerne à segurança e desenvolvimento de *software*, como por exemplo as credenciais deverão estar encriptadas.

4. Modelação

É na fase de modelação que são propostos os modelos que suportam o desenvolvimento da aplicação, criados através do trabalho em equipa com todos os stakeholders.

Estes modelos são capazes de representar o sistema, a sua estrutura e o seu comportamento.

4.1. Casos de Uso

A aplicação deverá conter todas as funcionalidades presentes no levantamento e análise de requisitos. Os casos de uso apresentados representam as várias componentes do sistema e a sua utilização por parte do administrador de sistemas.

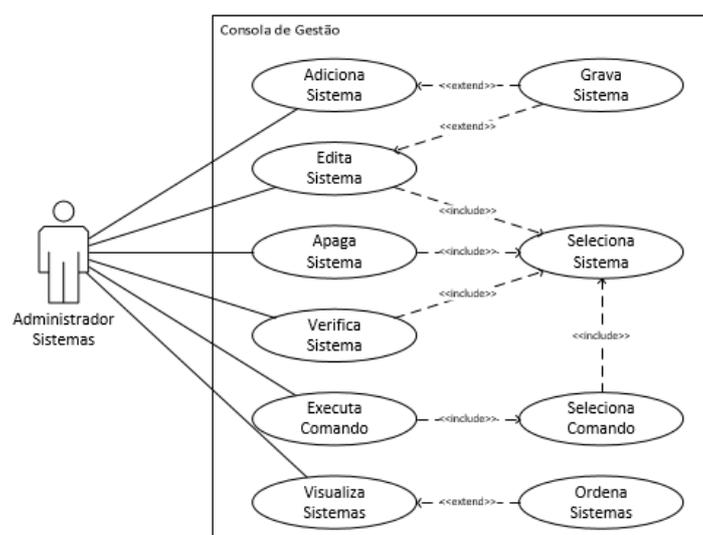


Figura 14 - Caso de Uso Consola de Gestão

O Administrador de Sistemas pode adicionar, editar, apagar e gravar novos sistemas, sendo que neste caso o sistema é a estação de trabalho do utilizador. Para efetuar as várias tarefas num sistema, este terá que ser selecionado em primeiro lugar, sendo que podem ser selecionados vários ao mesmo tempo. No que concerne à visualização dos sistemas, esta pode ser ajustada e ordenada ou não por ordem alfabética por forma a facilitar a interação homem-máquina.

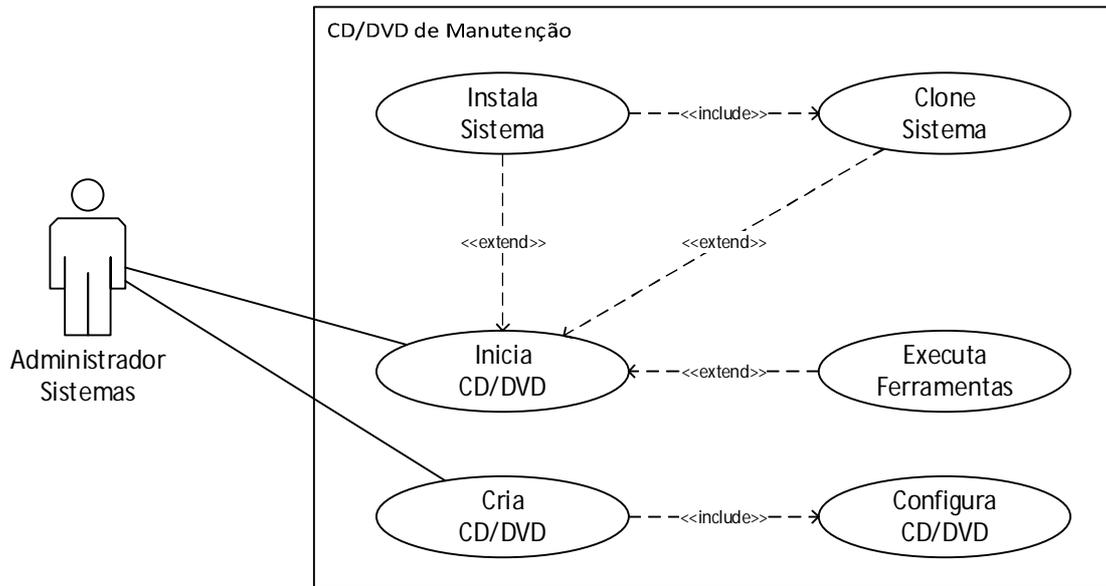


Figura 15 - Caso de Uso CD/DVD de Manutenção

Para instalar um novo sistema o administrador recorre ao CD/DVD de arranque, este permite instalar novos sistema ou clonar existentes. Também podem ou não ser executadas as ferramentas disponíveis e referidas no levantamento de requisitos. O CD/DVD deve ser previamente criado e configurado.

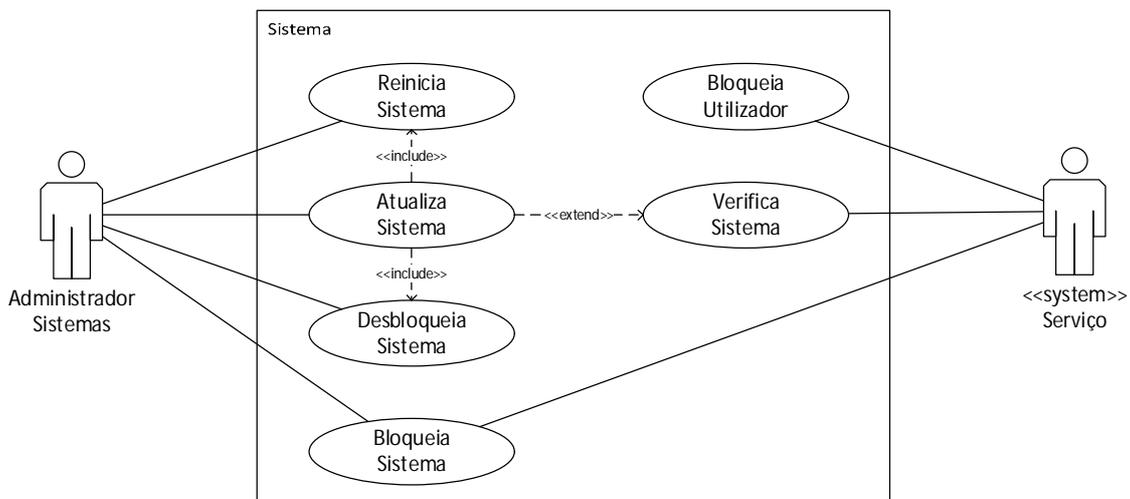


Figura 16 - Caso de Uso Administração do Sistema

No computador ou estação de trabalho o serviço desenvolvido deverá ser capaz de verificar, atualizar, desbloquear e bloquear e reiniciar o sistema. O administrador de sistema também poderá executar estas tarefas manualmente.

5. Arquitetura

No que concerne à arquitetura e tendo em conta os requisitos e natureza destes, a arquitetura proposta é a seguinte:

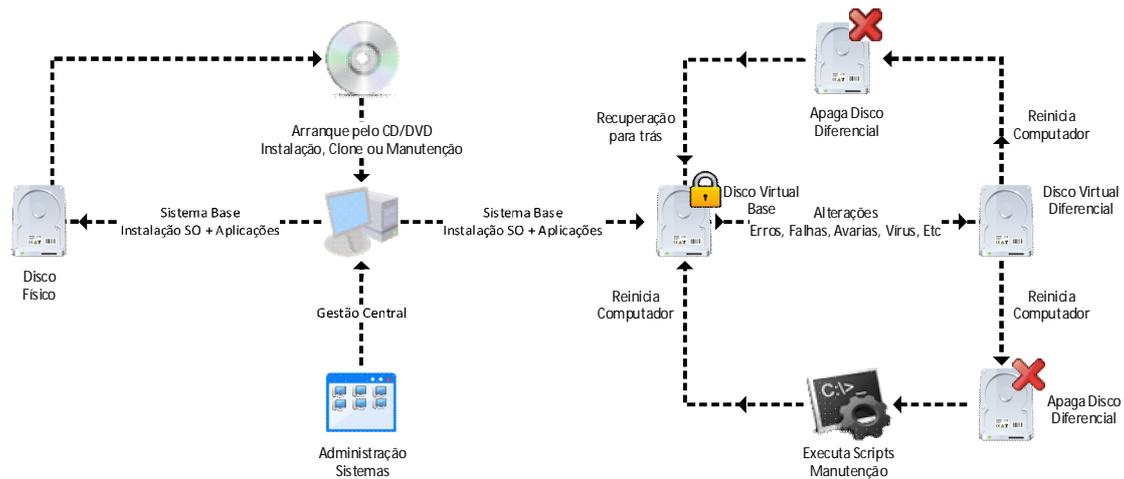


Figura 17 - Arquitetura do Sistema

Como podemos observar na figura anterior, para instalar um novo computador é necessário arrancar pelo CD/DVD de instalação do Sistema Operativo ou então utilizar o CD/DVD de manutenção. No último caso existe a possibilidade de criar ou restaurar um clone de um computador já instalado.

Após a instalação do Sistema Operativo e as aplicações, será possível proteger o computador para que não sejam aplicadas alterações permanentes, sejam estas voluntárias ou não, sendo que ao reiniciar o SO todas as diferenças serão descartadas.

Para instalar atualizações ou efetuar alterações existirá um serviço responsável por executar todas as tarefas necessárias a uma determinada hora do dia, em seguida o disco é protegido novamente contra alterações.

Com a consola de administração central o administrador de sistemas será capaz de efetuar as tarefas remotamente, monitorizar o estado dos sistemas e alterar as configurações destes se necessário.

6. Desenvolvimento

6.1. Introdução

No que concerne ao desenvolvimento de *software*, foi adotada a metodologia que mais se adequa ao tipo de projeto e as necessidades deste, sendo que o objetivo é garantir a estrutura e qualidade da aplicação utilizando a metodologia como um guia.

Para materializar a aplicação foram utilizadas diversas tecnologias e ferramentas, bem como algoritmos criadas especificamente para o efeito e apresentados nos pontos seguintes.

6.2. Metodologia

Como a arquitetura da aplicação é composta por vários componentes que trabalham em conjunto, optei pela divisão destes em módulos e desenvolvi cada um deles individualmente.

Apesar das metodologias ágeis serem as mais aconselhadas recentemente devido à sua flexibilidade e adaptabilidade, a escolhida para esta aplicação foi a metodologia de desenvolvimento tradicional em cascata, recorrendo a estratégias de desenvolvimento iterativo e incremental, facilitando com isto o desenvolvimento de tarefas em paralelo que serão integradas quando concluídas, deixando de lado a rigidez imposta pelo desenvolvimento em cascata tradicional.

O modelo em cascata é composto por várias fases, o levantamento de requisitos, desenho da aplicação, implementação, testes e manutenção.

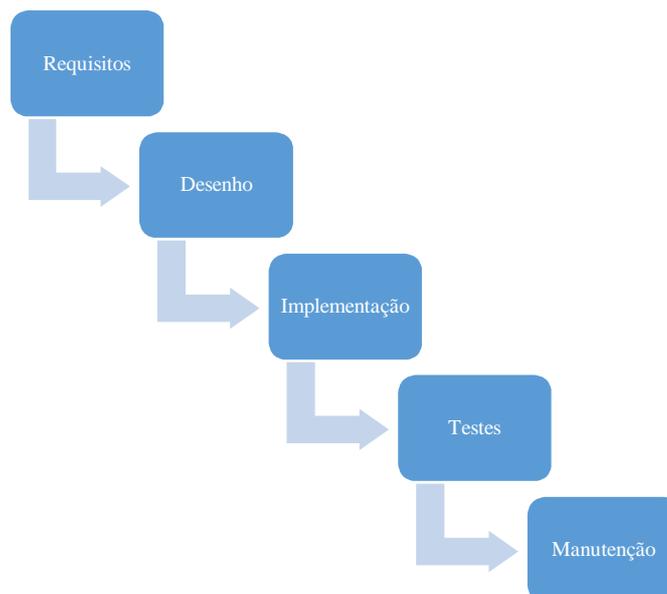


Figura 18 - Modelo em Cascata

Já o modelo incremental é muito semelhante mas com várias iterações ou incrementos, como pode ser observado na figura abaixo.

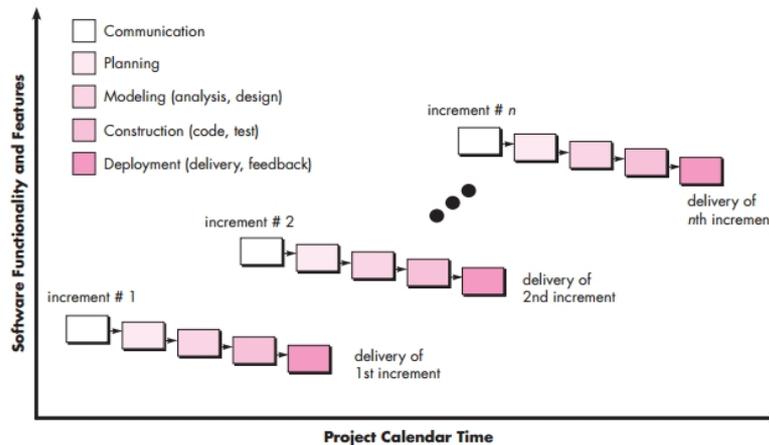


Figura 19 - Modelo Incremental (Pressman, 2010)

Na primeira fase foi efetuada análise de requisitos funcionais e não funcionais descritos anteriormente. Nesta fase é essencial que sejam reunidos os stakeholders com o intuito de perceber quais as necessidades da nova aplicação, bem como todas as funcionalidades que esta deve conter. Estas reuniões são caracterizadas por uma interação elevada e como resultado deve ser apresentada uma solução para os problemas.

Na fase de desenho foram criados e apresentados os *mookups* com o *layout* esperado da aplicação, sendo efetuados alguns ajustes necessários à medida que iam sendo sugeridos.

A fase de implementação foi a mais morosa e caracterizou-se pela codificação da solução e desenvolvimento de todas as funcionalidades pedidas na análise de requisitos.

Os testes foram inicialmente efetuados em conjunto com a equipa de IT num ambiente controlado e sem colocar em causa à operação, após essa fase foram executados dois pilotos em lojas distintas a fim de perceber o comportamento da aplicação em produção. Depois de validados os requisitos e concluídos os testes em produção, foi implementada a aplicação nas restantes lojas.

A fase final ou manutenção foi acompanhado e corrigidas pequenas falhas na aplicação e explicado o seu funcionamento aos Administradores de Sistemas.

6.3. Tecnologias

As tecnologias adotadas foram de encontro aos requisitos não funcionais, sendo que o fabricante de *software* principal da organização é a Microsoft. Por esse motivo as principais ferramentas e linguagens de programação adotadas foram as descritas nos pontos seguintes.

6.3.1. Visual Studio

Para o desenvolvimento deste projeto o IDE utilizado foi o Microsoft Visual Studio 2013, por ser uma ferramenta versátil e muito utilizada no mercado de trabalho e também a nível académico. Este *software* está disponível em várias versões, que podem ser descarregadas do site da Microsoft.

A versão utilizada é paga e está disponível para *download* no DreamSpark através do acordo que a UATLA tem com o fabricante, no entanto o código fonte pode ser visualizado, editado e compilado na versão gratuita ou *express*.

6.3.2. C#

O C# “C sharp” foi a linguagem de programação eleita para o desenvolvimento da consola de administração, sendo que as aplicações nela criadas são executadas sobre o .NET Framework e compatível com os sistemas operativos mais recentes da Microsoft.

O C# foi eleito por ser uma linguagem simples, poderosa, segura e orientada a objetos. Esta linguagem em conjunto com o IDE permite o desenvolvimento rápido de aplicações, mantendo o estilo e “elegância” da linguagem C.

O Visual C# é uma implementação da linguagem C# pela Microsoft. Como já foi referido o Visual Studio suporta o Visual C# com um editor de código completo, compilador, modelos de projetos, *designers*, assistentes de código, um depurador poderoso e fácil de usar e outras ferramentas.

A biblioteca de classes do .NET Framework fornece acesso a vários serviços do sistema operacional e outras classes úteis e bem estruturadas que aceleram significativamente o ciclo de desenvolvimento.

6.3.3. VB .NET

À semelhança do C# o Visual Basic .net é uma linguagem de programação orientada a objetos com suporte a UML, também criada pela Microsoft, não é tão popular a nível das empresas mas muito fácil de programar principalmente para iniciantes. Neste caso foi escolhida para desenvolver o serviço que é instalado no computador e a escolha recai sobre a reutilização de código efetuado previamente, sendo que este estava testado e aprovado.

6.3.4. Shell script

O Shell script é uma linguagem de programação utilizada em vários sistemas operativos e que são executados através de um interpretador de comandos.

Podem ser criados ficheiros *batch* para simplificar e automatizar tarefas repetitivas, sendo que estes ficheiros podem ser criados com recurso a um bloco de notas e apenas contem comandos de forma encadeada. Também é possível incluir comandos que permitam um processamento condicional, tais como *if*, *goto*, *for*. Quando é executado um comando é devolvido um código de erro ou resultado, este assume o valor 0 na maioria das tarefas e quando estas são executadas com sucesso.

6.3.5. Windows Automated Installation Kit

O Windows AIK é um conjunto de ferramentas e de documentação que suporta a configuração e a implementação de sistemas operativos da Microsoft. A utilização desta ferramenta permite automatizar as instalações do Windows, capturar imagens do Windows com ImageX, configurar e modificar imagens utilizando o Gestão e Atualização de Imagens de Implementação (DISM), criar imagens do Windows PE, etc.

6.3.6. Microsoft Diagnostics and Recovery Toolset

O Microsoft DaRT é uma das ferramentas que pertence ao Microsoft Desktop Optimization Pack e está disponível para clientes do Software Assurance. Nesta *suite* estão incluídos aplicativos que ajudam a resolver problemas, efetuar diagnósticos e outro tipo de ferramentas uteis para os administradores de sistemas. Para além disso é possível criar ou executar aplicações simples e que tenham compatibilidade com o sistema WinPE⁵.

6.4. Interface Homem-Máquina

Na consola de administração é possível adicionar e remover novos sistemas bem como efetuar a monitorização e as tarefas descritas no levantamento de requisitos.

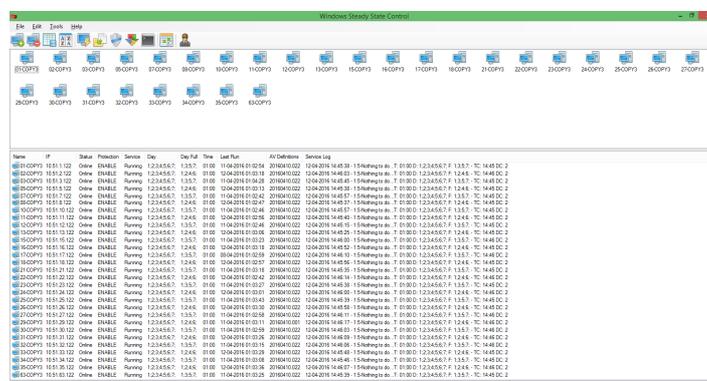


Figura 20 - Consola de Administração

⁵ Windows Preinstallation Environment



Figura 21 - Menu Principal

Depois de seleccionados os sistemas pretendidos pode ser escolhida a tarefa a executar disponível no menu principal ou então no menu de contexto.

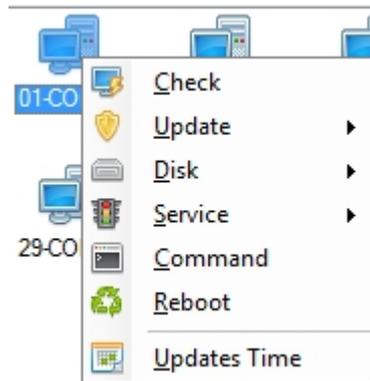


Figura 22 - Menu de Contexto

O menu de contexto permite ao administrador de sistemas verificar o estado do cliente, efetuar atualizações, proteger e desproteger o disco, manutenção do serviço, executar comandos, reiniciar o computador e alterar a hora e dia das atualizações automáticas.

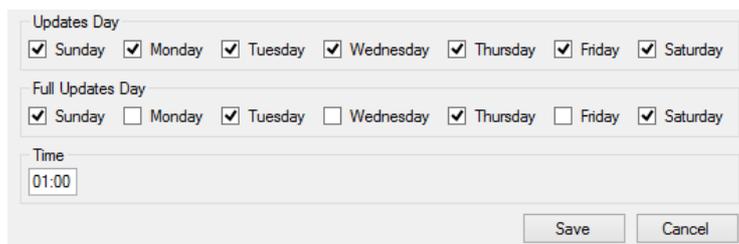


Figura 23 - Configuração atualizações

O resultado das operações e monitorização dos clientes são apresentados na grelha da figura seguinte.

Name	IP	Status	Protection	Service	Day	Day Full	Time	Last Run	AV Definitions	Service Log
01-COPY3	10.51.1.122	Online	ENABLE	Running	1,2,3,4,5,6,7	1,3,5,7	01:00	11-04-2016 01:02:54	20160410.022	12-04-2016 14:45:38 - 1.5-Nothing to do...T: 01:00 D: 1,2,3,4,5,6,7; F: 1,3,5,7; - TC: 14:45 DC: 2
02-COPY3	10.51.2.122	Online	ENABLE	Running	1,2,3,4,5,6,7	1,2,4,6	01:00	12-04-2016 01:03:18	20160410.022	12-04-2016 14:46:03 - 1.5-Nothing to do...T: 01:00 D: 1,2,3,4,5,6,7; F: 1,2,4,6; - TC: 14:46 DC: 2
03-COPY3	10.51.3.122	Online	ENABLE	Running	1,2,3,4,5,6,7	1,3,5,7	01:00	11-04-2016 01:04:28	20160410.022	12-04-2016 14:45:45 - 1.5-Nothing to do...T: 01:00 D: 1,2,3,4,5,6,7; F: 1,3,5,7; - TC: 14:45 DC: 2
05-COPY3	10.51.5.122	Online	ENABLE	Running	1,2,3,4,5,6,7	1,2,4,6	01:00	12-04-2016 01:03:13	20160410.022	12-04-2016 14:45:38 - 1.5-Nothing to do...T: 01:00 D: 1,2,3,4,5,6,7; F: 1,2,4,6; - TC: 14:45 DC: 2
07-COPY3	10.51.7.122	Online	ENABLE	Running	1,2,3,4,5,6,7	1,3,5,7	01:00	11-04-2016 01:02:42	20160410.022	12-04-2016 14:45:57 - 1.5-Nothing to do...T: 01:00 D: 1,2,3,4,5,6,7; F: 1,3,5,7; - TC: 14:45 DC: 2

Figura 24 - Tabela resultados

Quando é iniciado o CD/DVD de arranque é possível ao administrador de sistemas efetuar a instalação e manutenção remotamente, sendo que o interface apresentado é o da figura abaixo.

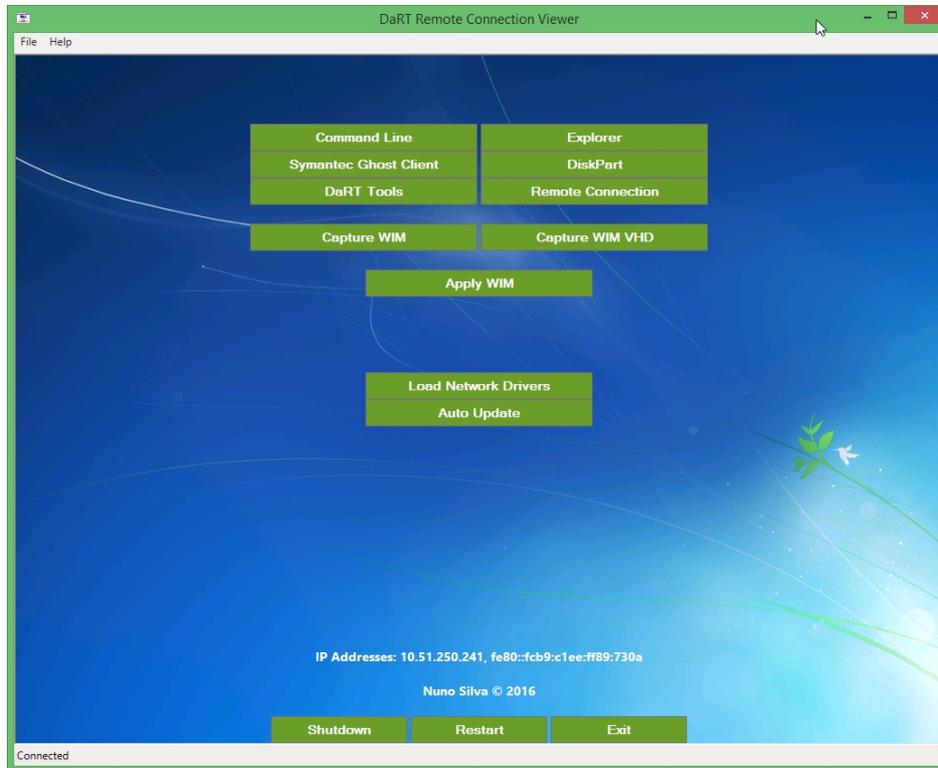


Figura 25 - Interface CD/DVD arranque

Nos computadores onde a aplicação está instalada é apresentada a informação do estado da proteção do disco, por forma a ser fácil para o utilizador final perceber se é seguro trabalhar ou não na máquina.

Host Name:	PTSTORE-XXCOPYZ	Host Name:	PTSTORE-XXCOPYZ
IP:	10.51.250.241	IP:	10.51.250.241
Boot Time:	12-04-2016 16:36	Boot Time:	12-04-2016 17:09
Disk Protection:	DISABLE	Disk Protection:	ENABLE

Figura 26 - Estado proteção disco

6.5. Código fonte

Os algoritmos e código fonte desenvolvidos estão presentes em anexo desta dissertação em suporte digital, no entanto os mais relevantes são apresentados abaixo.

Para executar comandos nas máquinas remotas foi criado um método da figura seguinte, onde têm que ser passados os argumentos IP e comando a executar.

```
private string runRemoteCommand(string IP, string command)
{
    try
    {
        ConnectionOptions ConnectionOptions = new ConnectionOptions();

        ConnectionOptions.Username = @"\" + USERNAME;
        ConnectionOptions.Password = PASSWORD;

        ManagementScope manScope = new ManagementScope(String.Format(@"\\{0}\ROOT\CIMV2", IP), ConnectionOptions);
        manScope.Connect();

        ObjectGetOptions objectGetOptions = new ObjectGetOptions();
        ManagementPath managementPath = new ManagementPath("Win32_Process");

        ManagementClass processClass = new ManagementClass(manScope, managementPath, objectGetOptions);

        ManagementBaseObject inParams = processClass.GetMethodParameters("Create");

        inParams["CommandLine"] = "CMD.EXE /C " + command;
        inParams["CurrentDirectory"] = @"C:\WINDOWS\SYSTEM32";
    }
}
```

Figura 27 - Método runRemoteCommand

O método “serviceControl” é responsável por parar, iniciar, reiniciar e devolver o estado atual do serviço, sendo que devem ser passados os parâmetros IP e a ação (Stop, Start, Restart e Status).

```
private string serviceControl(string IP, string action)
{
    try
    {
        string result = null;
        string serviceName = "Windows SteadyState Service";

        string networkPath = @"\" + IP + @"\D$";
        NetworkCredential NetworkCredential = new NetworkCredential(USERNAME, PASSWORD);

        using (new NetworkConnection(networkPath, NetworkCredential))
        {
            ServiceController ServiceController = new ServiceController();
            ServiceController.ServiceName = serviceName;
            ServiceController.MachineName = IP;
            result = ServiceController.Status.ToString();

            if (action == "Start")
            {
                if (ServiceController.Status == ServiceControllerStatus.Stopped)
                {
                    ServiceController.Start();
                    ServiceController.WaitForStatus(ServiceControllerStatus.Running);
                    result = ServiceController.Status.ToString();
                }
            }
            else if (action == "Stop")
            {
                if (ServiceController.Status == ServiceControllerStatus.Running)
                {
                    ServiceController.Stop();
                    ServiceController.WaitForStatus(ServiceControllerStatus.Stopped);
                    result = ServiceController.Status.ToString();
                }
            }
        }
    }
}
```

Figura 28 - Método serviceControl

Para ler a chave de registo onde consta o valor que indica a versão da base de dados do antivírus foi adaptado o método “readRemoteRegistry” da figura seguinte.

```
private string readRemoteRegistryAV(string IP)
{
    try
    {
        string result = null;

        ConnectionOptions ConnectionOptions = new ConnectionOptions();

        ConnectionOptions.Username = @"\" + USERNAME;
        ConnectionOptions.Password = PASSWORD;

        string softwareRegLoc = @"SOFTWARE\Symantec\Symantec Endpoint Protection\CurrentVersion\SharedDefs";

        ManagementScope manScope = new ManagementScope(String.Format(@"\\{0}\ROOT\CIMV2", IP), ConnectionOptions);
        manScope.Connect();

        ObjectGetOptions objectGetOptions = new ObjectGetOptions();
        ManagementPath managementPath = new ManagementPath("StdRegProv");
    }
}
```

Figura 29 - Método readRemoteRegistryAV

O serviço instalado no computador terminal está em constante funcionamento e verifica todos os requisitos a cada um minuto. Se alguma das condições explícitas no código ocorrerem é então iniciada a tarefa correspondente.

```

Do While True
  'Every Minute Check Current PC Hour
  Dim UPDATES_RUNNING As String
  UPDATES_RUNNING = My.Computer.FileSystem.ReadAllText(DISK_SYS & "FLAGS\UPDATES_RUNNING")

  If (UPDATES_RUNNING = 1) Then
    WriteLog("Disabling normal user accounts...")

    System.Diagnostics.Process.Start(DISK_SYS & "SCRIPTS\DISABLE_ACCOUNTS.CMD")

    Dim objWriter As New System.IO.StreamWriter(DISK_SYS & "FLAGS\UPDATES_RUNNING")

    objWriter.Write("2")
    objWriter.Close()

    WriteLog("Restarting again...")
    System.Diagnostics.Process.Start(DISK_SYS & "SCRIPTS\RESTART_FOR_UPDATES_NOW.CMD")
    Exit Do

  ElseIf (UPDATES_RUNNING = 2) Then
    Dim UPDATES_TIME As Date
    UPDATES_TIME = My.Computer.FileSystem.ReadAllText(DISK_SYS & "FLAGS\UPDATES_TIME")
    Dim UPDATES_WAIT As String
    UPDATES_WAIT = My.Computer.FileSystem.ReadAllText(DISK_SYS & "FLAGS\UPDATES_WAIT")

```

Figura 30 - Excerto código fonte serviço

Para facilitar a criação do CD/DVD de arranque foram criados scripts que executam as tarefas repetitivas de forma automática através de comandos.

```

WinPE_Creator.CMD x
173 ECHO.
174
175 REM CHANGE STARTNET.CMD
176 SET /P "=CHANGE STARTNET.CMD....." < NUL
177 ECHO @ECHO OFF > "%MOUNT%\WINDOWS\SYSTEM32\STARTNET.CMD"
178 ECHO WPEINIT >> "%MOUNT%\WINDOWS\SYSTEM32\STARTNET.CMD"
179 ECHO CLS >> "%MOUNT%\WINDOWS\SYSTEM32\STARTNET.CMD"
180 ECHO @ECHO WINPE CREATED BY NUNO SILVA 2015 >> "%MOUNT%\WINDOWS\SYSTEM32\STARTNET.CMD"
181 ECHO %SYSTEMDRIVE%\SOURCES\TOOLS\START.HTA >> "%MOUNT%\WINDOWS\SYSTEM32\STARTNET.CMD"
182 ECHO CLS >> "%MOUNT%\WINDOWS\SYSTEM32\STARTNET.CMD"
183 IF %ERRORLEVEL%==1 ECHO OK!
184 IF NOT %ERRORLEVEL%==1 ECHO ER! && ECHO ERROR: %ERRORLEVEL% && PAUSE && GOTO EXIT
185 ECHO.
186
187 REM ADD FIX OF FOLDERBROWSERDIALOG
188 SET /P "=ADD FIX OF FOLDERBROWSERDIALOG....." < NUL
189 REG LOAD HKLM\WINPETEMPHIVE "%MOUNT%\WINDOWS\SYSTEM32\CONFIG\SOFTWARE" > NUL
190 REGEDIT /S "%REGISTRY%\FOLDERBROWSERDIALOG.REG" > NUL
191 REG UNLOAD HKLM\WINPETEMPHIVE > NUL
192 XCOPY "%DLLFO%\EXPLORERFRAME.DLL" "%MOUNT%\WINDOWS\SYSTEM32\" /Y > NUL
193 IF %ERRORLEVEL%==0 ECHO OK!
194 IF NOT %ERRORLEVEL%==0 ECHO ER! && ECHO ERROR: %ERRORLEVEL% && PAUSE && GOTO EXIT
195 ECHO.

```

Figura 31 - Excerto código fonte script CD/DVD

O script “DISKPART_IMAGE_APPLY” é responsável por instalar e configurar uma máquina bare metal.

```

DISKPART_IMAGE_APPLY.CMD
193 IF %ERRORLEVEL%==0 ECHO OK!
194 IF NOT %ERRORLEVEL%==0 ECHO ER! && ECHO ERROR: %ERRORLEVEL% && PAUSE && GOTO EXIT
195 ECHO.
196
197 REM SET DEFAULT BOOT TO VHD_SYST AND DISK PROTECTION DISABLE
198 SET /P "=SET DEFAULT BOOT TO VHD_SYST AND DISK PROTECTION DISABLE....." < NUL
199 BCDEDIT -DEFAULT {%VHD_BASE_GUID%} > NUL
200 ECHO.|SET /P ="DISABLE" > %DISK_SYS%\FLAGS\DISK_PROTECTION REM /P DONE TO AVOID NEW LINE
201 IF %ERRORLEVEL%==1 ECHO OK!
202 IF NOT %ERRORLEVEL%==1 ECHO ER! && ECHO ERROR: %ERRORLEVEL% && PAUSE && GOTO EXIT
203 ECHO.
204
205 REM COPY SOME STUFF TO DISK_SYS
206 SET /P "=SET SOME DEFAULT FLAGS....." < NUL
207 ECHO.|SET /P ="0" > %DISK_SYS%\FLAGS\UPDATES_RUNNING REM /P DONE TO AVOID NEW LINE
208 ECHO.|SET /P ="01:00" > %DISK_SYS%\FLAGS\UPDATES_TIME
209 ECHO.|SET /P ="1;2;3;4;5;6;7;" > %DISK_SYS%\FLAGS\UPDATES_DAY REM Sunday = 1...Saturday=7
210 ECHO.|SET /P ="1;2;3;4;5;6;7;" > %DISK_SYS%\FLAGS\UPDATES_DAY_FULL REM Sunday = 1...Saturday=7
211 ECHO.|SET /P ="1800000" > %DISK_SYS%\FLAGS\UPDATES_WAIT REM 1000 = 1 Second - 1800000=30 Minutes
212 IF %ERRORLEVEL%==1 ECHO OK!
213 IF NOT %ERRORLEVEL%==1 ECHO ER! && ECHO ERROR: %ERRORLEVEL% && PAUSE && GOTO EXIT
214 ECHO.
...

```

Figura 32 - Excerto script DISKPART_IMAGE_APPLY.CMD

Cada vez que o computador é reiniciado, este arranca para uma versão do Windows PE configurado para executar o script “REBUIL_SCRIPT.CMD”, este é responsável por apagar e voltar a criar o disco diferencial.

```

REBUILD_SCRIPT.CMD
13 REM DELETE DIFFERENCING DISK
14 SET /P "=DELETE DIFFERENCING DISK....." < NUL
15 DEL %DISK_SYS%\VHD\VHD_DIFF.VHD /F /Q > NUL
16 IF %ERRORLEVEL%==0 ECHO OK!
17 IF NOT %ERRORLEVEL%==0 ECHO ER! && ECHO ERROR: %ERRORLEVEL% && PAUSE && GOTO EXIT
18 ECHO.
19
20 REM CREATE NEW DIFFERENCING DISK
21 SET /P "=CREATE NEW DIFFERENCING DISK....." < NUL
22 ECHO CREATE VDISK FILE=%DISK_SYS%\VHD\VHD_DIFF.VHD PARENT=%DISK_SYS%\VHD\VHD_SYST.VHD | DISKPART > NUL
23 IF %ERRORLEVEL%==0 ECHO OK!
24 IF NOT %ERRORLEVEL%==0 ECHO ER! && ECHO ERROR: %ERRORLEVEL% && PAUSE && GOTO EXIT
25 ECHO.
26
27 REM SET DEFAULT BOOT AND TIMEOUT
28 SET /P "=SET DEFAULT BOOT AND TIMEOUT....." < NUL
29 FOR /F "TOKENS=2 DELIMS=" %I IN ("TYPE %DISK_SYS%\FLAGS\WIN_BOOT_MANAGER_GUID | FIND "VHD_DIFF") DO SET GUID=%I
30 BCDEDIT /BOOTSEQUENCE {%GUID%} > NUL
31 REM FOR TROUBLESHOOTING PURPOSES
32 BCDEDIT /TIMEOUT 1 > NUL
33 IF %ERRORLEVEL%==0 ECHO OK!
34 IF NOT %ERRORLEVEL%==0 ECHO ER! && ECHO ERROR: %ERRORLEVEL% && PAUSE && GOTO EXIT
35 ECHO.

```

Figura 33 - Excerto script REBUIL_SCRIPT.CMD

7. Testes e Resultados

7.1. Introdução

Os testes e a apresentação de resultados faz com que o trabalho seja validado e que no fundo se perceba se foi possível ou não usufruir das soluções adotadas e implementadas.

Nos pontos seguintes vou referir como foi implementado o sistema e quais os resultados obtidos.

7.2. Implementação e testes

Como já foi referido a implementação e testes da aplicação foram efetuados em conjunto com a equipa de IT num ambiente controlado de laboratório, ao mesmo tempo foi explicado o funcionamento e dada a respetiva formação aos administradores de sistemas.

No decorrer dos testes foram otimizados os processos e melhorados alguns dos algoritmos por forma a garantir a estabilidade do sistema.

Após a validação por parte da equipa de IT foram criadas duas lojas piloto, onde foi possível testar o sistema num ambiente real e ao mesmo tempo dada formação e explicados alguns detalhes ao utilizador final, nomeadamente que o sistema deve permanecer ligado todos os dias para ser atualizado e também que quando surgir algum problema devem reiniciar o computador.

Com o sucesso dos pilotos foi avançada a instalação nas restantes lojas do grupo, perfazendo até a data de entrega desta dissertação um total de 28 computadores.

7.3. Resultados

No que concerne aos resultados foi possível observar uma redução das incidências para 0, sendo que foi analisado previamente um período de 3 meses e comparado um período igual após a instalação da aplicação.

INCIDÊNCIAS REPORTADAS		
LOJA	01-05-2015/31-07-2015	01-12-2015/29-02-2016
Alfragide	6	0
Matosinhos	3	0
Braga	5	0
Cascais	6	0
Olivais	6	0
Sintra	4	0

V.F. Xira	3	0
Aveiro	3	0
Faro	3	0
Guimarães	9	0
V. Castelo	4	0
Montijo	3	0
Penafiel	4	0
Santarém	3	0
Barcelos	4	0
Setúbal	3	0
S.M. Feira	3	0
Viseu	2	0
Coimbra	2	0
V. Conde	6	0
Maia	3	0
Ermesinde	2	0
Lixa	2	0
Almada	4	0
Valongo	2	0
Palmela	2	0
Tejo	2	0
T. Vedras	3	0

Tabela 1 - Incidências nos Computadores das Lojas

Como podemos observar na tabela anterior, foi possível aumentar a confiança no funcionamento do sistema, ao mesmo tempo que foi garantida a compatibilidade e as melhorias introduzidas pelo fabricante nesta nova versão do sistema operativo, sem nunca afetar as operações diárias das lojas.

Também foi possível melhorar significativamente a produtividade dos stakeholders, uma vez que o sistema se tornou mais rápido e tolerante a falhas.

Uma vez que as incidências nos sistemas reduziram, os administradores de sistemas também foram beneficiados, não só com a redução para zero destas, mas também com a manutenção realizada numa base diária, pois passou a ser possível executar tarefas em todos os computadores de forma centralizada.

8. Conclusão

Nesta dissertação foi explicado o trabalho efetuado com o intuito de criar uma aplicação para resolver um problema organizacional nas áreas das tecnologias de informação e engenharia de software numa empresa multinacional.

A metodologia de investigação adotada ajudou a que todo este processo fosse seguido de forma lógica, ordenada e sistemática, resultando num artefacto que resolveu um problema real ou mais especificamente uma necessidade da empresa.

Com a revisão e análise da literatura compreendeu-se melhor o fenómeno, uma vez que nesta fase já eram conhecidos os problemas e os objetivos a concretizar. Foi também possível perceber caminhos distintos ou formas de resolver o problema.

Na fase de análise de requisitos procurou-se conhecer o contexto tecnológico, através do estabelecimento de alguma interatividade com os vários *stakeholders* associados ao problema organizacional previamente descrito.

Foi através da modelação dos casos de uso que se apresentou de uma forma simples o que se iria materializar na aplicação desenvolvida. Já a arquitetura demonstrou uma visão geral do funcionamento desta.

A metodologia de desenvolvimento adotada serviu de guia e garantiu o sucesso do projeto de *software*. Inicialmente foram escolhidas as ferramentas adequadas e que foram de encontro aos requisitos da organização.

Em paralelo com o desenvolvimento da aplicação foram criadas rotinas de testes, sendo entregues várias versões do *software* à medida que eram concluídos, o que foi possível através da utilização do modelo incremental.

No que concerne à implementação e testes, seguiram-se os padrões da organização e aplicou-se a solução em lojas piloto, sendo isto essencial para perceber o comportamento da aplicação num ambiente de produção.

Os resultados demonstraram que foram cumpridos os objetivos definidos inicialmente, ou seja, foi possível efetuar a migração do sistema operativo na organização e reduziu-se significativamente ou eliminou-se mesmo, o número de incidências.

No fundo o artefacto desenvolvido foi capaz de garantir a continuidade dos serviços da organização, garantindo a confiança no funcionamento e evitando custos desnecessários. Com a resolução dos problemas foi possível aumentar a qualidade de serviço prestada ao cliente interno e externo.

Como trabalho futuro propõe-se que a aplicação aqui apresentada seja desenvolvida e adequada a outros contextos organizacionais bem como ao mercado de retalho, na medida em que a massificação das tecnologias a que hoje assistimos origina novas ameaças que podem levar a falhas do sistema informático. Neste contexto, conforme foi demonstrado, a aplicação de *software* desenvolvida pode contribuir significativamente para evitar algumas dessas falhas.

9. Bibliografia

- Alves, M., & Vasques, F. (1998). *Confiança no Funcionamento Conceitos Básicos e Terminologia*. Porto.
- Bourque, P., & Fairley, R. E. (2014). *SWEBOK, Guide to the Software Engineering Body of Knowledge, Version 3.0*. IEEE Computer Society.
- Chervenak, A. L., Vellanki, V., & Kurmas, Z. (1998). Protecting File Systems: A Survey of Backup Techniques / Joint NASA and IEEE Mass Storage Conference.
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (1 de March de 2004). Design science in information systems research. *MIS Quarterly*, pp. 75-105.
- Hui, L. Y., Seok, K. H., & Ki, K. B. (2015). Desktop Computer Virtualization for Improvement Security, Power Consumption and Cost by SBC(Server Based Computer).
- Laprie, J.-C. (1985). *Dependable Computing Concepts and Fault Tolerance: Concepts and Terminology*. Toulouse, France.
- Laprie, J.-C., Avizienis, A., & Randell, B. (2000). *Fundamental Concepts of Dependability*. Newcastle, England.
- Lopes, P. (2004). *Avaliação da Confiança no Funcionamento de Redes de Campo*. Porto, Portugal.
- Microsoft Corporation. (15 de 10 de 2015). *Virtual Disk*. Obtido de Microsoft: [https://msdn.microsoft.com/en-us/library/windows/desktop/dd323684\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/dd323684(v=vs.85).aspx)
- Peffer, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, pp. 45-78.
- Pressman, R. (2010). *Software engineering : a practitioner's approach*. McGraw-Hill.
- Pullum, L. L. (2001). *Software Fault Tolerance Techniques and Implementation*. Norwood: ARTECH HOUSE.
- Shah, M. (2002). *Fault Tolerant Distributed Computing*. Texas, U.S.
- Trivedi, K. S., Kim, D. S., Roy, A., & Medhi, D. (2009). *Dependability and Security Models*. USA.