Plano de testes

Laboratório de Gestão de Projectos
FEUP, 06 de Junho de 2008

Responsável pelo documento

Luís Matias

Revisto por

Paulo Marques

Validado por

Hugo Zenha

Histórico

Versão	Descrição	Data
0.1	Estruturação do Relatório	30 De Março de 2008
0.2	Elaboração dos Capítulos Iniciais	1 De Abril de 2008
0.3	Elaboração do Capítulo de Visão Geral	3 De Abril de 2008
0.4	Elaboração do Capítulo de Cobertura, Abordagens e Critérios	4 De Abril de 2008
0.5	Acoplamento dos Capítulos de Necessidades e Agenda e Definição Preliminar de Acrónimos e Abreviaturas	5 De Abril de 2008
0.6	Acoplamento dos Capítulos de Riscos, Responsabilidades e Documentação. Término dos Acrónimos e Abreviaturas. Versão pronta para Revisão e Validação.	5 De Abril de 2008
1.0	Validação do Documento	6 De Abril de 2008
1.1	Actualização dos conteúdos	6 De Junho de 2008

Resumo

O Plano de Testes foi desenvolvido no âmbito da *indago*, enquanto empresa envolvida no projecto *Seed* para *Novabase – Octal2Mobile*. Este tem como propósito definir as metodologias dos Testes à aplicação.

Índice

Lista de Figuras

Não existem figuras neste documento.

Lista de Tabelas

1 Introdução

1.1 Objectivo

Criar um plano de testes não é diferente de criar qualquer outra parte dum projecto de *software*. Um plano possibilita que a decisão possa ser tomada tendo em conta os objectivos do projecto e como eles são cumpridos, que recursos estão disponíveis, para quando os diferentes componentes do projecto são necessários e com que qualidade, enquanto se controlam os riscos.
O Plano de Testes tem então como objectivo a definição do planeamento para a execução dos Testes, tendo em conta os seguintes tópicos:

- Abrangência;
- Abordagem;
- Recursos;
- Cronograma;
- Riscos;

No que à abrangência diz respeito, são identificadas as funcionalidades a serem, ou não, testadas bem como a forma como estas vão ser validadas pelos testes. Na Abordagem irá essencialmente definir-se a forma de abordagem de cada um dos itens de testes, que documentação de Testes será elaborada e as diferentes tarefas a levar a cabo para completar o processo de Testes. Ao nível dos Recursos, serão definidas as necessidades do ambiente de teste bem como as diferentes responsabilidades dentro da equipa sobre os Testes de aplicação. O cronograma é a definição da agenda para os Testes.
Finalmente, e tendo em conta todas os itens anteriores, é necessário ter em conta na elaboração processo de Testes, os grandes riscos presentes nesse processo.

1.2 Âmbito

Este Plano de Testes é realizado no âmbito do Projecto *Seed* da equipa *indago*, realizado para a Novabase no âmbito da disciplina do 4º ano, 2º Semestre do Mestrado Integrado de Engenharia Informática e Computação, Laboratório de Gestão de Projectos

1.3 Definições, Acrónimos e Abreviaturas

Termo	Descrição
Comunicação Wireless	A comunicação *wireless* (ou simplesmente *wireless*) é a transferência de informação ao longo duma determinada distância sem o uso de condutores físicos ou fios.
Listener	Elemento do projecto, incluído no lado do servidor, que estará sempre à "escuta" de informação enviada por um dispositivo móvel. Um listener é constituído por um conjunto de condições-acções.
CPU	A *Central Processing Unit*, ou normalmente chamada processador, é a descrição da classe de máquinas lógicas que executam programas de computador.
GPS	*Global Positioning System* é um sistema de posicionamento por satélite utilizado para determinação da posição de um receptor na superfície da Terra
Framework	É uma estrutura de suporte definida em que um outro projeto de software pode ser organizado e desenvolvido. Um framework pode incluir programas de suporte, bibliotecas de código, linguagens de script e outros softwares para ajudar a desenvolver e juntar diferentes componentes de um projeto de software.
PDA	*Personal Digital Assistant* (também conhecido como Handheld), ou Assistente Pessoal Digital, é um computador de dimensões reduzidas, cumprindo funções de agenda e sistema informático de escritório elementar, com possibilidade de interconexão com um computador pessoal e uma rede informática sem fios. Pode ser dotado de um sistema GPS.
Ecclema	A Ecclema é uma *framework* de análise de graus de cobertura de código em testes para a plataforma de desenvolvimento Eclipse.
GSM	*Global System for Mobile Communications* implementa um sistema global para comunicações móveis.

Termo	Descrição
GPRS	*General Packet Radio Service* é uma tecnologia que aumenta as taxas de transferência de dados nas redes GSM existentes.
FindBugs	O *FindBugs* é uma aplicação que procura por bugs em código Java. Utiliza análise estática para identificar vários tipos de erros em programas desenvolvidos em código Java.
Framework	Uma framework é uma estrutura conceptual utilizada para resolver problemas mais complexos.
IEEE	O *Institute of Electrical and Electronics Engineers* é uma organização internacional, profissional e sem fins lucrativos para o avanço da tecnologia relacionada com a electrotecnia e computadores.
USB	*Universal Serial Bus* é um tipo de conexão Plug and Play que permite a conexão de periféricos sem a necessidade de desligar o computador.
JUnit	A JUnit é uma *framework* de testes unitários para linguagem de programação Java.
PMD	O PMD é uma ferramenta de análise automática de código baseada em regras de sintaxe e semântica estáticas para linguagem Java.

Tabela 1 - Termos

1.4 Referências

[1] **Página da *Wikipedia* sobre a norma IEEE 829**
-> Wikimedia
Acessível em *http://en.wikipedia.org/wiki/IEEE_829*
(Acedido em Abril 2008)

[2] **Norma IEEE 829**
-> IEEE
Acessível em *http://se.inf.ethz.ch/teaching/ss2005/0050/exercises/REMOVED/IEEE%20Std%20829-1998%20test.pdf*
(Acedido em Março 2008)

[3] **Testes de Usabilidade**
-> Luís Matias e Rui Pinto
Trabalho realizado no âmbito da disciplina de Testes e Qualidade de Software do MIEIC, FEUP no ano lectivo de 2007/2008
(Consultado em Abril 2008)

[4] **Página Oficial do JUNit**
-> Object Mentor.
Acessível em *http://www.junit.org/*
(Acedido em Março 2008)

[5] **Página Oficial do EclEmma**
-> EMMA Project
Acessível em *http://www.eclemma.org*
(Acedido em Março 2008)

[6] **Página oficial do FindBugs**
-> University of Maryland
Acessível em *http://FindBugs.sourceforge.net/*
(Acedido em Março 2008)

[7] **Página oficial do PMD**
-> PMD Project
Acessível em *http://PMd.sourceforge.net*
(Acedido em Março 2008)

[8] **PLQP -** *Plano de Qualidade do Projecto*
-> *indago*
(Acedido em Abril 2008)
[9] **RERE -** *Relatório de Especificação de Requisitos*
-> *indago*
(Acedido em Abril 2008)

[10] **RIGP -** *Relatório Intermédio de Gestão de Projecto*
-> *indago*
(Acedido em Abril 2008)

1.5 Estrutura do Documento

O Documento possui uma capa, um histórico, um resumo, um índice, uma lista de figuras e um conteúdo com a Introdução e o Planeamento de Testes.

A Introdução divide-se em Objectivo e Âmbito, descrição de Definições Acrónimos e Abreviaturas, Referências bibliográficas e a descrição da Estrutura deste Documento.

O Planeamento de Testes divide-se nas seguintes secções:

- **Visão Geral** - Definição Geral dos Testes do Produto;

- **Cobertura** - Definição dos itens a ser testados, bem como as funcionalidades que devem, ou não, ser testadas;
- **Abordagem e Critérios** - Define a forma geral de abordar os testes, o conjunto de tarefas a levar a cabo, e ainda os critérios de aceitação e suspensão de testes;
- **Documentação** - Descreve a documentação de testes a produzir no Projecto;
- **Necessidades** – Descreve, de forma generalista, as necessidades para realizar testes de *software*;
- **Responsabilidades** - Define quem da equipa testa o quê no projecto;
- **Agenda** - Definição das principais datas para a realização dos testes;
- **Riscos** - Definição dos grandes riscos inerentes ao plano de testes.

2 Planeamento dos Testes

2.1 Visão Geral

Esta aplicação consiste na criação dum conjunto de funcionalidades para localização e filtragem da posição geográfica de dispositivos com tecnologia GPS. Para isso são disponibilizadas ferramentas que possibilitam que um dispositivo, desde que ligado, envie periodicamente as suas coordenadas GPS para um servidor remoto, onde estas podem ser armazenadas e processadas. O Servidor remoto pode obter várias configurações, entre as quais a criação de limitações às coordenadas GPS dum dispositivo e à criação de mecanismos de alarme para quando estes se encontram fora dessas limitações. Esta Framework pode ser utilizada como base para a definição de várias aplicações com diferentes utilizações e conceitos como Gestão de Frotas ou a definição de Portagens Virtuais.

Uma vez que a utilização deste software implica a utilização de vários dispositivos do tipo PDA (ou outros com tecnologia GPS) separados fisicamente, os testes ao *software* irão incidir na sua maioria nas funcionalidades centradas no servidor, funcionalidades essas muito mais críticas para o bom funcionamento da aplicação. As funcionalidades da aplicação que irão ser testadas de forma mais crítica são as seguintes:

- Envio de Dados do Cliente para o Servidor;
- Recepção e Armazenamento de Dados;
- Processamento de Dados;
- Mecanismos de Aviso;
- Configuração dos "Filtros" de Informação.

Para que tal aconteça, irá ser definida a estratégia de teste, que define basicamente o nível e o critério de teste a ser utilizado. O nível de teste está intrinsecamente ligado à fase de

desenvolvimento do *software* em que o teste será aplicado bem como as tecnologias (*FindBugs* ou *Ecclema*, por exemplo) e técnicas usadas (ex.: caixa preta *vs.* caixa branca). No nosso projecto, visto tratar-se de uma mera Framework, iremos apenas utilizar Testes de Unidade.

Este plano de testes, assim como toda a documentação de testes subsequente a este plano irá ter em conta a norma IEEE 829 - *IEEE Standard for Software Test Documentation*. Para criar este documento, vamos também utilizar a documentação da aplicação já existente, entre as quais:

- *indago*, PLQP - Plano de Qualidade do Projecto, 2008
- *indago*, RERE - Relatório de Especificação de Requisitos, 2008
- *indago*, RIGP - Relatório Intermédio de Gestão de Projecto, 2008

2.2 Cobertura dos Testes

Neste capítulo, serão definidos os itens e as funcionalidades de teste. Deve considerar-se como itens de teste tudo aquilo que é avaliável e mensurável em testes de *software*. Nesta aplicação, serão considerados como itens de teste as funcionalidades do projecto os seguintes:

- **Exequibilidade**: a funcionalidade é executável sem erros e produz resultados;
- **Coerência funcional**: a funcionalidade executa o que é suposto;
- **Coerência de requisitos**: a funcionalidade dá resposta aos requisitos da aplicação;
- **Eficiência**: a execução da funcionalidade utiliza o mínimo de recursos (lógicos e físicos) que necessita para desempenhar a sua função;
- **Usabilidade**: a funcionalidade é usável pelo grupo de utilizadores tipo do produto;

Tendo em conta estes itens de teste, são definidas em seguida e de forma sumária, as funcionalidades que deverão ser testadas, bem como uma escala relativa da importância da existência de testes para essas funcionalidades. Esta escala divide-se entre os seguintes valores: Alta, Média e Baixa e deverá ser utilizada caso se verifique a necessidade de suprir um conjunto de testes à aplicação, devendo ser supridos aqueles que tiverem menor importância para a qualidade do produto final. As funcionalidades que deverão ser testadas são as seguintes:

Funcionalidade	Descrição da criticidade do Teste	Nível
Enviar dados (Cliente)	É importante que o programa cliente consiga estabelecer uma ligação de dados com o servidor e os envie de forma periódica, para seu posterior processamento.	Média
Guardar dados (Cliente)	É também importante que o programa cliente consiga armazenar os dados que não consiga temporariamente enviar para posterior envio, uma vez que garante que não existe qualquer perda de dados, factor que é importante para a performance do produto	Média
Receber dados	Caso o servidor não consiga receber os dados em perfeitas condições, a aplicação não tem qualquer sentido de existir.	Alta
Processar dados	O processamento dos dados garante a lógica de negócio da existência da aplicação o que torna o seu funcionamento crítico.	Alta
Guardar dados	Caso o servidor não consiga armazenar os dados que recebe, não conseguirá também processá-los *a posteriori*. Essencial.	Alta
Gerar acções	As acções geradas em consequência do processamento dos dados garantem também a lógica de negócio da aplicação.	Alta
Extrair dados	Partindo do pressuposto que os dados estão armazenados na base de dados, a forma como eles são ou não consultados, bem como a sua coerência, não completam por si só um processo crítico. Contudo, este tipo de relatórios é também importante para a lógica de negócio do produto.	Média
Ver *listener*	Em relação aos pares critério/acção, a sua visualização não é essencial mas sim o seu bom funcionamento e performance.	Baixa
Modificar *listener*	O funcionamento da alteração dum par critério/acção é essencial pois o funcionamento dos *triggers* é essencial para a lógica de negócio da aplicação.	Alta
Adicionar *listener*	A criação dos pares critério/acção é uma funcionalidade essencial.	Alta

Tabela 2 - Funcionalidades a ser testadas e o seu grau de importância de teste.

2.3 Abordagem e Critérios

2.3.1 Metodologia

Pretende-se com este processo estabelecer um relacionamento de confiança e segurança, garantindo no produto cinco características essenciais:

- Fiabilidade;
- Portabilidade;
- Usabilidade;
- Segurança;
- Performance

Neste capítulo vai definir-se como abordar os testes ao software, os itens pretendidos e as funcionalidades a testar. Para efectuar os testes ao produto, iremos utilizar apenas testes unitários do tipo caixa preta. A razão para esta escolha reside no facto do nosso produto ser uma Framework e não possuir, de origem, por exemplo, qualquer tipo de interface. Podemos dividir os testes em dois módulos diferentes.

- Codificação dos módulos do sistema – Teste de Unidade ou Unitário;
- Cobertura dos requisitos – Teste de Sistema/Integração;

No capítulo de visão geral foi referido um quarto tipo de testes (Regressão) relativo ao período de manutenção do produto, teste esse que irá ser deixado de fora desta abordagem.

Tendo em conta estes tipos de testes, estes serão essencialmente vocacionados para a fase de desenvolvimento.

No sentido de dar resposta aos itens de teste, será seguida uma ordem de tarefas (metodologia) de teste que é a seguinte:

- Verificação de resposta a requisitos;
- Verificação manual de falhas no código fonte;
- Realização de testes unitários com cobertura total ao código fonte das funcionalidades a testar;
- Estudo da carga de utilização de recursos lógico/físicos.
- Realização de testes de integração com cobertura total ao código fonte das funcionalidades a testar;

No sentido de dar resposta à metodologia de testes acima definida, será usado um conjunto de metodologias e ferramentas de testes e qualidade de *software* que se define em seguida:

- Framework de testes unitários (*NUnit*);
- Framework de análise do grau de cobertura de código fonte (*NCover*);
- *Test CheckList*;

No sentido de utilizar a presente definição de técnicas e ferramentas de testes de *software* na metodologia, começa por utilizar-se a *Test CheckList*. A *Test Checklist* é uma simples lista de condições que se devem verificar no código ou na aplicação. Neste caso, irão ser definidas várias *Test Checklist* ao longo do desenvolvimento do Projecto para verificar que o seu desenvolvimento vai de encontro aos requisitos e arquitectura do produto. Ao longo do desenvolvimento do mesmo código fonte, este deverá sempre ser revisto antes de ser testado por alguém que não tenha participado no seu desenvolvimento. Antes da fase de integração, deve proceder-se a duas tarefas: cobrir o código com uma bateria de testes unitários, que testam a sua coerência e fiabilidade com o recurso a uma *framework* com funcionalidades desse tipo; efectuar um estudo da carga de utilização de recursos lógico/físicos, isto é, se o código fonte que se possui explora racionalmente os recursos lógicos e físicos (CPU, memória, portas e outras infra-estruturas de comunicação, etc.) que tem à sua disposição. Após o código estar devidamente validado, deve partir-se para a integração do produto, que deverá ocorrer sem problemas caso os testes definidos anteriormente tenham sido eficazes. Finalmente procede-se aos testes finais de integração.

2.3.2 Critérios

Ao definir os critérios de teste, definem-se medidas claras para avaliar e conduzir os testes de *software*. Dentro dos critérios, podem definir-se dois tipos: os critérios de aceitação de testes e os critérios de suspensão de testes. Os critérios de aceitação definem as razões pelas quais os testes passam/chumbam enquanto os critérios de suspensão descrevem as situações em que os testes devem ser suspensos para serem retomados mais tarde.

Os critérios de aceitação de testes devem ser definidos em conjunto com a qualidade definida nas métricas de Teste presentes no PLQP - Plano de Qualidade do Projecto. Devemos ter sempre em conta que as saídas obtidas terão de ser 100% consistentes com as saídas previstas.

Os critérios de suspensão dos testes deverão ser aqueles que prejudicam no presente ou no futuro o desempenho ou desenvolvimento da aplicação, sendo apontados os seguintes:

- Existe sobreaquecimento do *hardware*;
- Existe corrupção da base de dados;
- Não há comunicação entre cliente e servidor;
- O sistema não responde;

- Existe um comportamento inesperado da aplicação;
- Existe outro tipo de perigos para a segurança dos utilizadores, do software ou do hardware;

Nos casos acima descritos, o teste deverá ser suspenso e só continuado após ter ocorrido o código fonte e/ou outros erros de projecto (desenho de arquitectura, etc.). Este processo deverá ocorrer também de acordo com a necessidade de qualidade descrita pelo PLQP.

2.4 Documentação

Neste capítulo identifica-se os diversos documentos que serão criados na documentação de testes.
Os documentos serão os seguintes:

- Plano de Teste:
  Plano onde são definidos quais os testes a serem realizados no projecto. Enumera todos os itens e funcionalidades no *software* que necessitam de ser testados, bem como grande parte das especificações gerais de teste.

- Plano de Especificação de Testes:
  O Plano de Especificação de Testes tem o objectivo de definir de forma concreta toda a metodologia de teste seguida durante todo o desenvolvimento do projecto de software. Podemos dividir a Especificação de Testes em três tarefas distintas: a Especificação de Desenhos de Teste, a Especificação de Casos de Teste e a Especificação de Procedimento de Testes.

  A Especificação de Desenhos de Teste refina a abordagem de teste e as funcionalidades que devem cobertas pelo Desenho de Teste e os casos de teste a si associados, bem como os critérios de passagem/chumbo de testes. A Especificação de Casos de Teste documenta os valores utilizados actualmente para *input* bem como a antecipação do *output* esperado. A Especificação de Casos de Teste define ainda as condições que devem resultar do uso do caso de utilização especificado, sendo que a separação desta especificação dos Desenhos de Teste permite o uso destes casos de teste em mais que um desenho e/ou situação. Por sua vez, a Especificação de Procedimentos de Teste identifica todos os passos necessários para utilizar o sistema e executar os casos de teste especificação no sentido de implementar o desenho de teste associado.

- Relatório de Testes:
  O Relatório de Testes tem o objectivo de documentar toda a informação relativa ao processo de teste de *software*. Podemos dividir o Relatório de Testes em quatro relatórios distintos: o Relatório da Localização dos Componentes de Teste, o

Registo de Testes, o Relatório de Incidentes e Anomalias e o Relatório de Sumário de Testes.

O Relatório da Localização dos Componentes de Teste identifica os itens de teste. O Registo de Testes é usado pela equipa de testes para documentar tudo o que acontece durante a execução de testes. O Relatório de Incidentes e Anomalias descreve qualquer evento que tenha ocorrido durante a execução de testes que requer uma análise mais profunda *a posteriori*. O Relatório Sumário de Testes resume todas as actividades de testes associadas com um ou mais especificações de desenho de testes.

2.5 Necessidades

Neste capítulo, irão ser descritas as necessidades que a prática dos testes de *software* exige neste projecto, quer em termos de ambiente de teste, quer em termos de equipamento e de execução.

2.5.1 Equipamento

Para os testes irão ser necessários os seguintes equipamentos para além dos terminais de trabalho e desenvolvimento dos respectivos programadores:

- Um dispositivo móvel equipado com dispositivo GPS para recepção de coordenadas GPS e a correr a plataforma Windows Mobile 5, assim como um cabo USB do tipo A para B padrão.

- Para a componente servidora será necessária uma máquina servidora, para escuta de pedidos, assim como uma base de dados associada a esse mesmo servidor.

2.5.2 Equipamento de *software*

Além do software de desenvolvimento do Projecto, serão necessárias as seguintes *frameworks*:

- *Framework* de testes unitários (*NUnit*);
- *Framework* de análise de cobertura de código (*NCover*);

2.5.3 Instalação

Os dispositivos móveis deverão ser ligados via cabo USB aos computadores de desenvolvimento, sendo a instalação no dispositivo efectuada a partir da plataforma Visual Studio 2008 e tendo como alvo o dispositivo móvel em questão.

2.5.4 Execução

Após efectuada a instalação, a aplicação cliente deverá ser executada manualmente no dispositivo móvel. No final do teste, deverá garantir-se que a aplicação se encontra efectivamente fechada (não se encontra ainda residente em memória).

A componente servidora deverá ser inicializada antes de qualquer tentativa de comunicação com os dispositivos móveis, excepto para testes que requeiram especificamente tal condição.

2.5.5 Condições Ambientais

O ambiente para testes está intrinsecamente relacionado com o dispositivo móvel. Como tal, devem ser asseguradas todas as condições para o funcionamento óptimo do dispositivo, tais como especificadas pelo fabricante do mesmo. Adicionalmente, devido aos requisitos específicos a serem testados deve ser assegurada uma boa recepção dos satélites GPS. Tal implicará as rotinas de teste do dispositivo se realizem em áreas tão abertas quanto possíveis, evitando-se assim a interferência de estruturas físicas.

É também de evitar áreas sujeitas a interferências electromagnéticas, as quais poderão influenciar o envio dos dados para o servidor (no caso de comunicação *wireless*).

2.5.6 Outras necessidades

Para o desenvolvimento dos *Test Checklist* (ver capitulo ***2.3 Abordagem e Critérios***) irão ser utilizadas, na pior das hipóteses, papel e lápis ou uma aplicação do estilo bloco de notas. Responsabilidades
Neste capítulo definiu-se dentro da Indago uma equipa de Testes responsável por efectuar todo o processo de testes ao *software* desenvolvido ao longo de todo o projecto. A equipa de testes é composta pelos seguintes cargos:

- Gestor de Testes (Luís Matias): É da responsabilidade do gestor fazer o planeamento dos testes e do esforço de teste, garantindo que os requisitos de qualidade são devidamente cumpridos com a maior eficiência possível. É da responsabilidade deste a distribuição dos recursos (humanos e materiais) com vista a melhorar a eficiência do processo de testes.

- Analista de Testes (Filipe Castro): O Analista de Testes é o responsável dentro da equipa que define quais as funcionalidades a testar, garantindo o cumprimento dos requisitos definidos. Este projecta os casos de teste e os resultados esperados, avaliando a qualidade do produto.

- **Designer dos Testes** (João Bernardes): É da responsabilidade do *designer* dos testes definir a maneira mais apropriada as seguintes actividades para a execução dos testes:
    - Escolha das ferramentas mais adequadas;
    - Definição dos testes automatizados/manuais;
    - Manutenção de uma plataforma automatizada de testes.

- ***Testers*** (Rui Lopes e Nuno Cruz): São responsáveis por executar os testes previamente definidos (testes não automatizados) e pelo registo dos resultados.

- ***Test developers*** (João Costa, António Barbosa e Paulo Marques): São responsáveis pela implementação do código de testes e de apoio aos mesmos.

2.6 Agenda

O planeamento de testes é necessário como medida de controlo da evolução do projecto. Dessa forma, a Indago apresenta o calendário de testes que deverá ser seguido pelos envolvidos.

Protótipo de Concepção	
29-03 a 31-03	Testes Unitários
04-04	Teste de Integração
Protótipo de Desenvolvimento	
10-04 a 19-05	Testes Unitários sobre Protótipos Incrementais
20-05 a 23-05	Testes Finais de Integração
26-05 a 29-05	Testes de Validação e Aprovação

Tabela 3 - Agenda de Testes

É importante referir que a Indago não é alheia à eventualidade de mudanças ou ajustes necessários ao longo do processo de desenvolvimento. A confiança na capacidade de adaptação e nos conhecimentos técnicos da equipa, bem como os mecanismos internos de controlo de evolução e qualidade do projecto permitem que seus elementos se considerem flexíveis o suficiente para ajustar tanto os testes, como as suas datas, de acordo com as necessidades verificadas.

2.7 Riscos

Aqui são definidos os maiores riscos na execução deste plano de testes, bem como planos de contingência para os contornar/minimizar.

Risco	Plano de Contingência
Perda avaria ou destruição do dispositivo móvel para testes.	Negociar com os clientes a atribuição de um novo dispositivo
Número insuficiente de dispositivos para os testes.	Negociar com os clientes a atribuição do número adequado de dispositivos
O código e a arquitectura não ser testável ou tornar a tarefa muito complicada.	Fazer *refactoring* de forma a melhorar a testabilidade do produto.
Desenvolvimento das funcionalidades não fica pronto a tempo de ser testadas devidamente.	Redefinição das prioridades dos testes.
Requisitos de qualidade requerem muitos mais testes do que é possível efectuar.	Redefinição dos requisitos de qualidade e/ou redefinição de prioridades de teste.
Dificuldades na preparação e configuração do ambiente de testes.	Obter o conhecimento e competências necessárias à preparação do ambiente de testes
Testes necessitam de deslocações demoradas e/ou custosas e/ou longínquas (de notar que se testa um sistema de GPS).	Simular os dados dessas posições
Problemas nos serviços dos quais dependem os produtos em teste, nomeadamente GPS e GPRS	Simular os dados de testes
Análise dos testes ser mal definida	Redefinição da análise dos testes
Desenho dos testes mal definido	Redefinição do desenho dos testes
Não existirem recursos suficientes para efectuar os testes.	Alocar mais recursos para os testes ou redefinir prioridades dos testes.
São detectados muitos erros ou erros graves durante os testes.	Redefinir a prioridade dos testes. Alocar recursos atribuídos aos testes à investigação dos erros encontrados.

Risco	Plano de Contingência
O código necessário para os testes é muito complexo e/ou trabalhoso.	Alocar mais recursos para os testes e/ou redefinir prioridades dos testes e/ou redefinir metodologia de testes.
Algumas das funcionalidades não terem possibilidade de teste automático, e serem muito complicadas/trabalhosas de testar	Alocar mais recursos para os testes e/ou redefinir prioridades dos testes e/ou redefinir metodologia de testes.
Os testes têm erros.	Reduzir a complexidade dos testes, de forma a minimizar os erros dos mesmos.

Tabela 4 - Riscos nos Teste