Departamento de Engenharia Informática

TRABALHO FINAL DE CURSO

do Curso de
LICENCIATURA EM ENGENHARIA INFORMÁTICA E DE COMPUTADORES (LEIC)

Ano Lectivo 2005 / 2006

N.º da Proposta: 150

Título: MobileREVS – votação electrónica

Professor Orientador:

Prof. Doutor Paulo Jorge Pires Ferreira _____(assinatura)_____

Co-Orientador:

Mestre Rui Filipe Lopes Joaquim (INESC-ID) _____(assinatura)_____

Alunos:

51041, Luís Miguel Silva Costa _____(assinatura)_____

51048, Nuno Alexandre Costa Fresta dos Santos_____(assinatura)_____

Resumo

Actualmente, um pouco por toda a parte, multiplicam-se experiências com vista a ganhar a confiança necessária para a utilização de sistemas de votação electrónica (SVE) em eleições.

O REVS (*Robust Electronic Voting System*) é um SVE, concebido para a Internet, que apresenta as características fundamentais dos sistemas de votação: democracia, privacidade, correcção e verificabilidade. Assim, e com base no sistema REVS, é aqui apresentado o sistema MobileREVS que foi desenvolvido para telemóveis, possibilitando um meio de votação mais facilmente acessível aos eleitores.

Tendo em conta as falhas de comunicação inerentes ao uso de um SVE em ambientes móveis e/ou falhas nos servidores que suportam o sistema de votação, o MobileREVS foi concebido para permitir: (i) interromper e reatar o processo de votação sem enfraquecer o protocolo de voto original do REVS; (ii) suportar falhas nos servidores através de replicação; (iii) impedir que um só servidor, sozinho ou até um determinado nível de conluio, consiga corromper o resultado da eleição; e (iv) lidar com os problemas inerentes da utilização das redes móveis, evitando que estes afectem o protocolo de voto.

Palavras-chave: Votação, *e-Voting*, Segurança, Mobilidade, Telemóveis, REVS.

Agradecimentos

Ao Prof. Doutor Paulo Ferreira pelo acompanhamento e dedicação com que abraçou este projecto. Agradecemos também o apoio e rigor dedicados ao mesmo.

Ao Mestre Rui Joaquim pelo apoio e indispensável contributo para o sucesso do projecto, nomeadamente no esclarecimento de dúvidas e no conhecimento e *insight* partilhados.

Um agradecimento especial ao Prof. João Silva, pela disponibilidade na resolução de todos os problemas técnicos da infra-estrutura de desenvolvimento.

A todos os colegas do Grupo de Sistemas Distribuídos do INESC-ID, pelo apoio e excelente ambiente de trabalho proporcionado.

Finalmente, um grande obrigado às nossas famílias, namoradas e amigos, por todo o apoio e incentivo.

Índice

Índice de figuras

Índice de tabelas

1. Introdução

Nos últimos anos foram conduzidas várias experiências para facilitarem o processo de eleição. As facilidades foram introduzidas através de novas formas de expressar votos, além das tradicionais baseadas em papel. Exemplos de novas interfaces e sistemas de votação são os *touch screens*, mensagens SMS de telemóveis e sistemas de votação distribuídos sobre a Internet [MSOA01 , UKD]. Estes novos sistemas de votação são apelativos por várias razões:

1. as pessoas estão cada vez mais habituadas a lidar com tecnologia;
2. oferecem a possibilidade de votar fora da zona de residência;
3. suportam vários tipos de boletins de voto e permitem verificar o seu correcto preenchimento.

Tendo em conta estes aspectos e toda a inovação da tecnologia móvel somos motivados a facilitar ainda mais o processo de voto. Isto para oferecer a qualquer cidadão a possibilidade de participação numa eleição, de um modo fácil e seguro, através de um dispositivo móvel.

Hoje em dia os Sistemas de Votação Electrónica (SVE) estão cada vez mais em voga. Um pouco por toda a parte multiplicam-se experiências com vista a ganhar a confiança necessária para a utilização dos SVE em novas e mais importantes eleições.

Com o aparecimento das tecnologias móveis torna-se hoje em dia possível oferecer o máximo de comodidade aos utilizadores evitando a sua deslocação a locais específicos. Isto, aliado ao facto de actualmente os telemóveis serem dispositivos de uso corrente entre a população, pode incentivar o eleitorado a uma maior participação em eleições.

O REVS (*Robust Electronic Voting System*) é um sistema de voto para a Internet desenvolvido no INESC ID e que já foi testado no Instituto Superior Técnico [JZF03]. O sistema REVS oferece todas as garantias de segurança de um sistema de votação electrónica (correcção, democracia, privacidade e verificabilidade). Com o projecto MobileREVS pretende-se desenvolver um sistema de votação para ambientes móveis suportando as mesmas funcionalidades e propriedades de segurança do sistema REVS.

Assim, o objectivo do projecto MobileREVS consiste em desenhar e implementar um sistema de votação electrónica que suporte utilizadores com dispositivos móveis, em particular, telemóveis. O sistema em causa deve respeitar um conjunto de requisitos bastante exigentes dada a especificidade dos processos de votação. Tal como veremos de seguida, é fundamental que a correcção, a democracia, a privacidade e a verificabilidade, do processo de votação, sejam asseguradas com toda a segurança.

O sistema MobileREVS poderá ser aplicado a diversos contextos actuais, quer a nível particular quer a nível da sociedade em geral. Alguns dos principais exemplos da utilização deste sistema são:

- **Eleições**: o sistema poderá ser usado na aplicação dos direitos cívicos de cada indivíduo da sociedade ao usufruírem do seu direito de voto. O MobileREVS poderá ser uma alternativa muito forte no combate às taxas de abstenção nas eleições legislativas, autárquicas e presidenciais, entre outras. Mais do que permitir que um indivíduo vote sem necessitar de sair de casa,

permitirá que o voto seja exercido mesmo quando se encontra deslocado do local de recenseamento;

- **Sondagens**: os organismos estatísticos poderão efectuar sondagens com o levantamento do número de votos através deste sistema. Será claramente um ponto positivo na redução de custos operacionais. O processo ganha na rapidez de execução da sondagem e, principalmente, na capacidade mais fina da verificação de resultados em tempo real;
- **Referendos**: os referendos (públicos ou privados) para decisão de diversos assuntos da sociedade poderão ser efectuados de forma rápida e fidedigna usando o MobileREVS.

1.1. Processo de *e-Voting*

Nesta secção é providenciada a informação essencial para a compreensão dos SVE. Começa-se por apresentar as fases usuais de um processo de votação. Posteriormente, descrevem-se as propriedades requeridas para um SVE.

1.1.1. Fases do processo

As fases de um processo de votação electrónica são semelhantes às fases de um processo de votação tradicional, baseado em papel:

- **Registo**: A fase de registo consiste na compilação da lista de eleitores elegíveis e na entrega aos mesmos de credenciais necessárias para utilização em eleições futuras;
- **Validação**: A fase de validação consiste na verificação das credenciais dos eleitores durante a eleição. Apenas os eleitores registados com as credenciais adequadas poderão ser autorizados a votar, e apenas uma vez por eleição;
- **Recolha**: Esta fase consiste na recolha dos votos de todos os eleitores participantes;
- **Verificação**: Nesta fase é verificada a validade dos boletins de voto. Apenas os boletins válidos serão usados na fase de contagem;
- **Contagem**: A fase de contagem consiste na contagem de votos dos boletins válidos. No final a contagem é publicada;
- **Reclamação**: Nesta fase todas as reclamações devem ser efectuadas e investigadas. A fase de reclamação deve decorrer em paralelo com todas as restantes fases. No final desta fase são publicados os resultados finais.

1.1.2. Propriedades

Investigadores no ramo da votação electrónica já chegaram a um consenso global sobre as quatro principais propriedades que um sistema de votação electrónica deve possuir [CC97]:

- **Correcção**: (1) não é possível alterar um voto válido, (2) não é possível que um voto válido seja eliminado da contagem, e (3) não é possível que um voto inválido seja considerado no resultado final;
- **Democracia**: (1) permite apenas que eleitores elegíveis votem, (2) garante que os eleitores elegíveis votam apenas uma vez, e (3) garante a igualdade de conhecimento, ou seja, sem resultados parciais;
- **Privacidade**: (1) nenhuma autoridade de voto nem ninguém consegue estabelecer uma ligação entre um boletim de voto e o eleitor que o emitiu, e (2) nenhum eleitor consegue provar que votou de uma determinada maneira. A segunda parte desta propriedade existe para assegurar que ninguém consegue coagir ou subornar um eleitor. Os sistemas com esta parte da propriedade "Privacidade" são normalmente designados por sistemas de votação *Receipt-free*;
- **Verificabilidade**: qualquer pessoa pode, independentemente, verificar que todos os votos foram contados correctamente.

Correcção, democracia e verificabilidade são, na maior parte dos casos dos sistemas eleitorais actuais, assegurados pela presença de representantes de diferentes partidos. A privacidade é actualmente assegurada pela existência de cabines privadas de voto controladas pelos comités eleitorais, permitindo aos eleitores votar secretamente e sem coacção.

As quatro propriedades apresentadas são propriedades essenciais que os sistemas de votação devem possuir, mas não lidam com os requisitos operacionais que se seguem:

- **Disponibilidade**: (1) o sistema funciona correctamente durante o período de votação e (2) qualquer eleitor pode ter acesso ao sistema desde o início até ao fim desse período;
- **Capacidade de continuação diferida**: o sistema permite que qualquer eleitor que interrompeu o seu processo de votação o possa continuar ou reiniciar durante o período de votação;
- **Resistência ao conluio**: a resistência ao conluio mede a capacidade de um sistema resistir imperturbado quando qualquer entidade ou grupo de entidades eleitorais se comporta incorrectamente. Se todas as entidades de supervisão conspirarem esta propriedade não é garantida. Por isso, esta característica deve ser medida em termos do número total de entidades que devem conspirar para garantir uma interferência bem sucedida na eleição.

Diz-se que um sistema é robusto se possuir os três últimos requisitos descritos.

1.2. Organização do documento

No Capítulo 2 são descritos os conceitos fundamentais ligados ao sistema MobileREVS, tais como as técnicas criptográficas mais utilizadas em SVE, as tecnologias associadas e alguns aspectos de segurança. O Capítulo 3 apresenta um conjunto de trabalhos relacionados com o MobileREVS, com especial foco na arquitectura e protocolo do REVS, o SVE de base para o desenvolvimento deste projecto. A arquitectura do MobileREVS é introduzida no Capítulo 4. São apresentados os requisitos do sistema e alguns dos problemas que o mesmo deve considerar. Segue-se

uma descrição da arquitectura geral do sistema e do seu protocolo, complementada com uma visão mais detalhada da arquitectura do Módulo Eleitor. No Capítulo 5 são descritos os pormenores de implementação do sistema, nomeadamente do Módulo Eleitor, dos servidores e do "REVS Ballot Editor", um editor gráfico de boletins de voto para o REVS e MobileREVS. A avaliação do sistema foi agendada para o Capítulo 6. Nele são analisados os aspectos qualitativos relativos ao cumprimento das propriedades pretendidas para este sistema, seguidos da apresentação de algumas métricas de desempenho do mesmo. As conclusões finais são explicitadas no Capítulo 7, reforçando-se as vantagens e desvantagens da utilização deste sistema. A terminar são apresentadas algumas sugestões futuras para o contínuo melhoramento do MobileREVS, que não foram contempladas no âmbito deste projecto.

2. Conceitos fundamentais

Neste capítulo são apresentados os conceitos fundamentais relacionados com o projecto MobileREVS. Inicialmente é feita uma análise a algoritmos e processos criptográficos ligados a sistemas de votação electrónica. Posteriormente são introduzidas as tecnologias existentes para as plataformas móveis, sobre as quais recairá o desenvolvimento do projecto.

2.1. Criptografia

Seguidamente apresentam-se algumas das técnicas criptográficas mais usadas no contexto dos SVE.

2.1.1. Cifra simétrica e assimétrica

As técnicas de cifra de mensagens podem ser divididas em duas subclasses: a cifra simétrica e a cifra assimétrica.

A **cifra simétrica** [PKC] permite que o processo de cifra e decifra de conteúdos seja realizada através da utilização de uma única chave. O processo é simétrico no sentido em que as operações de cifra e decifra utilizam a mesma chave. Ou seja,

```
Cifra[k](M) = M' ---> Decifra[k](M') = M
```

A cifra de uma mensagem *M* com a chave *k* permite obter a mensagem cifrada *M'*. No processo inverso, a decifra utiliza a mesma chave *k* para produzir a mensagem original *M* a partir da mensagem cifrada *M'*.

Alguns dos mais conhecidos algoritmos de cifra simétrica são: DES/3DES, AES, IDEA, Blowfish, RC4/RC5 e A5 [EA , SKC].

Na **cifra assimétrica** [PKC] o processo de cifra e decifra de conteúdos é realizado através da utilização de duas chaves distintas: uma pública, *k1*, e outra privada, *k2*. O que é cifrado com a chave pública só pode ser recuperado com a chave privada e vice-versa.

```
Cifra[k1](M) = M' ---> Decifra[k2](M') = M
Cifra[k2](M) = M" ---> Decifra[k1](M") = M
```

A denominação do par chave pública/privada deve-se ao facto de uma delas ser disponibilizada publicamente e a outra ser mantida em segredo. Qualquer interveniente que pretenda comunicar seguramente com o proprietário desse par de chaves deve utilizar a chave pública. Por outro lado, a chave privada deve ser guardada em segredo pelo seu proprietário, que mantém a exclusividade da sua utilização. Os algoritmos mais conhecidos de cifra assimétrica são o RSA [RA] e o ElGamal [EGE].

Existem entidades, denominadas Autoridades de Certificação (AC), que gerem este tipo de chaves. Se uma entidade *X* requerer um par de chaves, estas ser-lhe-ão

dadas pela AC após confirmar (de alguma forma) que *X* é de facto quem diz ser. A AC compromete-se a guardar segredo da chave privada.

Assim, sempre que alguém queira contactar *X* com a certeza que só *X* vai entender a mensagem enviada, basta obter a chave pública de *X* (e.g. previamente afixada num local público) e enviar a mensagem cifrada com essa chave. Noutra perspectiva, para *Y* ter a certeza que uma mensagem foi de facto enviada por *X*, *X* terá que a cifrar com a sua chave privada e *Y* terá que a decifrar com a chave pública de *X*. É nesta base que assenta o conceito das assinaturas digitais (ver secção seguinte). No que respeita aos sistemas de votação electrónica estas técnicas são normalmente usadas para a geração de canais de comunicação seguros entre as várias entidades.

2.1.2. Assinaturas digitais

As assinaturas digitais são um método de autenticação da informação, análogo às assinaturas físicas em papel, implementado com técnicas baseadas em cifra assimétrica. Estas assinaturas satisfazem três propriedades fundamentais de segurança:

- **Autenticidade** – uma assinatura permite ao receptor verificar que uma determinada mensagem foi de facto enviada por quem se afirma como o seu emissor;
- **Integridade** – uma assinatura permite verificar que uma determinada mensagem não foi alterada;
- **Não repúdio** – o emissor de uma assinatura não pode negar a sua autenticidade.

Por razões de desempenho utilizam-se funções de resumo (e.g. SHA-1, MD5), também designadas de *hashing* ou *digest*. Um resumo permite reduzir uma mensagem a uma sequência de bits de tamanho fixo. Este resumo deve possuir as seguintes características:

- Impossibilitar a determinação da mensagem original a partir do seu resumo;
- Parecer aleatório, mesmo que o algoritmo seja conhecido. A alteração de uma pequena porção da mensagem original deve produzir grandes alterações no resumo gerado;
- Dificultar a determinação de uma mensagem diferente que produza o mesmo resumo que a mensagem original.

O processo de assinatura digital é simples: é criado um resumo da mensagem original *M*, que é depois assinado com a chave privada do emissor, *k2*, e anexado à mensagem para envio.

```
r = Resumo(M)
a = Assinatura[k2](r)
M' = M,a
```

O receptor deve validar a correcção da mensagem mediante a assinatura do emissor. Para isso deve gerar um resumo da mensagem recebida e compará-la com o resumo presente na assinatura, obtido a partir do algoritmo de verificação da assinatura com a chave pública do emissor, *k1*:

```
M' = M,a
r = Verificação[k1](a)
r' = Resumo(M)
r = r' (?)
```

Se o resumo enviado e o resumo produzido forem iguais então a mensagem é simultaneamente íntegra e autêntica.

Assinaturas cegas

As assinaturas cegas são uma subclasse das assinaturas digitais. A técnica de assinatura cega permite que uma entidade *X* assine uma mensagem de *Y* sem saber o conteúdo da mesma. Informalmente, o método consiste em *Y* enviar a *X* uma mensagem num envelope fechado (na prática, aplicar um factor de cegamento). O interior do envelope está revestido a papel químico. Seguidamente, *X* assina o envelope e a assinatura passa através do papel químico para a mensagem. Ao receber de volta o envelope assinado, *Y* retira a mensagem de dentro do mesmo (desfaz o cegamento) e pode verificar que a mensagem está assinada por *X*.

O algoritmo original foi apresentado por David Chaum [Chaum82] e é baseado no algoritmo RSA para assinaturas digitais.

No âmbito dos SVE as assinaturas cegas são realizadas pela entidade que controla o número de vezes que os eleitores votam. Assim, esta técnica permite a esses SVE assegurar a propriedade de democracia respeitando a privacidade.

2.1.3. *Zero-knowledge proofs*

Em algumas situações é necessário provar o conhecimento de alguma informação sem revelar nada a respeito da mesma. Estas provas são conhecidas por *zero-knowledge proofs*. Inicialmente introduzidas por Goldwasser, Micali e Rackoff, em 1985 [GMR85], as *zero-knowledge proofs* consistem em provas probabilísticas que permitem demonstrar que uma entidade conhece algo sem revelar nenhum conhecimento adicional sobre o que conhece. Através delas um verificador consegue alcançar um nível de garantia sobre o conhecimento que essa entidade detém.

Um exemplo que se encontra geralmente na literatura é o da gruta de Ali Baba. O objectivo é o Ali Baba provar ao amigo João que conhece o segredo que lhe permite mover a pedra que bloqueia a gruta (ver Figura 1).

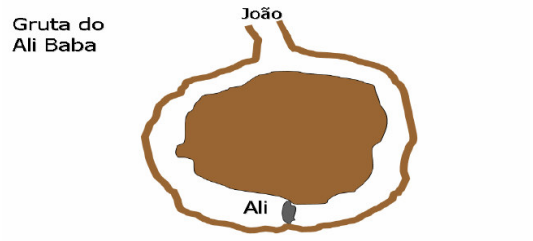

Figura 1 - Exemplo da gruta do Ali Baba

O Ali Baba entra na gruta sem o João ver e escolhe um caminho. Em seguida, o João (na entrada da gruta) pede ao Ali Baba para sair por um dos túneis, esquerdo ou direito. Se o Ali Baba sair por onde o João escolheu prova que conhece o segredo. Da

primeira vez a prova é parcial, isto é o João fica apenas com 50% de certeza que o Ali Baba conhece o segredo, mas podem repetir o processo até o João ficar convencido.

2.1.4. Cifra de múltipla decifra (*threshold decryption*)

A cifra de múltipla decifra [Schilcher04] consiste em dispor de uma chave pública para a cifra e, em vez de existir uma única chave privada, existem várias. Ou seja, é gerada uma chave (*t,n*) em que *t* é o número mínimo de partes da chave necessários para a decifra, e *n* é o número de partes da chave que são geradas. Assim uma mensagem cifrada pela chave pública só pode ser decifrada por quem possuir *t* partes da chave privada.

Normalmente este esquema é utilizado distribuindo as *n* partes da chave privada por *n* entidades distintas evitando assim que qualquer coligação de *t-1* entidades consiga quebrar a cifra e ter acesso à mensagem.

Esta técnica é utilizada como um método de verificação da correcção da contagem dos votos, visto que existem várias combinações possíveis de entidades para decifrar os votos. Introduz-se também uma melhor protecção contra fugas de resultados parciais pois o apuramento destes não depende só de uma entidade mas sim de *t* entidades.

2.1.5. Cifra com recifra aleatória

Considerando *M'* a cifra da mensagem *M*, a técnica de cifra com recifra aleatória consiste em alterar *M'* para *M"*, tal que *M'* seja diferente de *M"* mas verificando que a decifra de *M"* é *M*.

```
C[k1](M) = M' ---> F(M') = M" ---> D[k2](M") = M
```

Esta técnica é utilizada essencialmente em *mix-nets*, apresentadas na secção seguinte. Porém, também pode ser usada nos SVE para introduzir confusão no voto, alterando-o ao passar de entidade em entidade. Só o eleitor consegue provar a sua votação junto dessas entidades através de *zero-knowledge proofs* (ver Secção 2.1.3), que validam a correcção das alterações introduzidas por cada entidade. Normalmente, nos SVE que utilizam esta técnica o eleitor vê o voto no início da cadeia de entidades que fazem a recifra e no final escolhe o seu voto.

2.1.6. *Mix-nets*

A técnica de *mix-nets*, introduzida por David Chaum em 1981 [Chaum81], permite a um emissor *X* enviar uma mensagem anónima de forma segura a *Y*. Uma *mix-net* é constituída por vários servidores *mix*. Cada servidor *mix* tem o papel de esconder a correspondência entre a sua entrada e a sua saída. Segundo este critério é possível distinguir duas aproximações para as *mix-nets*: a de cifra e a de recifra.

Método de cifra

Cada servidor *mix* possui um par de chaves assimétricas. O processo é simples: o emissor cifra sucessivas vezes a mensagem que quer enviar com as chaves públicas dos servidores *mix*. Na cifra usa um factor aleatório para evitar uma correlação directa entre a entrada e saída de cada servidor *mix*. Depois envia essa mensagem cifrada para o primeiro servidor *mix*. Este efectua a primeira decifra, mistura as várias mensagens que tenha recebido e envia-as para o próximo servidor *mix*. O processo repete-se até a mensagem chegar completamente decifrada ao receptor.

Este processo tem a desvantagem de tornar imperativa a disponibilidade de todos os servidores *mix*, o que é um pressuposto muito forte em ambientes propícios a falhas. Outra desvantagem deste processo consiste no crescimento do boletim de voto cifrado.

Método de recifra

Nesta aproximação cada servidor *mix* recifra a mensagem e permuta-a com as restantes, entregando o resultado ao servidor *mix* seguinte. No final a mensagem recifrada é decifrada com um algoritmo *(t,N)-threshold* (ver Secção 2.1.4).

Esta aproximação providencia um método de utilização mais simples e, dado o tamanho constante das cifras, um menor tamanho de dados dos boletins de voto. Além disso, no método de cifra existe uma ordem fixa para a decifra das mensagens; neste método a decifra pode ser feita aleatoriamente e quaisquer *t* entidades, partilhando a chave privada, podem decifrar a mensagem no final.

Qualquer das duas aproximações (cifra ou recifra) requer a comprovação de que cada *mix* foi correctamente realizado. Isto é conseguido através de *zero-knowledge proofs*, seguindo o procedimento referido na Secção 2.1.3, embora sejam muito pesadas em termos de tamanho e consumo de recursos. Uma outra alternativa consiste na técnica RPC (*Randomized Partial Checking*) [JJR02], apresentada por Jacobsson, Juels and Rivest em 2002, que se adapta a qualquer aproximação das *mix-nets*. Esta determina que um servidor *mix* deve revelar a ligação entre algumas entradas seleccionadas aleatoriamente e as saídas correspondentes. Se a verificação entre as entradas e as saídas for positiva então o verificador obtém um nível de garantia acerca de todo o processo. Ao mesmo tempo, para esconder correlações directas entre a entrada do primeiro servidor *mix* e a saída do segundo servidor *mix*, os servidores *mix* devem ser agrupados dois a dois, e as saídas seleccionadas do primeiro servidor *mix* não devem ser as entradas seleccionadas do segundo servidor *mix*.

As *mix-nets* são úteis aos SVE para alcançarem a propriedade da privacidade, pois os interlocutores comunicam anonimamente. Normalmente este mecanismo é utilizado nos SVE para o depósito anónimo dos votos.

Os principais problemas ligados aos SVEs baseados em *mix-nets* são a complexidade das provas de correcção e a tolerância a falhas dos servidores *mix*.

2.1.7. Cifra homomórfica

Uma função f é dita homomórfica [BM02] em relação a "×" (multiplicação) e "+" (adição) se para todo X e Y temos

$$f(X) \times f(Y) = f(X + Y)$$

As funções de cifra homomórfica (e.g. a variação homomórfica da função de cifra ElGamal) podem ser usadas para determinar o resultado final das eleições sem revelar cada um dos votos, protegendo o anonimato dos eleitores. Cada eleitor *i* cifra o seu voto (*v[i]*) com a chave pública da eleição (1). No final do período de votação, as autoridades de voto multiplicam os votos cifrados (2), obtendo a cifra do resultado. Por fim, as autoridades efectuam a decifra dessa multiplicação em conjunto (ver Secção 2.1.4), obtendo o resultado final (3).

1. `C(v[1]), C(v[2]), ... , C(v[n])`
2. `C(v[1]) x C(v[2]) x ... x C(v[n]) = C(v[1]+v[2]+...+v[n])`
3. `D(C(v[1]+v[2]+...+v[n])) = v[1]+v[2]+...+v[n]`

O problema que surge na utilização deste tipo de cifra é a forte limitação ao formato dos boletins de voto, normalmente restringidos a votos Sim/Não. Isto porque para um voto apenas deve ser usado 1 bit dadas as limitações na flexibilidade da propriedade matemática que verificam. A utilização de mais bits em votações do tipo *1-de-n* traduz-se em elevados custos de computação. Uma alternativa possível é a transformação da votação *1-de-n* em *n* votações do tipo Sim/Não.

2.2. Máquinas virtuais Java

Nesta secção são introduzidos alguns conceitos tecnológicos inerentes aos ambientes e dispositivos móveis, nomeadamente aos telemóveis. Em primeiro lugar é realizada uma descrição da plataforma J2ME, incluindo a sua arquitectura e funcionalidades disponibilizadas. Posteriormente, é efectuada uma análise da constituição física e tecnológica dos *smart cards* assim como das capacidades oferecidas pelos mesmos.

2.2.1. Introdução ao J2ME

A plataforma J2ME [JSR68] foi concebida para dar suporte a dispositivos com reduzidas capacidades de memória e de processamento, tais como telemóveis, PDAs e TV *set-top boxes*, em que tanto a plataforma J2EE (*Java 2 Enterprise Edition*) como a J2SE (*Java 2 Standard Edition*) se tornavam num recurso demasiado pesado. A plataforma J2ME é constituída por um conjunto de APIs padrão definidas pelo programa *Java Community Process* [JCP]. A plataforma J2ME está hoje em dia presente em milhões de dispositivos (e.g. telemóveis, PDAs, dispositivos automóveis, ...) o que permite que uma aplicação escrita uma única vez possa ser executada numa larga gama de dispositivos. A Figura 2 apresenta uma vista geral da arquitectura da família de plataformas Java, incluindo o J2EE, J2SE, J2ME e *Java Cards*.

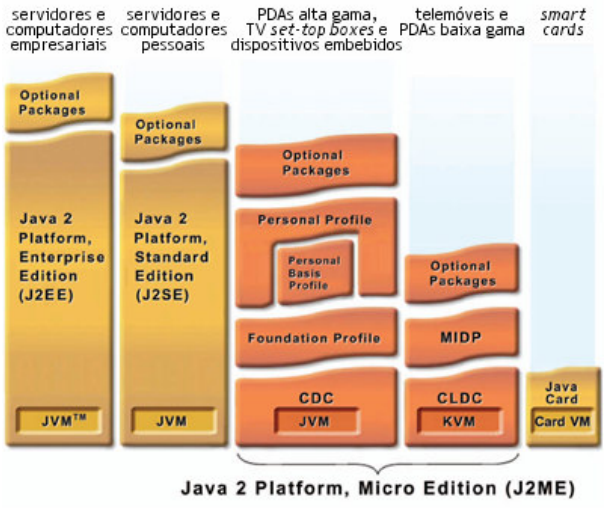

Figura 2 - Arquitectura do J2ME e relação com a restante família Java

A arquitectura do J2ME define os componentes *Configuration*, *Profile* e *Optional Packages*. Juntos estes componentes satisfazem muitos dos requisitos do mercado actual de dispositivos móveis, oferecendo o suporte para a construção de um ambiente Java.

Tendo em conta o facto do J2ME servir uma larga gama de dispositivos, existem duas possíveis arquitecturas da plataforma: uma dirigida a dispositivos com maior poder computacional; outra dirigida a dispositivos com poder computacional limitado. Cada arquitectura é definida por uma *Configuration* e um ou mais *Profiles*, podendo conter, eventualmente, *Optional Packages*.

A *Configuration* é a parte nuclear da plataforma J2ME. É composta por uma máquina virtual Java e um conjunto mínimo de bibliotecas para a execução de aplicações Java. Através dela é providenciada a funcionalidade base para uma gama específica de dispositivos que partilham características semelhantes, como tipos de conectividade e recursos de memória. Existem dois tipos de *Configurations*: a CDC (*Connected Device Configuration*), destinada aos dispositivos com maior poder computacional, e a CLDC (*Connected Limited Device Configuration*), para os dispositivos com poder computacional limitado.

Para providenciar um ambiente Java completo para o desenvolvimento de aplicações Java é necessário combinar a *Configuration* com um *Profile*. O *Profile* é composto por um conjunto de APIs de alto nível que definem a interface utilizador e o acesso a propriedades específicas dos dispositivos. Cada *Profile* é específico para uma determinada *Configuration*.

Adicionalmente, a plataforma J2ME pode ser estendida através de *Optional Packages*. Estas extensões à plataforma oferecem APIs padrão para áreas específicas de funcionalidade, como comunicação *bluetooth*, *web services*, segurança, *wireless messaging*, multimédia, entre outras. Desta maneira, cada fabricante de dispositivos tem à sua disposição um conjunto de *Optional Packages* que pode utilizar conforme as necessidades de cada dispositivo.

O espaço total ocupado pelo J2ME na memória do telemóvel depende dos componentes arquitecturais que forem instalados em cada dispositivo, sendo que os componentes *Configuration* e *Profile* são os obrigatórios da plataforma. Embora a *Configuration* CDC opere sobre a máquina virtual Java convencional, JVM (*Java*

Virtual Machine), a CLDC opera sobre a máquina virtual KVM (*KiloByte Virtual Machine*). Esta máquina virtual foi construída de raiz para dispositivos com limitações de memória e processamento, sendo personalizada para o sistema operativo de cada dispositivo.

Seguidamente, são apresentados com mais detalhes os componentes da arquitectura da plataforma J2ME utilizados nos telemóveis (CLDC e MIDP), dado o seu enquadramento no projecto MobileREVS.

CLDC

Conforme foi indicado anteriormente, a *Configuration* CLDC [Giguere02a] foi desenhada para dispositivos com limitações ao nível das capacidades de processamento e armazenamento (e.g. telemóveis e alguns PDAs). Estes dispositivos têm, tipicamente, CPUs de 16 ou 32 bits e, em 2002, tinham entre 128KB e 512KB de memória disponível para a instalação da plataforma J2ME e aplicações associadas. No entanto, as limitações em termos de memória têm vindo a diminuir ao longo do tempo com a introdução de cartões de memória no mercado com capacidades superiores, que já atingem hoje em dia 1GB.

O objectivo da criação da CLDC foi a definição de um padrão que permitisse às aplicações móveis abstraírem-se de operações nativas específicas dos sistemas operativos dos dispositivos, oferecendo-lhes portabilidade. Além disso, a máquina virtual KVM, usada pela CLDC, possui algumas restrições relativamente à convencional JVM (para efeitos de minimização do espaço ocupado em memória e consumo de energia dos dispositivos). A CLDC permite colmatar algumas das restrições da máquina virtual providenciando suporte a objectos de vírgula flutuante e a *weak references*, i.e. referências que não evitam a recolha dos objectos referenciados por parte do *garbage collector.*

Adicionalmente, a CLDC fornece um pequeno subconjunto de classes J2SE, para utilização por parte das aplicações, e um conjunto de interfaces genéricas de suporte a funções de entrada/saída de dados, a *Generic Connection Framework* (GCF). Note-se que nem a GCF nem o CLDC implementam nenhuma função de entrada/saída de dados, pois elas estariam inevitavelmente associadas a modelos de funcionamento que nem todos os dispositivos poderiam suportar. Por isso, e para manter a compatibilidade, os aspectos específicos da implementação devem ficar a cargo dos *Profiles* utilizados (ver secção seguinte) ou dos fabricantes dos dispositivos. Estes devem ser capazes de mapear um protocolo de ligação (definido pelo programador) num objecto de comunicação que implemente a API da GCF, para manipulação das funções de entrada/saída de dados.

Todos os aspectos relacionados com a interface utilizador não são definidos pelo CLDC.

Hoje em dia já existem duas versões do CLDC: a 1.0 e a 1.1. No âmbito do projecto MobileREVS qualquer uma delas é plausível para o desenvolvimento do sistema. A principal diferença da versão 1.1 para a versão 1.0 consiste na colmatação das restrições da máquina virtual, anteriormente referidas.

MIDP

O *Profile* MIDP [Giguere02b] é uma especificação imposta pela Sun para a utilização de Java em sistemas embebidos como os telemóveis. Esta oferece as funções requeridas e específicas dos dispositivos em causa (neste caso telemóveis), incluindo a

interface do utilizador, conectividade com a rede, armazenamento local de dados e gestão do ciclo de vida da aplicação J2ME.

Uma aplicação J2ME, designada de *MIDlet*, possui as características que permitem a gestão do seu ciclo de vida pelo MIDP. A sua instalação no telemóvel envolve a criação de uma *MIDlet suite*. Uma *MIDlet suite* é um arquivo Java que contém: (1) um ou mais *MIDlets*, juntamente com outras classes e recursos utilizados (imagens, sons, ...); (2) um descritor da aplicação, usado para verificar informações sobre o *MIDlet* antes deste ser instalado; e (3) um *manifest*, que descreve o conteúdo da *MIDlet suite* e as propriedades de cada *MIDlet* presente.

Actualmente, existem duas versões do MIDP, a 1.0 e a 2.0. Apesar de muitos dos telemóveis ainda possuírem a versão 1.0, a versão 2.0 já se está a estabelecer no mercado. As principais diferenças da versão 2.0 para a 1.0 [WNM], relevantes para o projecto MobileREVS, são as seguintes:

- Interface melhorada – novas adições e melhoramentos aos elementos de interface da versão anterior garantem melhor usabilidade, portabilidade das aplicações e extensibilidade;
- Maior conectividade – suporte para mais e actuais protocolos *standard* de comunicação, tais como HTTPS, *sockets* e datagramas, entre outros;
- Segurança ponto-a-ponto – modelo de segurança ponto-a-ponto robusto, para protecção da rede, das aplicações e dos dispositivos. A segurança advém do suporte do protocolo HTTPS, e dos *standards* associados como o SSL (*Secure Sockets Layer*) e o WTLS (*Wireless Transport Layer Security*) [SS], e da introdução do conceito de domínio de segurança. Um domínio de segurança previne acessos não autorizados de *MIDlet suites* (instaladas no dispositivo) a dados, aplicações e outros recursos da rede ou do dispositivo. Por omissão, as *MIDlet suites* não são de confiança, consequentemente estão confinadas a domínios de segurança muito restritos, sem acesso a funcionalidades privilegiadas. Para obterem esse acesso privilegiado as *MIDlet suites* necessitam de ser assinadas digitalmente por uma entidade de confiança.

2.2.2. SATSA

A SATSA (*Security and Trusted Services API*) é uma *Optional Package* que estende as funcionalidades de segurança para a plataforma J2ME [Ortiz05]. Esta extensão adiciona APIs criptográficas, de serviços de assinaturas digitais e de gestão de certificados. Adicionalmente, são definidos dois módulos – SATSA-APDU e SATSA-JCRMI – que, juntamente com extensões à *Generic Connection Framework* (ver secção anterior), providenciam uma API que permite às aplicações J2ME comunicar com elementos seguros, como os *smart cards* (ver Figura 3). No contexto do projecto, o *smart card* será o único elemento seguro referenciado nas secções seguintes.

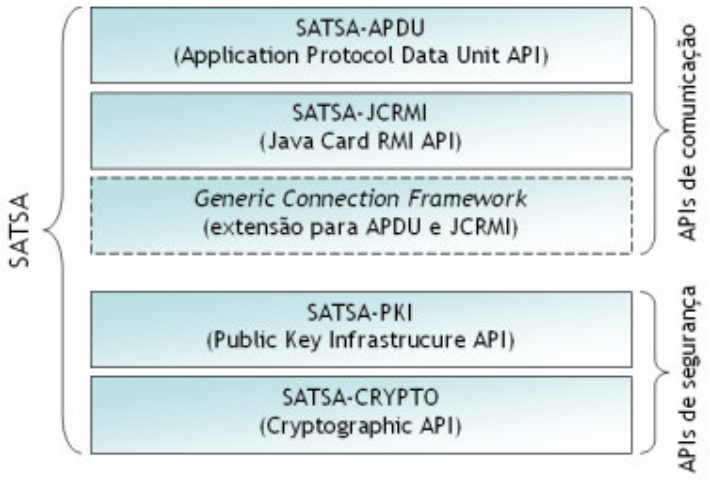

Figura 3 - APIs do SATSA

Módulos arquitecturais

A SATSA é constituída pelos seguintes quatro módulos, cuja funcionalidade se descreve:

- **SATSA-APDU**: oferece suporte para a comunicação com as aplicações que estão presentes no *smart card*;
- **SATSA-JCRMI**: permite que aplicações J2ME (i.e. *MIDlets*) efectuem invocações sobre objectos Java a executarem-se no *smart card*;
- **SATSA-PKI**: permite efectuar a geração de assinaturas digitais ao nível da aplicação e gestão de certificados;
- **SATSA-CRYPTO**: é uma biblioteca criptográfica genérica para J2ME.

Modelo de comunicação

O modelo de comunicação do SATSA relaciona-se com a utilização das APIs SATSA-APDU e SATSA-JCRMI. Ao estenderem o J2ME estas suportam a interacção das aplicações J2ME do telemóvel, os *MIDlets*, com aplicações residentes nos *smart cards*, as *applets* (ver Figura 4). Através desta interacção é possível realizar diversos serviços, tais como o armazenamento seguro de dados.

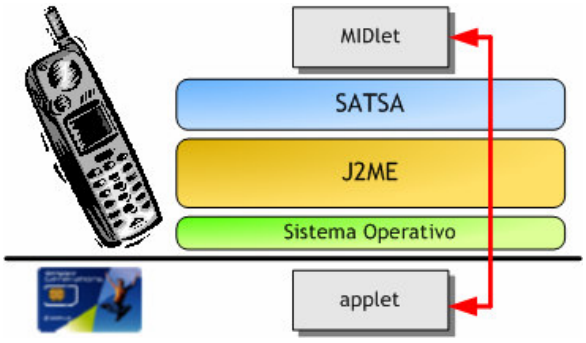

Figura 4 - Interacção dos *MIDlets* com os *applets*

A comunicação entre os *MIDlets* e os *applets* é baseada num modelo síncrono de pedido/resposta (ver Figura 5), onde o *MIDlet* toma o papel de cliente na interacção e o *applet* toma o papel de servidor. Esta comunicação é realizada através do envio de APDUs (*Application Protocol Data Units*), que correspondem a tramas de dados em bits. Na comunicação com *Java Cards* este modelo é simplificado através do módulo SATSA-JCRMI, providenciando um modelo orientado a objectos que abstrai a troca dessas tramas.

Figura 5 - Modelo de comunicação do SATSA

Controlo de acessos

Os módulos SATSA-APDU e SATSA-JCRMI definem um modelo de controlo de acessos [ISCSE] entre os *MIDlets* e os *applets*. Este modelo foi desenhado com o propósito de: (1) proteger os *applets* e os dados do *smart card* do acesso mal intencionado dos *MIDlets*; (2) permitir, temporariamente, o acesso de um *MIDlet* a um objecto do *smart card* em regime de exclusão mútua; e (3) proteger os PINs do *smart card* de uso impróprio por parte dos *MIDlets*.

A biblioteca SATSA permite separar a lógica de negócio, o *MIDlet*, da lógica associada à segurança do mesmo, contida no *smart card*. Esta separação oferece a vantagem da utilização do *smart card* como recurso de segurança para o armazenamento de dados sensíveis. No entanto, tal facto não inviabiliza a necessidade de proteger o acesso a esses dados; *MIDlets* mal intencionados podem tentar aceder a informações privadas armazenadas no cartão. Como tal, é necessário garantir que o acesso é providenciado apenas ao *MIDlet* detentor dos dados armazenados no *smart card*.

2.2.3. *Smart cards*

A utilização do *smart card* requer a presença, no cartão, de *applets* conhecidos pelos *MIDlets* que o acedem. No entanto, não foi possível determinar que *applets* estão disponíveis para acesso nos cartões utilizados actualmente.

Seguidamente, os *smart cards* são apresentados com mais pormenor. Num primeiro passo é feita uma introdução à sua arquitectura física. Posteriormente, são demonstradas algumas das utilizações dos *smart cards* nos telemóveis.

Arquitectura

Um *smart card* [SCE] é um pequeno circuito micro-controlador integrado com capacidade de processamento e armazenamento de dados. É fisicamente constituído pelos seguintes componentes (ver Figura 6):

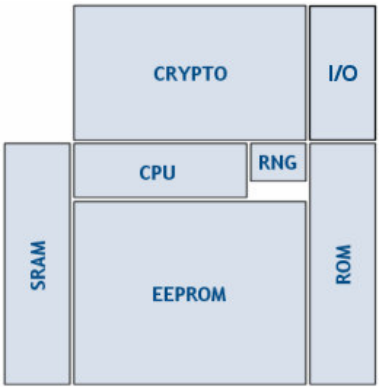

Figura 6 - Arquitectura de um *smart card*

- **CPU (Control Processing Unit)**: o CPU é uma unidade de processamento capaz de executar um sistema operativo (Symbian OS, Windows for *smart cards* OS, MPCOS-EMV OS, ...). Hoje em dia são muito comuns os CPUs de 8 bits (e.g. 8051, 6805, HC05, AVR, ...). Para maiores capacidades de processamento e armazenamento de dados existem CPUs de 16 e 32 bits;
- **SRAM (Static Random Access Memory)**: área de memória volátil onde o sistema operativo e as aplicações armazenam dados temporários durante a sua execução. A sua capacidade mais comum varia entre os 256 Bytes e os 2 KBytes. Para maiores capacidades de computação são usados *smart cards* de 4 KBytes;
- **ROM (Read-Only Memory)**: área de memória permanente, apenas de leitura. Contém as camadas de código de baixo nível do sistema operativo juntamente com as instruções micro-programadas (instruções de baixo nível do processador). A sua capacidade varia normalmente entre os 8 KBytes e os 64 KBytes;
- **EEPROM (Electrically Erasable and Programmable ROM)**: área de memória não volátil que pode ser apagada e escrita. É usada para armazenamento de dados, partes do sistema operativo sujeitos a modificações e *software* utilizador, tal como Java *applets*. Actualmente, a capacidade da EEPROM varia entre os 8 KBytes e os 32 KBytes, embora também já existam EEPROM de 64 e 128 KBytes. Este tipo de tecnologia está a ser ultrapassada pela utilização de EEPROMs em Flash, descritas seguidamente, devido às maiores velocidade de escrita e retenção de dados (durabilidade);
- **Flash-EEPROM**: área de memória com as mesmas características e funcionalidades da EEPROM. Difere apenas nos aspectos tecnológicos usados para o armazenamento de dados. É usada em alternativa à EEPROM;
- **I/O (Input/Output)**: linha única de escrita e leitura de dados entre o cartão e o exterior. Esta linha não é diferenciada (*half-duplex*), ou seja, a troca de dados tem de ser realizada num sentido de cada vez;
- **CRYPTO (Cryptographic Processor)**: co-processador opcional dedicado a tarefas específicas de criptografia, especialmente as que envolvem maior poder computacional como a cifra assimétrica (e.g. RSA). Tem capacidade para geração de pares de chaves RSA de 1024 bits, operações de *hashing* como SHA e MD5, e operações de cifra simétrica como o 3DES;

- **RNG (Random Number Generator)**: gerador de números aleatórios, especialmente usado para a geração de chaves criptográficas.

Cartões (U)SIM

Muitas vezes confundido, o termo SIM (*Subscriber Identity Module*) não se refere ao cartão em si. O *smart card* físico, usado nos telemóveis, é designado de UICC (*Universal Integrated Circuit Card*). O módulo SIM é uma aplicação que corre nos *smart cards* UICC e que permite a identificação do cliente numa rede telefónica móvel GSM. O USIM (Universal SIM) é o equivalente ao módulo SIM numa rede telefónica móvel UMTS (com capacidades de 3ª geração). Neste relatório foi adoptada a designação de cartão (U)SIM para identificar os cartões UICC presentes nos telemóveis de hoje em dia.

Cada cartão (U)SIM é único, com um número de série associado. A sua utilização está a crescer cada vez mais hoje em dia como um elemento de identificação dos utilizadores de uma rede telefónica móvel. Permite o acesso aos serviços telefónicos assim como a outros serviços electrónicos oferecidos pela rede. Além disso é um local seguro para o armazenamento de informação sensível, tal como (1) a chave que identifica a operadora móvel, (2) informação de estado da rede móvel, e (3) informação pessoal (e.g. lista de contactos ou mensagens escritas). A sua compatibilidade com diferentes tipos de hardware (telemóveis, PDAs, computadores pessoais, TV digital, ...) é outro aspecto pelo qual a aposta na criação de uma Identidade Electrónica [WEI], com base no cartão (U)SIM, poderá ser bastante proveitosa para a sociedade (esta medida já foi adoptada na Finlândia em Junho de 2005 [ECPR]).

Principais fabricantes

Entre os principais fabricantes [SCM] de *smart cards* encontram-se as seguintes empresas:

- Axalto - http://www.axalto.com
- Gemplus - http://www.gemplus.com
- Giesecke & Devrient - http://www.gdm.de
- Oberthur Card Systems - http://www.oberthurcs.com

Porém, após uma investigação aprofundada, não foi possível identificar quais os cartões usados pelas operadoras de telecomunicações em Portugal. Apenas foi constatado que pelo menos alguns dos novos cartões, que acompanham os novos telemóveis de 3ª geração, são fornecidos pela Gemplus (e.g. cartões 3G da TMN). Para além disso, os fabricantes dos cartões apenas disponibilizam ao público informações muito gerais acerca das funcionalidades oferecidas pelos mesmos, pelo que estes pormenores não forma explorados para efeitos do desenvolvimento do projecto.

2.3. Segurança nos telemóveis

A segurança é um dos principais requisitos num sistema de votação electrónica. Sendo o MobileREVS constituído por um *MIDlet* e um conjunto de servidores remotos há que garantir tanto a segurança do próprio *MIDlet* como da comunicação efectuada com os servidores. Para implementar a segurança do sistema foram analisadas diversas

opções, cada qual com as suas vantagens e desvantagens, e determinado um veredicto final acerca das mesmas.

2.3.1. Considerações

Nesta secção são apresentadas algumas considerações sobre aspectos de segurança ligados com o projecto MobileREVS.

Autenticação utilizador/palavra-passe

Hoje em dia, a forma mais comum de autenticação de um utilizador é através da utilização de uma palavra-passe. Mas, apesar de frequentemente utilizadas, as palavras-passe não são muito seguras. Esta fraca segurança que lhes é associada advém principalmente de factores humanos [MY01]:

- Muitas das palavras-passe escolhidas pelos utilizadores são consideradas palavras-passe fracas, i.e. podem ser descobertas através de ataques por força bruta;
- Os utilizadores necessitam de conhecer muitas palavras-passe para diferentes sistemas. O seu esquecimento é a consequência mais frequente, inviabilizando-os da utilização do sistema em causa. Ou então, para evitar isso, os utilizadores
    - apontam-nas (em papel ou em formato digital), ficando a sua segurança comprometida por quem conseguir acesso às mesmas;
    - utilizam sempre a mesma em vários sistemas, tornando-a assim mais fraca aos ataques.

Nestas condições, as palavras-passe não oferecem nenhuma prova sobre quem de facto a está a utilizar. São muitas e problemáticas as fragilidades denotadas e que sujeitam a personificação por parte de outros utilizadores.

Certificados digitais

Por si só, a criptografia de chave pública não permite associar uma entidade com uma determinada chave pública. Teoricamente, um atacante poderia apresentar uma chave pública e reivindicar que a mesma pertence a outra entidade. Na falta de mecanismos para associar chaves com entidades qualquer outro participante da transacção não teria nenhuma garantia sobre a validade dessa associação. Por esta razão, é necessária uma forma segura de efectuar a associação das chaves públicas com as entidades respectivas. Para combater este problema são introduzidos os certificados digitais e uma terceira entidade: a Autoridade de Certificação. A Autoridade de Certificação é uma entidade de confiança que verifica a identidade do dono do certificado antes de o emitir.

O certificado digital contém a chave pública de uma entidade juntamente com outra informação de identificação, incluindo o nome da entidade, um número de série e uma data de expiração (altura a partir da qual será revogado, tendo que ser renovado). Adicionalmente, o certificado digital contém o nome e a assinatura digital da Autoridade de Certificação que emitiu o certificado. Assim, os certificados digitais são

como passaportes internacionais pois permitem a interoperabilidade entre diversos sistemas através do *standard* X.509 [WX].

Os certificados digitais, se estiverem assinados por uma Autoridade de Certificação de confiança, permitem garantir de facto a ligação entre uma chave e a entidade a que pertence (ver Secção 2.1.1).

De modo a garantir a autenticidade, integridade e não-repúdio da informação trocada entre duas entidades, qualquer mensagem enviada deve ser assinada pela entidade que a envia. Para a geração desta assinatura é criado um resumo da mensagem a enviar que é cifrado com a chave privada da entidade que assina. O ponto fraco dos certificados digitais advém de serem apenas baseados na segurança da chave privada da entidade que assina. Qualquer atacante com acesso à chave privada consegue enviar mensagens assinadas com outra assinatura que não a sua.

Desta maneira, e apesar dos certificados digitais associarem uma identidade a uma chave pública, o certificado digital por si só não permite garantir que a entidade que apresenta o certificado é quem de facto diz ser. Consequentemente, a protecção das chaves privadas é o aspecto de maior importância na utilização de certificados digitais, já que a sua descoberta abrirá as portas aos atacantes para a personificação.

No âmbito do sistema MobileREVS os certificados são utilizados para autenticar os servidores perante os eleitores, pelo que a segurança das comunicações estará dependente da protecção das chaves privadas dos servidores.

2.3.2. Segurança associada aos *smart cards*

Os *smart cards* oferecem mais segurança que os telemóveis no processamento e armazenamento de dados. Com efeito, no *smart card* pode ser armazenada informação sensível e confidencial como chaves privadas, números de contas, palavras-passe ou informações pessoais. É, também, um local seguro para a execução de algumas operações, evitando que as mesmas sejam expostas na rede. Além disso, os *smart cards* apresentam funcionalidades de resistência a ataques, descritas na secção que se segue, que tornam a relação custo/benefício pouco proveitosa para o atacante.

Os *smart cards* têm grandes vantagens: além de portáteis e fáceis de usar, são capazes de armazenar dados e executar algoritmos criptográficos no seu circuito interno. Isto significa que os segredos do utilizador, sejam chaves ou o código PIN, nunca necessitam de ser transmitidos para fora do cartão, oferecendo maior segurança a todo o sistema em que o cartão está envolvido.

A protecção extra de segurança oferecida pelos *smart cards* não se deve apenas ao conhecimento de um PIN de acesso, que invalida o cartão ao fim de um número fixo de tentativas de introdução. Também é necessário ter a posse física do cartão para se conseguir ter acesso aos dados sensíveis que nele se encontram alojados. Esta protecção conjunta oferece maior garantia de que nos certificados digitais a chave privada é de facto usada pelo respectivo detentor do certificado.

É importante referir outras duas vantagens: (1) os *smart cards* são únicos e muito difíceis de replicar em caso de extravio; e (2) os *smart cards* são dotados de técnicas de *tamper-resistance*. Estas técnicas permitem proteger o código e dados do cartão de leituras e modificações através de meios exteriores. Apenas o *software* embebido no cartão, dotado das medidas de segurança adequadas, tem os privilégios de acesso a estes dados.

Funcionalidades criptográficas

Os *smart cards*, para além das vantagens referidas de protecção e autenticação, possuem um co-processador específico para a realização de operações criptográficas: o módulo *Crypto*. Este módulo, conforme foi referido na Secção 2.2.3, realiza operações de cifra simétrica e assimétrica, das quais se destacam respectivamente os algoritmos 3DES e RSA. Comparativamente, os tempos de execução de uma operação de cifra de um bloco de dados num *smart card* são aproximadamente os seguintes [PS]:

- O algoritmo 3DES, com uma chave de 112 bits, demora 110 μs;
- O algoritmo RSA, utilizando uma chave de 1024 bits, demora 400 ms;

Os tempos apresentados dizem respeito ao modelo SmartXA P16WM064 da Philips, um *smart card* com um CPU de 16 bits e um co-processador criptográfico de 32 bits. O bloco de dados do 3DES tem 64 bits e o de RSA tem 1024 bits.

2.3.3. Mecanismos de segurança

O sistema MobileREVS não é um *MIDlet* isolado. A sua arquitectura é baseada num modelo distribuído composto pelo Módulo Eleitor, que comunica através de uma rede aberta com outros servidores (ver Secção 4.3). De modo a poder garantir a segurança dessa comunicação, e ao mesmo tempo assegurar os requisitos do sistema, deve-se recorrer a técnicas criptográficas que forneçam computacionalmente essa segurança.

Sendo a plataforma J2ME dotada de poucas funcionalidades criptográficas é necessário recorrer a mecanismos adicionais para desempenhar este papel. Para o efeito, existem várias soluções possíveis cuja utilização é plausível no sistema MobileREVS:

- Funções criptográficas do módulo *Crypto* dos *smart cards*;
- APIs de certificados digitais oferecidas pelo MIDP;
- *Optional Package* SATSA do J2ME;
- *Light Edition* da biblioteca criptográfica da *Legion of the Bouncy Castle*.

Seguidamente, é feita uma descrição e análise dessas soluções e, no final, é apresentada uma avaliação das mesmas.

Módulo *Crypto* dos *smart cards*

O módulo *Crypto* é um módulo interno ao cartão dos telemóveis que oferece funcionalidades criptográficas e de geração de números aleatórios (ver Secção 2.2.3). A sua utilização reduziria significativamente o espaço total ocupado pelo *MIDlet*, já que não seria necessário recorrer a bibliotecas externas, para pelo menos algumas funções criptográficas. Para o utilizar o *MIDlet* tem de recorrer ao envio de tramas de bits (APDUs) segundo protocolos específicos.

Apesar da vantagem em termos de segurança é necessário ter em conta que a utilização deste módulo pode aumentar significativamente o tempo de execução das operações criptográficas, dado o recurso a comunicações entre o *MIDlet* e o cartão. No entanto, não encontrámos nenhum artigo que nos clarificasse em relação a este aspecto. Deste modo, esta opção não foi considerada para a realização das operações criptográficas necessárias.

MIDP

O MIDP (ver Secção 2.2.1), apesar de não disponibilizar funções criptográficas, está acompanhado (na versão 2.0) de uma interface Java para lidar com certificados digitais. Esta disponibiliza métodos que permitem obter os dados dos certificados, tais como a Autoridade de Certificação envolvida, a entidade a que o certificado está associado, o número de série, o algoritmo com o qual foi assinado, e a data de validade.

Apesar do MIDP permitir a utilização de comunicação segura, não oferece nenhuma função criptográfica que permita assegurar a cifra de dados no protocolo do MobileREVS (ver Secção 4.4). Como tal, a sua utilização deve ser sempre feita em conjunto com outra (ou outras) das soluções aqui apresentadas.

SATSA

A SATSA (ver Secção 2.2.2) é uma *Optional Package* que estende a plataforma J2ME. Esta é composta por vários módulos opcionais, dispondo de dois deles para lidar com problemas de segurança:

1. O módulo SATSA-PKI providencia:
    - um pacote para gestão de certificados, que inclui a criação de CSR (*certificate signing request*), e a adição e remoção de certificados assinados num elemento seguro como um *smart card*. Estes certificados podem posteriormente ser usados para gerar assinaturas digitais;
    - um pacote de serviços de assinatura para mensagens cifradas.
2. O módulo SATSA-CRYPTO é um subconjunto da API criptográfica do J2SE. Este módulo inclui pacotes que providenciam o suporte para:
    - obter resumos de mensagens;
    - gerar e verificar assinaturas digitais;
    - cifrar e decifrar dados.

Nos telemóveis, o módulo SATSA-PKI usufrui dos *smart cards* como elementos seguros de armazenamento dos certificados do *MIDlet*. No entanto, não oferece suporte para guardar certificados de entidades terceiras. Estes certificados têm de ser armazenados externamente, na memória do telemóvel (e.g. em *record stores* providenciados pelo MIDP).

Em relação ao módulo SATSA-CRYPTO da API SATSA, este não recorre ao uso do módulo *Crypto* dos *smart cards* para efectuar operações criptográficas. Todas as operações criptográficas são realizadas ao nível da aplicação, i.e. no CPU do telemóvel.

Bouncy Castle (*Light Edition*)

Esta biblioteca é uma implementação Java dos mais variados algoritmos criptográficos, existindo uma versão destinada a dispositivos móveis: a *Light Edition* [LBC]. Embora disponha de todas as funções de segurança necessárias para implementar o protocolo do MobileREVS, esta biblioteca não usufrui das capacidades de armazenamento seguro e processamento criptográfico dos *smart cards*, sendo toda a funcionalidade realizada ao nível da aplicação.

Os dois últimos mecanismos apresentados – a *Optional Package* SATSA e a biblioteca Bouncy Castle (*Light Edition*) – foram alvo de uma avaliação detalhada, apresentada com mais pormenor no Anexo A. A avaliação não teve em consideração o MIDP visto não se apresentar como uma solução única para o cumprimento do protocolo do MobileREVS. O módulo *Crypto* dos *smart cards* também foi excluído da avaliação dada a inexistência de informação sobre o modo de utilização pelas aplicações.

As principais conclusões obtidas da avaliação realizada são as que se seguem:

- A biblioteca Bouncy Castle (*Light Edition*) oferece todos os algoritmos de segurança requeridos para o MobileREVS. Ao ser incluída na *MIDlet suite* tem a vantagem de garantir portabilidade à aplicação, visto não estar dependente da implementação do J2ME. Porém, tem como principal desvantagem o aumento significativo do tamanho da aplicação;
- A *Optional Package* SATSA permite a comunicação da aplicação com o cartão, potenciando a utilização das funções criptográficas do mesmo e, como tal, aumentando a segurança do sistema. Como vem instalada com a implementação do J2ME não contribui para o aumento do tamanho da aplicação. Por outro lado, a sua inclusão na máquina virtual Java dos telemóveis não é usual, havendo actualmente muito poucos dispositivos com esta *Optional Package*. Outra das desvantagens é a limitação existente ao nível da utilização de algumas funções criptográficas, que não permitem o cumprimento da totalidade do protocolo do MobileREVS.

Pesadas as vantagens e desvantagens de cada mecanismo, optou-se por utilizar a biblioteca Bouncy Castle (*Light Edition*) dada a sua independência da implementação do J2ME presente em cada telemóvel.

3. Trabalho relacionado

Este capítulo apresenta algum do trabalho desenvolvido, directamente relacionado com o MobileREVS. Em primeiro lugar é dado destaque ao REVS, o SVE de base sobre o qual foi desenvolvido o MobileREVS. Posteriormente serão apresentadas diversas experiências internacionais na área da votação electrónica através de dispositivos móveis.

3.1. REVS

O principal objectivo do REVS, "*Robust Electronic Voting System*" [JZF03], é oferecer as principais propriedades dos sistemas electrónicos de votação (ver Secção 1.1.2) e providenciar um sistema robusto em três aspectos fundamentais:

1. **Disponibilidade**, providenciando um sistema sem um ponto de falha único, e com um protocolo de voto que suporte falhas nos servidores e na comunicação;
2. **Capacidade de continuação diferida**, permitindo que o processo de votação seja interrompido, intencionalmente ou não, e retomado mais tarde, durante o período de votação;
3. **Resistência ao conluio**, não deixando um servidor, ou um subconjunto deles dependendo de um nível configurável de conluio, interferir com a eleição.

Para o REVS foi escolhido um protocolo de votação baseado em assinaturas cegas. Consequentemente, tal como todos os sistemas baseados em assinaturas cegas, o REVS tem algumas vantagens intrínsecas interessantes das quais se destacam: simplicidade, custos baixos (é eficiente tanto computacionalmente como na rede) e independência do tipo de boletins de voto. Estes são aspectos importantes que facilitam a utilização do REVS no suporte a eleições de larga escala.

3.1.1. Arquitectura

No REVS existem quatro tipos de servidores (ver Figura 7): Distribuidores de Boletins, Administradores, Anonimizadores e Contadores. Existem também: um Módulo Eleitor, que é usado pelos eleitores para suportar as suas votações; e um módulo para preparar a eleição, o Comissário. A função de cada um dos módulos é a seguinte:

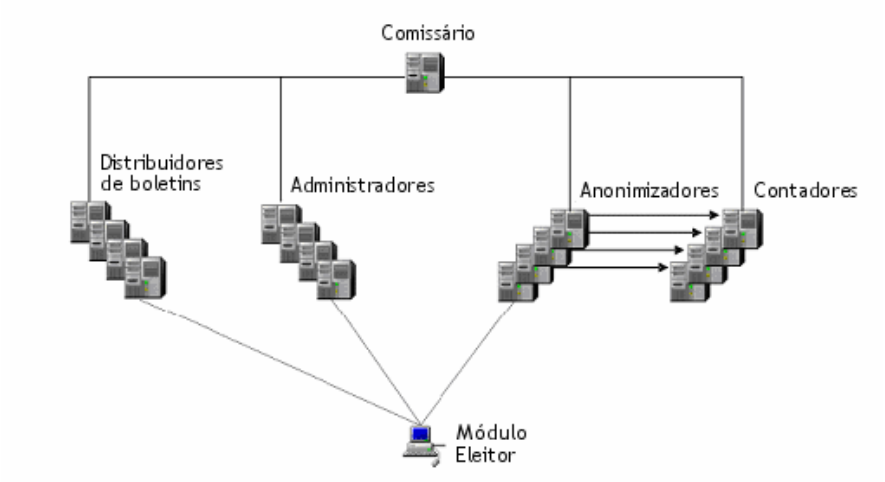

Figura 7 - Arquitectura do sistema REVS

- **Comissário** – é o módulo usado para preparar a eleição: registar os eleitores, definir as configurações operacionais (par de chaves da eleição, boletim, endereços e chaves públicas dos servidores, número de assinaturas requeridas, etc.). O Comissário assina todos os dados relativos à eleição de modo a que qualquer pessoa possa verificar a sua autenticidade;
- **Distribuidor de Boletins** – é responsável pela distribuição dos dados definidos pelo Comissário da eleição: boletins e configurações operacionais. Toda a informação distribuída por este servidor tem de ser assinada pelo Comissário, entidade de confiança dos eleitores. Como tal, podem haver vários Distribuidores de Boletins, aumentando assim a eficiência e providenciando tolerância a falhas. Deste modo é aumentada a robustez do processo de distribuição de boletins;
- **Administradores** – são as entidades responsáveis pela validação dos votos. Um voto apenas é aceite na contagem final caso tenha a assinatura de *n* Administradores diferentes, em que *n* representa uma maioria dos Administradores. O eleitor utiliza palavras-passe diferentes para se autenticar perante cada Administrador, não permitindo assim que estes personifiquem o eleitor;
- **Anonimizador** – este módulo tem a responsabilidade de providenciar o anonimato, não permitindo que o Contador associe um boletim de voto a um eleitor. O Anonimizador baralha e retém temporariamente os votos, protegendo a privacidade dos eleitores;
- **Contador** – verifica a validade de cada boletim certificando-se que estão presentes todas as assinaturas requeridas dos Administradores. Os eleitores enviam os seus votos para os Contadores, através dos Anonimizadores, cifrados com a chave pública da eleição, prevenindo que estes possam interpretar os votos durante o processo eleitoral;
- **Módulo Eleitor** – é o módulo usado pelo eleitor para participar na eleição, executando todas as interacções necessárias com os servidores.

No REVS os Distribuidores de Boletins, os Anonimizadores e os Contadores podem ser replicados. A replicação é útil para evitar pontos únicos de falha que comprometam a realização do protocolo. Os Administradores requerem especial atenção

pois cada boletim de voto deve ser assinado por uma maioria destes de modo a que o boletim seja considerado válido.

3.1.2. Protocolo

O protocolo REVS está dividido em três passos: distribuição de boletins, assinatura de boletins e submissão de boletins (ver Figura 8).

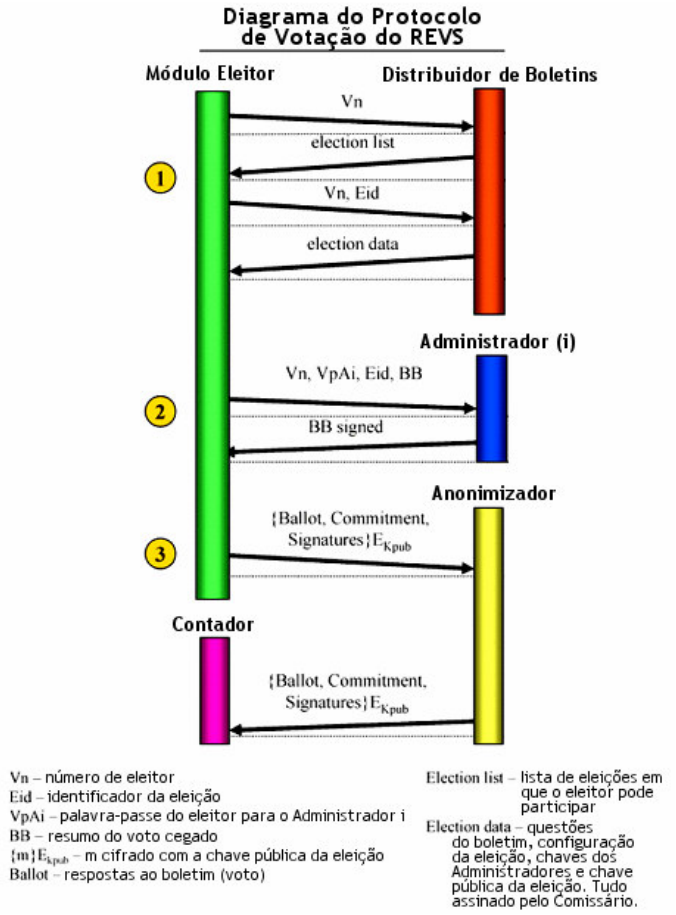

Figura 8 - Protocolo de votação do sistema REVS

- **Distribuição de boletins (1)** – o eleitor contacta um Distribuidor de Boletins e providencia o seu número de eleitor, *Vn*, para receber uma lista de eleições que estão a decorrer e nas quais ele pode participar, *election list*. Seguidamente o eleitor requer um boletim (*election data*) ao Distribuidor de Boletins, indicando-lhe a eleição escolhida (*Eid*);
- **Assinatura de boletins (2)** – após o eleitor expressar o seu voto, é criado um resumo do mesmo ao qual é aplicado um factor de cegamento (*BB*). Depois envia-o para mais de metade dos Administradores, juntamente com as palavras-passe respectivas (*VpAi*), de modo a obter as suas assinaturas (*BB signed*). Cada Administrador após receber o pedido do eleitor verifica se ainda não assinou um voto para o mesmo eleitor. Em caso negativo, este assina-o e devolve-o ao eleitor, guardando-o; em caso positivo, apenas devolve ao eleitor o voto previamente assinado;

- **Submissão de boletins (3)** – neste passo o eleitor constrói um pacote do qual faz parte o seu voto (*Ballot*), as assinaturas (*Signatures*) e um conjunto de bits que identificam a submissão do voto por parte de um eleitor (*Commitment*). Nesta altura o Módulo Eleitor pode guardar esta informação em memória segura melhorando assim o desempenho do sistema. Seguidamente, o pacote é cifrado com uma chave de sessão juntamente com a chave da eleição (E_{Kpub}) e submetido através dos Anonimizadores. Cada eleitor pode submeter o seu voto as vezes que achar necessário até se sentir confiante que o voto chegou ao seu destino.

Tal como foi referido, cada eleitor pode submeter o seu voto as vezes que pretender, para qualquer um dos Contadores. Portanto, diferentes Contadores podem ter diferentes conjuntos de votos no final da eleição, e esses conjuntos podem conter votos repetidos. Para solucionar este problema é escolhido um Contador Mestre, que possui acesso aos restantes Contadores. O Contador Mestre obtém a contagem final depois de juntar os votos de todos os Contadores, eliminando os repetidos.

No final da eleição todos os Contadores publicam os pacotes recebidos e os Administradores os boletins assinados. Deste modo, cada eleitor pode então verificar se o seu voto foi tido em conta para a eleição. Caso isso não tenha acontecido, este pode reclamar apresentando anonimamente o pacote previamente guardado no acto da submissão de boletins.

3.2. Outros trabalhos

Nesta secção são apresentados outros trabalhos da área da votação electrónica cujo desenvolvimento se baseou, exclusivamente ou não, em dispositivos móveis. São também apresentados trabalhos ligados a tecnologias associadas, como os *smart cards*, visto poderem ser utilizadas em conjunto com dispositivos móveis.

3.2.1. Utilização de *smart cards* na Estónia

Durante o ano de 2001 debateu-se a temática do voto não presencial na Estónia, que culminou com o estabelecimento das condições legais para a sua implementação. No Verão de 2003 o Comité Eleitoral Nacional coordenou o desenvolvimento do projecto de voto actual, tendo a Cybernetica, Lda como parceiro na produção do SVE [ENEC].

Nesta altura a utilização de cartões digitais de identificação pela população já constituía uma prática regular e cerca de 41% das casas do país possuíam pelo menos um computador com acesso à Internet. Desta forma, o sistema foi desenvolvido para a Internet e assenta na utilização de *smart cards* (os cartões digitais de identificação) e assinaturas electrónicas para a identificação dos eleitores. O sistema oferece dois meios para prevenir a compra do voto e a coacção da votação: (1) é possível efectuar várias votações durante o período da eleição, contando como válido o último voto; (2) a votação pelos meios tradicionais é prioritária, pelo que desta forma caso exista um voto electrónico para essa pessoa ele será eliminado da contagem.

3.2.2. Experiências em Espanha

A Espanha tem assistido a diversas experiências-piloto de voto electrónico. Em 1995 foi testado um sistema de cartões de banda magnética nas eleições para o Parlamento da Catalunha. Em 1997, as eleições para o Parlamento regional da Galiza contaram com um piloto de voto electrónico presencial.

A 16 de Novembro de 2003 [GC], novamente nas eleições para o Parlamento da Catalunha, foram utilizados três tipos de sistemas diferentes: (1) uma urna electrónica, onde o processo de votação é muito semelhante ao processo habitual contando, porém, com a verificação e contagem do voto de modo electrónico; (2) um sistema de ecrã táctil, com autenticação baseada em *smart cards*; e (3) um sistema de votação para a Internet, contando com a participação de mais de 730 eleitores.

Em Março de 2004 foram conduzidas experiências de voto electrónico em vários municípios espanhóis: 400 eleitores, em Jung, e 274, em Zamora e Lugo, testaram sistemas de votação através da Internet; ainda em Jung, 197 eleitores experimentaram um sistema de votação através de telemóveis, via SMS.

3.2.3. CyberVote

A Comissão Europeia lançou, em 2000, o projecto CyberVote [CV] com o objectivo de demonstrar a possibilidade de execução de "*eleições online totalmente verificáveis, garantindo a privacidade absoluta dos votos e usando terminais de Internet fixos e móveis*". O projecto envolveu parceiros da indústria (a francesa EADS Matra Systèmes & Information, o Centro de Investigação da Nokia na Finlândia e a British Telecommunications do Reino Unido), universidades (a KU Leuven Research & Development da Bélgica e a Technische Universiteit Eindhoven da Holanda) e utilizadores potenciais.

O projecto compreendeu o desenvolvimento de um protocolo de votação e de um sistema de voto electrónico, para PC e clientes móveis (aplicação em Java). Posteriormente foi testado em diferentes eleições (França, Alemanha e Suécia) entre 2002 e 2003, tendo terminado oficialmente a Julho de 2003.

3.2.4. Voto não presencial no Reino Unido

Em Maio de 2002, 30 autoridades locais levaram a cabo um total de 36 procedimentos inovadores de voto, incluindo o voto não presencial através do telefone, da Internet ou por SMS através de telemóveis. Em Maio de 2003, tiveram lugar mais experiências-piloto para testar métodos inovadores de votação e contagem em 59 autoridades locais inglesas. Mais de 14% do eleitorado inglês pôde ter acesso a estas experiências que englobaram 17 esquemas de votação, incluindo, pela primeira vez, a televisão digital interactiva. Mais informações em [ERS].

3.2.5. Denominadores comuns

Alguns dos denominadores comuns das experiências anteriormente referidas são:

- Muitos dos esforços desenvolvidos continuam a centrar-se nas plataformas presenciais. Porém, a redução das taxas de abstenção é um factor que não é potenciado pelas mesmas;
- A maioria das experiências envolvendo a utilização de dispositivos móveis recai na utilização de SMS e das interfaces de texto associadas. No entanto, este canal continua a mostrar-se de utilização pouco amigável em eleições, sendo utilizado apenas pelas camadas mais jovens;
- A taxa de abstenção não tem sido significativamente reduzida com a introdução dos novos sistemas de votação, principalmente por falta de credibilidade e dificuldades na utilização dos mesmos;
- Os dispositivos móveis têm sofrido um rápido desenvolvimento tecnológico e tem-se notado uma convergência crescente da tecnologia neste sector, oferecendo melhores promessas ao voto electrónico num futuro próximo.

Todos estes denominadores estão por detrás da motivação do projecto MobileREVS.

A utilização de dispositivos móveis em eleições ainda se encontra numa fase de avaliações experimentais, especialmente ao nível de factores sociais e éticos. A sua adopção não constitui um foco comercial significativo, pelo que a produção destas plataformas ainda não é alvo de exploração. Como tal, não foi encontrada informação pormenorizada dos requisitos e modos de funcionamento destes sistemas, para a realização de uma comparação mais detalhada.

4. Arquitectura

O REVS é um sistema de voto desenhado para ambientes distribuídos sujeitos a falhas, nomeadamente a Internet. O sistema cumpre as características desejadas de um sistema de votação, como a correcção, democracia, privacidade e verificabilidade. O REVS também lida com falhas em cenários do mundo real, como falhas nas máquinas ou na comunicação, o que pode levar a interrupções no protocolo, permitindo assim um processo de voto seguro mesmo nesses ambientes.

No seguimento deste sistema de votação foi concebido o sistema MobileREVS, uma versão do REVS que permite a votação a partir de dispositivos móveis. A votação será realizada via rede telefónica móvel, nomeadamente GPRS/UMTS, fazendo uso das capacidades do próprio telemóvel.

4.1. Requisitos

Para que os sistemas de votação electrónica possam ser utilizados sem receio, são necessários protocolos que satisfaçam as propriedades patentes em praticamente todas as eleições. Deste modo, e à semelhança do REVS, o sistema MobileREVS cumpre igualmente todas as propriedades que um processo de *e-Voting* deve oferecer (ver Secção 1.1.2).

No entanto, e apesar dos dois sistemas assentarem em ambientes distribuídos, os seus ambiente de execução são diferentes. Enquanto que o REVS foi desenhado para clientes com computadores pessoais ligado à rede fixa, o sistema MobileREVS foi desenhado para ambientes móveis. Tal facto traduz-se na introdução de novos requisitos, para além do cumprimento das propriedades anteriormente definidas para sistemas de votação electrónica:

- **Mobilidade:** todos os serviços são oferecidos de igual modo em comparação a um local de votação fixo. As limitações da rede móvel (e.g. falta de rede) são as únicas restrições impostas;
- **Espaço ocupado pela aplicação:** tendo em conta o facto do espaço disponível na memória dos telemóveis ser um recurso limitado, o espaço ocupado pela aplicação deve ser minimizado;
- **Desempenho:** a pouca capacidade de processamento dos telemóveis impõe um cuidado acrescido no desenvolvimento de aplicações para o mesmo, pois a realização de operações pesadas (e.g. funções criptográficas) pode levar ao "congelamento" da interface do utilizador;
- **Consumo de energia:** a bateria é outra das limitações impostas a ter em conta; com efeito, a sua exaustão pode implicar uma interrupção involuntária no processo de votação;
- **Custos associados:** o serviço de comunicação GPRS/UMTS providenciado pelos operadores de telecomunicações móveis é taxado em função do volume de tráfego trocado na rede. Como tal, a quantidade de dados trocados na rede deve ser minimizado, para que os custos associados não se tornem um problema na utilização do sistema.

4.2. Problemas levantados

Embora alguns dos requisitos acima mencionados sejam de fácil resolução, outros necessitam de cuidados acrescidos. Isto para que, na prática, o processo de votação não termine com uma série de erros impossíveis de recuperar, ou mesmo erros que não sejam sequer detectados.

Durante uma eleição são gerados inúmeros dados sensíveis que têm que atravessar meios de comunicação controlados por terceiros. Consequentemente, tornam-se acessíveis a intervenientes mal intencionados, sendo objecto de ataque e de abuso. Assim, é necessário recorrer a técnicas criptográficas que permitam garantir a segurança do sistema.

Em relação ao protocolo do REVS, o principal problema levantado para a sua migração para ambientes móveis prende-se com o cumprimento do requisito de privacidade. Este requisito pode ser satisfeito recorrendo a várias técnicas criptográficas. No entanto, são escassas as oferecidas pelas bibliotecas base da plataforma J2ME. Assim, é necessário recorrer ao uso de bibliotecas externas às oferecidas de base.

Um dos requisitos das aplicações com interface utilizador é que mantenham a sua interface sempre disponível de modo a responder a qualquer pedido do utilizador. No entanto, a troca de mensagens na rede com servidores ou a realização de operações de cifra cujo processamento é potencialmente demorado pode levar ao "congelamento" da interface. Este problema advém da fraca capacidade de processamento dos telemóveis. Como tal, a solução passa pelo uso de *threads* que impeçam o bloqueio da interface durante a realização de operações de processamento intensivo.

Embora hoje em dia os telemóveis de gama alta já recorram a cartões de memória, esta continua a ser um recurso muito limitado nestes dispositivos. Como tal, o espaço ocupado pela aplicação desenvolvida deverá ser minimizado, de modo a poder abranger o maior número possível de dispositivos.

4.3. MobileREVS

Tendo em conta a robustez associada ao sistema REVS, a arquitectura do sistema MobileREVS manteve-se semelhante (ver Secção 3.1.1). Obviamente, o módulo de interacção do cliente com o restante sistema (Módulo Eleitor) foi substituído por um telemóvel (ver Figura 9), tendo havido uma reavaliação da funcionalidade suportada por este. Em particular, foi necessário avaliar que operações de segurança podem ser suportadas pelo telemóvel.

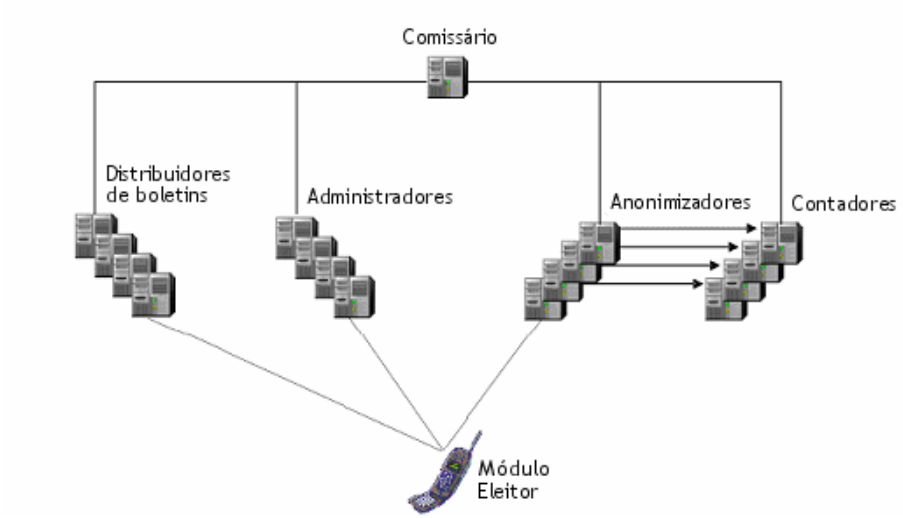

Figura 9 - Arquitectura do sistema MobileREVS

O Módulo Eleitor, descrito na Secção 3.1.1, foi desenvolvido para a plataforma J2ME (*Java 2 Micro Edition*), descrita com mais detalhe na Secção 2.2.1.

A comunicação com os servidores REVS é realizada exclusivamente através de RMI (*Remote Method Invocation*) [JZF03]. Porém, o RMI ainda não está presente de raiz nas máquinas virtuais J2ME, ficando apenas disponível através da utilização da *Optional Package* com o mesmo nome [JSR66]. Actualmente, esta *Optional Package* não está presente nos telemóveis dos diferentes fabricantes, ficando desta forma inviabilizada a comunicação directa entre o Módulo Eleitor e os servidores.

Para solucionar este problema optou-se pela criação de *proxies* para os servidores. Os *proxies* permitem criar um nível de indirecção que torna transparente no sistema os diferentes tipos de comunicação usados entre o Módulo Eleitor e os servidores. No fundo estes *proxies* não são mais do que *servlets*, i.e. pequenas aplicações *web* em Java que se executam nos servidores *web* interpretando os pedidos que lhe são direccionados e gerando as respostas adequadas.

Desta forma, cada servidor *web* passa a conter um *servlet* diferente. Cada *servlet* é responsável por interpretar o pedido HTTP/HTTPS do Módulo Cliente e transformá-lo num pedido RMI equivalente para o servidor J2SE respectivo (ver Figura 10). Para garantir a segurança nas comunicações os *servlets* devem estar localizados no mesmo servidor *web* de cada servidor J2SE respectivo do REVS. Assim, a comunicação entre o *servlet* e o servidor J2SE encontra-se protegida sempre que se assumir que o servidor *web* é, no seu todo, seguro.

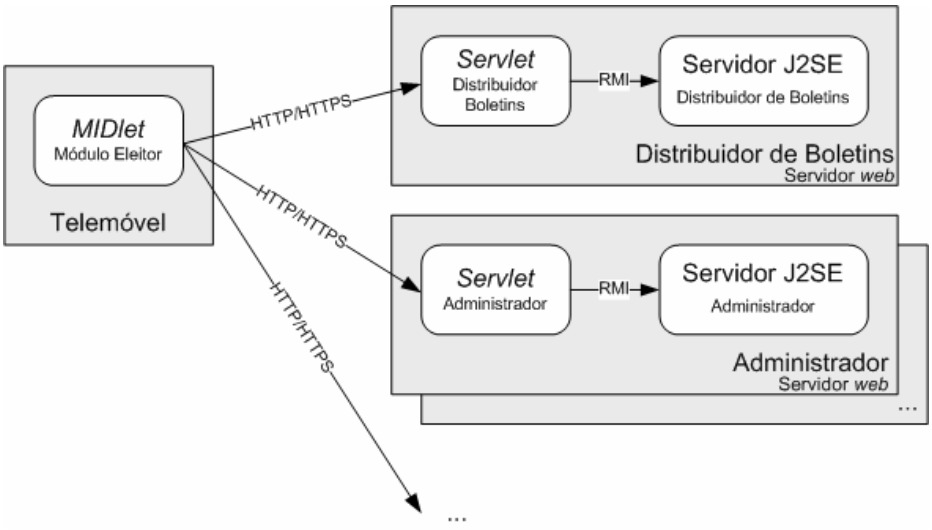

Figura 10 - Estrutura interna dos componentes do MobileREVS

4.4. Protocolo

Conforme foi referido anteriormente, o Módulo Eleitor do sistema MobileREVS não pode comunicar directamente com os servidores REVS, dada a inexistência de comunicação via RMI nas implementações de J2ME. Esta restrição implicou a criação de *proxies* nos servidores *web*, funcionando como intermediários da comunicação entre o *MIDlet* e os servidores REVS, de acordo com a Figura 10.

Desta forma, manteve-se o protocolo original do REVS, descrito na Secção 3.1.2, com as respectivas adaptações para a nova arquitectura, assegurando-se a manutenção das características e propriedades originais (ver Figura 11). A comunicação entre o telemóvel e os servidores passa, então, a ser mediada pelos *servlets*, que permitem traduzir os pedidos HTTP/HTTPS em pedidos RMI equivalentes para os servidores do REVS. Ao mesmo tempo, os *servlets* permitem adaptar algumas das estruturas de dados de J2ME para J2SE, e vice-versa, dadas as limitações existentes da máquina virtual para telemóveis (e.g. o J2ME não implementa estruturas de dados do tipo `List`).

A serialização de objectos para transmissão na rede é definida pela interface `ISerialize`. Através desta interface, todos os objectos de domínio implementam dois métodos de serialização: um método que transforma o objecto em questão num *byte array* para envio, e outro que realiza a operação inversa, ou seja, a transformação de um *byte array* no próprio objecto.

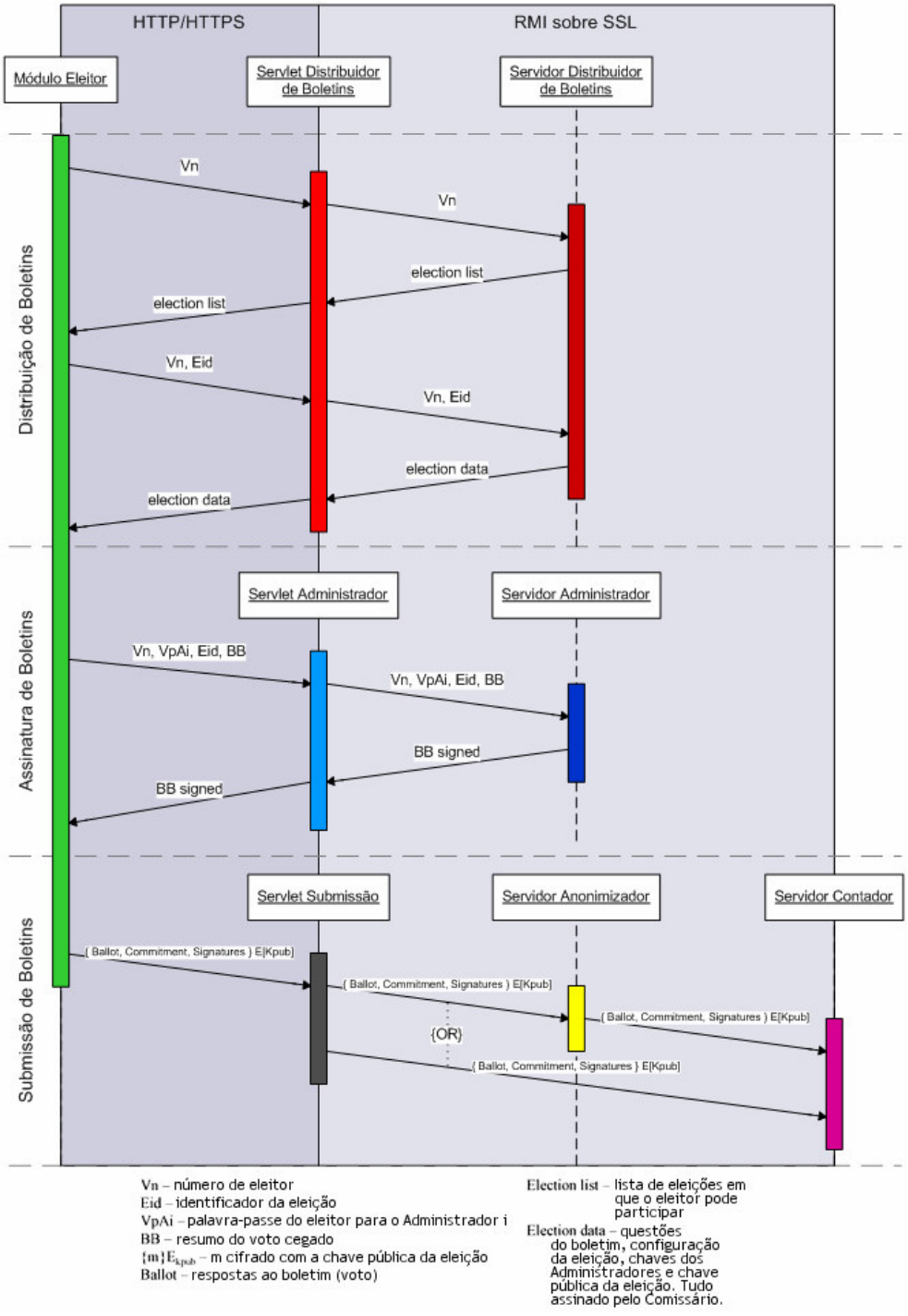

Figura 11 - Protocolo do MobileREVS

4.5. Módulo Eleitor

O Módulo Eleitor do MobileREVS decompõe-se estruturalmente em diversos módulos de código com diferentes finalidades (ver Figura 12): o módulo aplicação, o

módulo apresentação, o módulo domínio, o módulo comunicação, e o módulo armazenamento.

O módulo aplicação define o comportamento do Módulo Eleitor durante as várias etapas do processo de votação. É nele que está programada toda a lógica funcional e onde é mantido o estado do processo de votação.

O módulo apresentação efectua o tratamento da interface do utilizador para o sistema, realizando a apresentação dos diversos elementos gráficos e interpretando os comandos do utilizador.

No módulo domínio são representados os objectos manipulados pelo módulo aplicação, evidenciando as ligações e restrições entre os mesmos. Através deste módulo o Módulo Eleitor é capaz de criar uma representação abstracta dos objectos, e.g. boletins, e do estado do sistema.

O módulo comunicação permite ao módulo aplicação abstrair-se das características e idiossincrasias da comunicação, tratando dos aspectos mais específicos na comunicação realizada entre o Módulo Eleitor e os servidores.

Finalmente, o módulo armazenamento oferece uma interface simples para o armazenamento persistente de dados no telemóvel, lidando internamente com os pormenores que dele advêm.

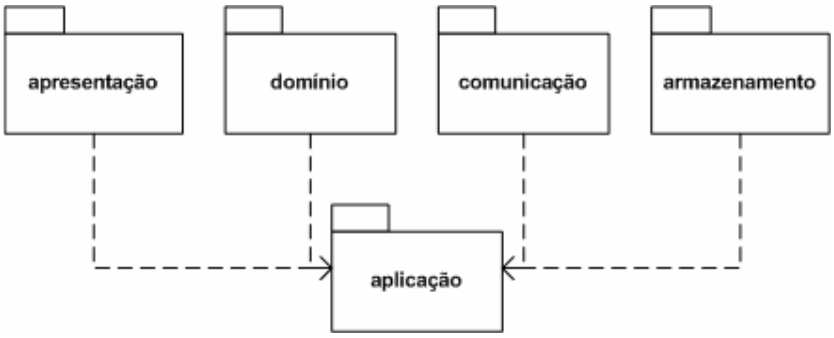

Figura 12 - Decomposição estrutural do Módulo Eleitor

Tal como a Figura 12 descreve, todos os módulos têm dependência com o módulo aplicação, pois é este último que detém o estado global e controla o fluxo do Módulo Eleitor.

Esta modularização tem o objectivo principal de tornar o código reutilizável e fácil de modificar (e.g., alterar a maneira como a apresentação dos boletins é realizada ao utilizador sem afectar o restante código).

4.5.1. Processo de votação

Conforme indicado anteriormente, o módulo aplicação define o comportamento do Módulo Eleitor durante o processo de votação. De seguida é apresentado em detalhe o fluxo normal desse processo e são contempladas as excepções ao mesmo.

Na óptica do eleitor o processo de votação decompõe-se em três etapas essenciais (ver Figura 13): (1) a escolha de uma eleição em que ele pode participar; (2) o preenchimento do respectivo boletim; e (3) a submissão do seu voto. Opcionalmente, na etapa de submissão do voto, o sistema possibilita o armazenamento do voto para continuação diferida do processo noutra altura (seja por falhas na comunicação ou para o eleitor se sentir mais seguro de que o seu voto foi entregue).

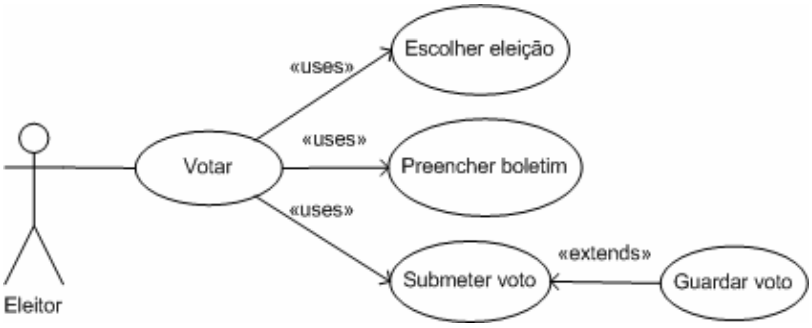

Figura 13 - Diagrama de casos de uso da interacção do eleitor com o sistema

Assim que iniciada, a aplicação apresenta ao utilizador um formulário para a autenticação. O utilizador deverá fornecer o seu identificador de eleitor e a palavra-passe correspondente. Ao contrário do REVS não é possibilitada a introdução conjunta de um PIN, facilitando a utilização da aplicação pelo eleitor. Optou-se antes pela utilização de uma palavra-passe com um número mínimo de 8 caracteres, de modo a reforçar a segurança da autenticação.

Os dados serão, então, enviados para um dos Distribuidores de Boletins que efectuará a autenticação do eleitor. Se os dados em causa estiverem correctos o utilizador tem a possibilidade de interagir com o sistema para votação. O utilizador também tem a capacidade de definir algumas opções de configuração da utilização do sistema nesta etapa.

Inicialmente são apresentadas as eleições disponíveis, i.e. as eleições em que o eleitor pode participar, fornecidas por um dos Distribuidores de Boletins na interacção anteriormente referida. Assim que o utilizador escolher uma das eleições possíveis o Módulo Eleitor verifica se já existe uma sessão armazenada no telemóvel para essa eleição. Entenda-se por sessão o estado que pode ser armazenado pelo utilizador durante uma eleição em que este participa (e.g., respostas ao boletim, assinaturas dos Administradores, …). Se já existir uma sessão armazenada é fornecida a possibilidade de enviar novamente o mesmo voto (não é possível alterá-lo). Caso não exista nenhuma sessão armazenada a aplicação requer ao Distribuidor de Boletins o boletim de voto da eleição escolhida, que é apresentado ao utilizador. Juntamente com o boletim são fornecidas as configurações necessárias do sistema (endereços dos vários servidores, chave pública da eleição, chaves públicas dos Administradores, …).

Depois do utilizador preencher o boletim terá de confirmar as suas respostas antes de as submeter. Nesta etapa o utilizador pode rever as suas respostas e detectar eventuais erros, tendo a possibilidade de as corrigir.

Logo após a confirmação das respostas do utilizador, o sistema está pronto para submeter o seu voto. Nesta etapa é verificada a opção de personalização relativa ao armazenamento da sessão. Por omissão, a opção está configurada para guardar a sessão, mas o utilizador também tem a possibilidade de nunca a guardar ou que lhe seja perguntado sobre o que fazer. Posteriormente, o voto é enviado aos Administradores para ser assinado. Se não for possível comunicar com a maioria deles o processo de votação é interrompido neste ponto, oferecendo a possibilidade ao utilizador de efectuar uma nova tentativa ou armazenar o voto, caso ainda não o tenha feito, para poder continuar a votação noutra altura (pode encontrar mais detalhes descritos adiante).

Aquando do sucesso da comunicação com os Administradores a sessão será actualizada se tiver sido guardada anteriormente. Note-se que caso o utilizador esteja a

efectuar a continuação diferida de uma sessão esta será sempre actualizada. Posteriormente, o voto é submetido para os Anonimizadores/Contadores. Mais uma vez poderão existir problemas na comunicação. Nesse caso o utilizador será avisado da situação de erro e ser-lhe-ão oferecidas as possibilidades de tentar novamente a comunicação ou armazenar a sessão, se ainda não o tiver feito, para completar a votação noutra altura.

Assim que o processo de votação for bem sucedido o utilizador é informado da sua correcta conclusão. Neste instante, o utilizador poderá escolher entre participar noutra eleição disponível ou abandonar a aplicação.

A Figura 14 esquematiza o fluxo de tarefas do Módulo Eleitor no processo de votação. Algumas tarefas da aplicação neste processo foram definidas para (1) optimizar o funcionamento de todo o sistema e (2) facilitar a utilização por parte dos eleitores. As excepções a este fluxo (situações de erro) não se encontram contempladas. Os pontos de decisão com influência directa do utilizador encontram-se delineados a tracejado.

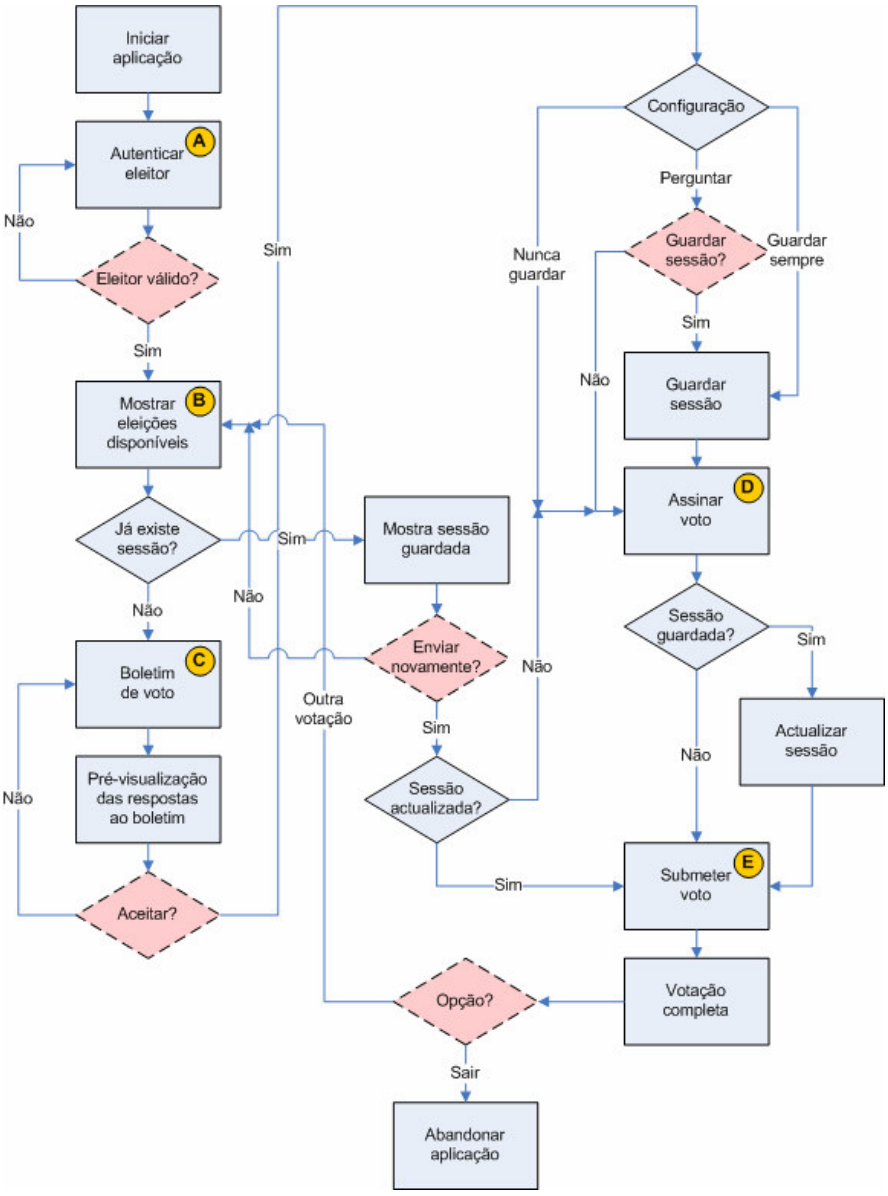

Figura 14 - Fluxo de tarefas do Módulo Eleitor

A comunicação com os servidores envolvidos na votação (Distribuidores de Boletins, Administradores e Anonimizadores/Contadores) é realizada em quatro instantes diferentes. Primeiro é efectuada a comunicação com o Distribuidor de Boletins para, simultaneamente, autenticar o eleitor **(A)** e obter a lista de eleições disponíveis **(B)**. Posteriormente, o Distribuidor de Boletins é novamente contactado para se obter um boletim de voto para a eleição escolhida **(C)**. Após o preenchimento do boletim a aplicação comunica com os Administradores para assinarem o voto **(D)**. Por fim, o voto é entregue aos Anonimizadores/Contadores para fazer parte da contagem final **(E)**. Note-se que estas duas últimas comunicações (para assinatura e submissão do voto) não são diferenciadas na perspectiva do utilizador. O utilizador efectua a sua votação completando as mesmas tarefas que no processo habitual em papel (escolha da eleição, preenchimento do boletim e entrega do voto), garantindo-se uma boa usabilidade da aplicação.

Falhas na comunicação

Conforme foi referido, o fluxo principal do processo de votação possui quatro etapas onde existe comunicação com os servidores envolvidos. A existência de falhas na comunicação em duas dessas etapas – **D** e **E** – pode ter repercussão na correcta conclusão do processo de votação. Desta maneira, é essencial que o Módulo Eleitor seja capaz de oferecer ao utilizador a possibilidade de recuperar de tais situações ou de, pelo menos, preservar os seus votos.

No envio do voto aos Administradores para ser assinado é possível que o Módulo Eleitor não consiga comunicar com a maioria deles (e.g., falha de rede, maioria dos Administradores em baixo, etc.). Assim, há que notificar o utilizador de tal facto e oferecer possibilidades para minimizar ao máximo os efeitos do problema. Neste caso, é apresentado ao utilizador a possibilidade de enviar novamente o voto para os Administradores, continuando o processo se a comunicação for recuperada. Por outro lado, a falha de comunicação pode ser demorada. Assim, e caso a sessão não tenha sido armazenada anteriormente, o utilizador tem a possibilidade de a armazenar para continuar o processo numa outra altura, sem perder o seu voto. A Figura 15 representa o fluxo de tarefas realizadas pelo Módulo Eleitor em caso de falhas na etapa de assinatura do voto.

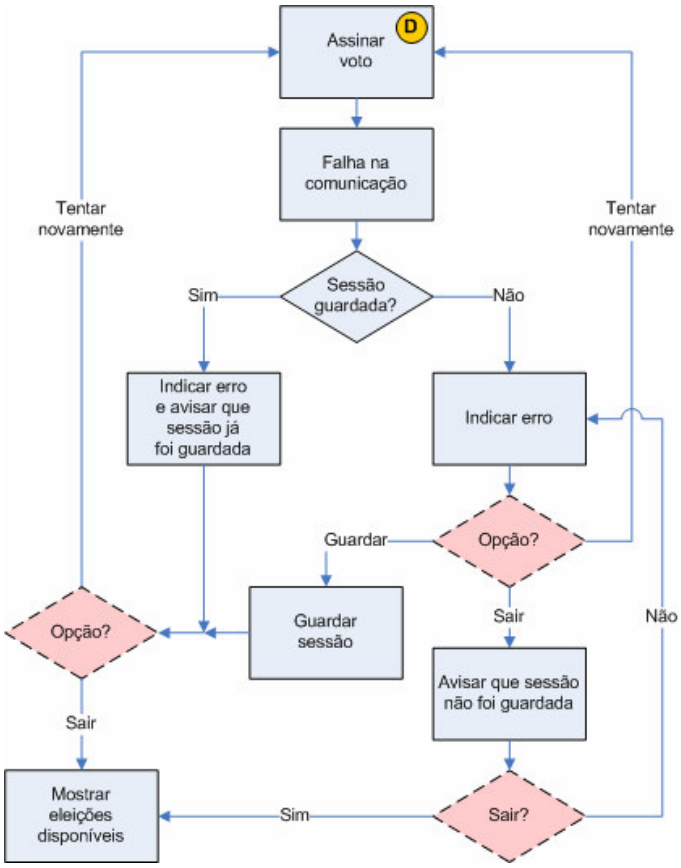

Figura 15 - Fluxo de tarefas em caso de falhas na assinatura do voto

Também existe a possibilidade de falhas na comunicação entre o Módulo Eleitor e os Anonimizadores/Contadores, durante a etapa de submissão do voto. O comportamento da aplicação será semelhante ao descrito anteriormente para a etapa de assinatura de votos.

4.5.2. Criptografia

O MobileREVS utiliza um conjunto de algoritmos criptográficos da biblioteca Bouncy Castle (*Light Edition*) para assegurar as propriedades de segurança, integridade, privacidade e não-repúdio. A escolha da utilização desta biblioteca no Módulo Eleitor deveu-se a duas razões: (1) a implementação raiz do J2ME não oferece nenhuma destas funcionalidades; e (2) as alternativas são escassas, e o seu acesso é muito limitado (ver Secção 2.3.3).

As assinaturas do MobileREVS são realizadas utilizando chaves RSA de 1024 bits e um resumo SHA-1 (160 bits). Para cifra simétrica é utilizado o 3DES. A cifra do pacote a enviar utiliza, assim, uma combinação híbrida RSA-3DES, ou seja, os dados são cifrados com 3DES e a chave simétrica é cifrada com RSA.

4.5.3. Comunicação

Genericamente existem três meios de comunicação usados pelos telemóveis. São eles o SMS (*Short Message Service*), o WAP (*Wireless Application Protocol*) e o GPRS (*General Packet Radio Service*), ou UMTS (*Universal Mobile Telecommunications System*) no caso das redes móveis de terceira geração. A utilização do GPRS/UMTS como meio de comunicação para o sistema MobileREVS, conforme apresentado no início do Capítulo 4, foi determinada com base na adequação do mesmo para os propósitos do sistema.

O SMS e o WAP revelam-se dois meios de comunicação visivelmente desajustados para o MobileREVS. Por um lado, ambos têm baixas taxas de transferência e latências elevadas, algo que no caso do MobileREVS não é aceitável dada a troca de quantidades elevadas de dados. Por outro lado, são meios de comunicação muito inflexíveis neste contexto. O SMS exige uma sintaxe de escrita de mensagens. O WAP exige a alocação permanente de recursos ponto-a-ponto, enquanto a ligação está activa. Além disso, a utilização destes meios de comunicação teriam custos impraticáveis no ambiente real das operadoras de telecomunicações móveis: (1) através do SMS seria preciso trocar muitas mensagens, dado o limite máximo de 160 caracteres por mensagem; e (2) o serviço WAP é cobrado por tempo de utilização, ao invés da quantidade de dados trocada como acontece em GPRS/UMTS.

O GPRS/UMTS tem vindo a ser utilizado cada vez mais frequentemente como meio de acesso à Internet e troca de dados a partir de dispositivos móveis, apresentando custos cada vez mais reduzidos. Com base nos dados apresentados, a escolha do meio comunicação usado recaiu sobre o GPRS/UMTS.

Relativamente ao protocolo de comunicação usado foi escolhido o HTTP/HTTPS. Apesar de existirem diversos tipos de ligação disponibilizados pelo MIDP 2.0 (HTTP/HTTPS, *sockets*, datagramas, …) a utilização de HTTP/HTTPS oferece algumas vantagens [FZ01]: (1) portabilidade, dada a menor dependência com as redes de comunicação utilizadas; (2) gestão automática de ligações; (3) encapsulação de qualquer tipo de dados, através dos MIMEs (*Multipurpose Internet Mail Extensions*); (4) facilidade de integração com os servidores actuais. Além disso, a utilização de HTTP oferece ainda uma vantagem adicional de compatibilidade, visto que este é o único tipo de ligação que deve ser obrigatoriamente disponibilizado pelos fabricantes de telemóveis [Feng].

Porém, visto que o desenvolvimento do sistema assenta sobre a versão 2.0 do MIDP, escolheu-se utilizar o protocolo HTTP/HTTPS, de forma a possibilitar a criação de canais seguros entre o telemóvel e os servidores, sempre que tal for possível.

4.5.4. *Parsing* dos boletins

De acordo com o protocolo do MobileREVS (ver Secção 4.4) o Distribuidor de Boletins envia um boletim de voto para o cliente com a estrutura XML definida em [JZF03]. Como tal, torna-se imperativo que o Módulo Eleitor seja capaz de processar esta metalinguagem, mapeando o boletim num objecto de domínio para manipulação. De maneira a cumprir esta tarefa é necessário (1) utilizar um dos *parsers* de XML existentes ou (2) produzir um *parser* específico para a aplicação.

A produção de um *parser* específico apenas faria sentido para a optimização dos recursos consumidos pela aplicação. Visto que actualmente já existem diversos *parsers* disponíveis que foram desenhados com esta finalidade optou-se pela utilização de um

parser existente [Knudsen02, Ghosh03]. A análise de *parsers* existentes encontra-se descrita mais à frente nesta secção.

Considerações de performance

O *parsing* de XML é uma tarefa que, tradicionalmente, consome bastantes recursos de processamento e memória. Conforme foi discutido na Secção 2.2.1, os telemóveis são dispositivos de processamento lento e com memória escassa. Além disso, alguns telemóveis limitam o tamanho máximo dos *MIDlets*, tornando-se importante minimizar o espaço ocupado pelos mesmos.

Pretende-se, portanto, que o *parser* utilizado seja simultaneamente eficiente e compacto, ocupando a menor quantidade de espaço possível em memória.

Técnicas e *parsers* existentes

As técnicas de *parsing* de XML estão ligadas com os três tipos fundamentais de *parsers*. Os aspectos que diferenciam a sua escolha são: (1) o modo como se pretende que a aplicação se comporte no *parsing* dos documentos; e (2) o tipo de documentos que se pretende manipular.

1. Um ***model parser*** interpreta todo um documento XML e cria uma representação do mesmo em memória;
2. Um ***push parser*** interpreta sequencialmente um documento XML. À medida que várias partes são encontradas o *parser* informa um *listener object*;
3. Um ***pull parser*** interpreta pequenas porções do documento de cada vez. A interpretação é guiada pela aplicação, que requer repetidamente o pedaço seguinte.

Ao criar uma representação de um documento XML inteiro em memória, o *model parser* é o que utiliza mais memória em execução. Claramente, esta solução não se adequa ao desenvolvimento de aplicações para telemóveis pelas restrições referidas anteriormente.

A grande diferença entre os restantes *parsers*, *push* e *pull*, reside no ponto de controlo da tarefa de *parsing*. No *push parser* o controlo está a cargo do *parser*, que informa a aplicação sempre que tem nova informação; no *pull parser* o controlo é da aplicação, que requer explicitamente as partes seguintes do documento.

Desta maneira é possível verificar que o *pull parser* é o tipo de *parser* mais adequado para os *MIDlets*. Ao permitir que a aplicação detenha o controlo da tarefa de *parsing* do XML, esta pode ser desenvolvida de maneira a manter apenas em memória a parte com que está a lidar no momento. Quando mais nada tiver a fazer com essa parte pode libertá-la da memória e passar à seguinte.

Actualmente existem alguns *parsers* Java específicos para dispositivos de recursos limitados, como é o caso dos telemóveis. A Tabela 1 resume as principais características de cada um deles.

Nome	Tipo	Tamanho (kB)	Integração MIDP
ASXMLP 020308	*push, model*	6	Sim
kXML 2.2	*pull*	11	Sim
kXML 1.2	*pull*	16	Sim
MinML 1.7	*push*	14	Não
NanoXML 1.6.4	*model*	10	*Patch*
TinyXML 0.7	*model*	12	Não
Xparse-J 1.1	*model*	6	Sim

Tabela 1 - *Parsers* Java específicos para telemóveis

A coluna "Tipo" indica o tipo de *parser* de acordo com o que foi indicado anteriormente. O tamanho representa o espaço ocupado pelas classes do *parser*, que corresponde a uma aproximação do aumento do tamanho total da aplicação que as utilize. A coluna "Integração MIDP" indica a facilidade com que o *parser* pode ser integrado num *MIDlet*. Embora três dos *parsers* não possam ser directamente utilizáveis pelos *MIDlets* apenas são precisas simples modificações para que tal passe a ser possível.

Conforme foi determinado, o *parser* ideal para colmatar as limitações impostas pelos telemóveis aos *MIDlets* deve ser do tipo *pull*. De acordo com a Tabela 1 apenas o *parser* kXML [KXP] é deste tipo, pelo que foi o escolhido para o desenvolvimento do projecto. Obviamente, optou-se pela *release* mais recente da versão 2 porque o seu tamanho é bastante inferior (aproximadamente 11 KB) e a versão já se encontra estável.

Optimização

Seguidamente são apresentadas algumas das optimizações de performance dos *MIDlets* que efectuam *parsing* de XML. Estas permitem que os *MIDlets* se possam executar com tempos de resposta aceitáveis perante as limitações de recursos impostas pelos telemóveis, indicadas anteriormente.

É importante reter que nas redes de comunicação sem fios usadas pelos telemóveis o estabelecimento de uma ligação é lento, assim como o ritmo de transferência de dados. Por isso, é essencial que a troca de dados em XML entre o servidor e o *MIDlet* seja feita com o menor número de mensagens possível e envolva apenas a informação realmente necessária para a aplicação.

Outra maneira de melhorar o tempo de resposta da aplicação é a de realizar as operações de *parsing* de XML numa *thread* separada. Deste modo evita-se que a interface "congele" com estas operações, oferecendo melhor usabilidade. Idealmente, enquanto o *parsing* do XML está a ser realizado a aplicação poderá estar a realizar outras tarefas ou a apresentar os dados à medida que estes vão ficando disponíveis (apenas possível quando não é utilizado um *model parser*).

4.5.5. Armazenamento

Durante o processo de votação o utilizador tem a capacidade de armazenar o seu voto, algo que lhe permitirá retomar o processo de votação numa outra altura. Como tal, é necessário recorrer a mecanismos que efectuem armazenamento persistente.

O armazenamento persistente pode ser efectuado em dois tipos de memória: (1) na memória interna do telemóvel; ou (2) num cartão de memória, instalado no

telemóvel. A principal vantagem do armazenamento do voto ser feito num cartão de memória é a capacidade do eleitor poder retomar a votação a partir de outros telemóveis.

No J2ME existem bibliotecas que permitem lidar com o armazenamento de dados nestes dois tipos de memória. O *Record Management Store* (RMS) oferece uma interface para os *MIDlets* armazenarem os seus dados na memória do telemóvel. Esta biblioteca é oferecida no perfil MIDP do telemóvel, encontrando-se presente em qualquer implementação J2ME. Por outro lado, o acesso aos cartões de memória está condicionado pela presença da *Optional Package FileConnection*, do pacote *PDA Optional Packages* [JSR75], dependendo assim do fabricante de cada telemóvel. No entanto, já existe uma boa quantidade de telemóveis a implementar esta *Optional Package*.

Perante os factores apresentados, a decisão recaiu sobre a utilização do RMS para o armazenamento do voto. A utilização da *Optional Package FileConnection* é uma outra possibilidade, podendo ser incluída numa *release* posterior do MobileREVS.

5. Implementação

Neste capítulo são apresentados os pormenores da implementação do projecto MobileREVS. Numa primeira parte são descritas algumas escolhas tomadas na implementação do Módulo Eleitor. Posteriormente, são referenciadas as alterações efectuadas aos servidores REVS para o sistema MobileREVS. Por fim, é apresentada a aplicação "REVS Ballot Editor", um editor gráfico de boletins para o REVS e para o MobileREVS.

5.1. Módulo Eleitor

O Módulo Eleitor do MobileREVS foi concebido com a preocupação de minimizar a utilização dos recursos dos telemóveis e oferecer uma boa experiência de utilização. Assim, nas secções seguintes são abordados alguns pormenores de implementação relativos à estruturação da lógica da apresentação e à comunicação com os servidores, um dos pontos fulcrais para o desempenho do sistema.

5.1.1. Lógica da apresentação

A lógica de apresentação engloba todas as classes que lidam com a apresentação de elementos no ecrã, assim como com os comandos introduzidos pelo utilizador. Foi desenvolvida segundo critérios que permitem manter a escalabilidade no desenvolvimento, separando a gestão dos conteúdos apresentados dos pormenores da funcionalidade.

Para tal, foi criado um tipo abstracto `MenuScreen`, juntamente com uma interface `IMenuScreen`, que representa um ecrã a apresentar ao utilizador e que, como tal, deve possuir as seguintes características:

- Criar os seus próprios elementos de interface (texto, formulário, lista, comandos, …), construídos através do método de interface `make()`;
- Revelar comportamento próprio específico, ou seja, saber decidir qual o próximo passo a dar em função dos comandos do utilizador nesse ecrã. Para isso, cada ecrã implementa a interface `CommandListener`, escutando os seus comandos específicos.

Desta forma é possível repartir e separar logicamente a interface gráfica e comportamental por cada ecrã. Cada ecrã é responsável por efectuar as transições e operações necessárias de acordo com os comandos do utilizador.

O encadeamento lógico dos vários ecrãs respeita uma configuração hierárquica, onde ecrãs mais específicos (filhos) sucedem aos mais gerais (pais), sendo que o resultado é um grafo de ecrãs interligados (ver Figura 16).

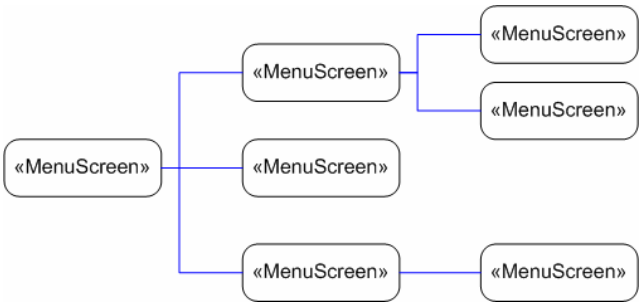

Figura 16 - Configuração hierárquica de ecrãs

Para que cada ecrã respeite as características acima indicadas e se insira num grafo hierárquico de ecrãs conforme acabou de ser descrito, adoptou-se uma política de transições baseada em criações e invocações (ver Figura 17) [IMG]. Segundo esta política, cada ecrã é responsável pela criação dos seus subsequentes e, simultaneamente, tem a capacidade de invocar o seu antecessor (e.g. para regressar ao ecrã anterior).

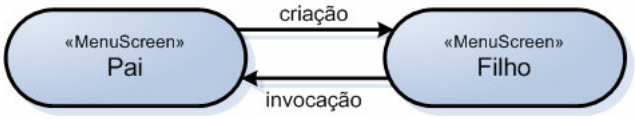

Figura 17 - Política de transições dos ecrãs

Na criação de um filho, o ecrã pai passa-lhe a sua referência que é depois mantida pelo filho para realizar as invocações conforme necessário.

Seguindo esta metodologia de implementação de ecrãs o desenvolvimento da interface mostra-se bastante adaptável a:

- **Adição de novos ecrãs**: caso se pretender adicionar um novo ecrã apenas é necessário: (1) criar uma classe que herde as características do ecrã abstracto `MenuScreen` e implemente a interface `IMenuScreen`; (2) definir os métodos `make()` e `commandAction()` para, respectivamente, definir o conteúdo e o comportamento desse ecrã; (3) criar um pequeno método no ecrã pai que faça a criação e transição para o ecrã filho;
- **Alteração de ecrãs existentes**: as alterações na interface são localizadas, e realizadas apenas na classe que representa o ecrã, mantendo-se o restante código inalterável.

5.1.2. Comunicação

Outro dos aspectos importantes da comunicação é o recurso a *threads* no contacto com os servidores nas fases de assinatura e submissão do voto. Esta decisão prende-se essencialmente com questões de desempenho. Como o estabelecimento de uma ligação é uma operação algo demorada, a utilização de *threads* permite que a aplicação contacte diversos servidores em simultâneo, sem ter de esperar pela resposta de um para passar ao próximo. Esta abordagem torna-se crítica em situações de falhas na comunicação, já que o *timeout* da ligação induz um atraso significativo no processo de votação.

Na obtenção das assinaturas dos Administradores são abertas ligações com *t* Administradores, onde *t* é o número de assinaturas requeridas para a eleição. Caso não se obtenham todas as assinaturas necessárias, e.g. devido a falhas na comunicação, são abertas $t - n$ novas ligações com outros Administradores, onde $t - n$ é o número de assinaturas não conseguidas. No limite são abertas tantas ligações quanto o número total de Administradores, de maneira a que todos sejam contactados pelo menos uma vez.

5.2. Servidores

O projecto MobileREVS também contemplou pequenas alterações aos servidores REVS.

Em primeiro lugar foi adicionada uma opção de depuração, que permite fazer o *logging* dos vários pedidos que vão chegando a cada um dos servidores. Nos *logs* é registado o eleitor que efectuou o pedido assim como o teor do mesmo.

Outra das alterações teve em consideração um problema actual do REVS: não existe forma dos eleitores se autenticarem perante os Distribuidores de Boletins. Desta maneira, eleitores mal intencionados poderão ter acesso a boletins cujo conteúdo possa ser confidencial, utilizando números de eleitores com esse privilégio. Assim, foi adicionada autenticação dos eleitores perante os Distribuidores de Boletins, ficando o acesso à eleição restrito tanto na fase de distribuição de boletins como na fase de assinatura dos votos.

5.3. REVS Ballot Editor

O "REVS Ballot Editor" é um editor gráfico de boletins XML para o REVS e para o MobileREVS. A aplicação foi desenvolvida em C# e Windows Forms, e permite criar novos boletins de voto de raiz ou editar existentes a partir do ficheiro XML respectivo. Seguidamente são apresentados alguns pormenores da sua implementação. No Anexo D encontra-se o Manual do Utilizador do "REVS Ballot Editor".

5.3.1. *Parsing* dos boletins de voto

Para o *parsing* dos boletins de voto em formato XML é utilizada uma versão adaptada para C# do SimpleDOMParser, desenvolvido para a plataforma Java [GY02]. O SimpleDOMParser foi escolhido para a aplicação por ser um *lightweight parser*. Ou seja, possui apenas as funções essenciais para a manipulação de XML, suficientes para os requisitos do "REVS Ballot Editor".

O *parsing* dos boletins tem em consideração a estrutura apontada em [JZF03] para os boletins do REVS. Para realizar a validação dessa estrutura foi criado o método `ValidateBallot`. Este método analisa a árvore DOM (Document Object Model) carregada a partir do ficheiro, validando a sua estrutura face à dos boletins do REVS. O método foi desenvolvido de forma a conseguir recuperar de algumas inconsistências dos ficheiros XML (e.g. atributos irrelevantes ou etiquetas XML repetidas), devolvendo uma excepção apenas quando uma etiqueta ou atributo obrigatórios não se encontram presentes.

5.3.2. Pré-visualização dos boletins

Como a interface da aplicação se baseia em listas e caixas de diálogo é possível que existam erros humanos na criação dos boletins (e.g. o esquecimento da introdução de uma resposta a uma questão). Assim, foi criada a funcionalidade de pré-visualização de um boletim para que o utilizador tenha uma percepção visual do aspecto final do mesmo, dando conta de eventuais erros antes de o armazenar num ficheiro XML.

5.3.3. Personalização

A aplicação "REVS Ballot Editor" pode ser personalizada em termos da sua utilização habitual.

É possível definir directorias por omissão para a leitura e armazenamento de boletins, de forma a não ser sempre necessário localizar os directórios se os mesmos se mantiverem entre diferentes utilizações.

Também é oferecida a capacidade de definir códigos automáticos para os diferentes elementos do boletim (grupos, questões e respostas) durante a criação e edição de boletins. Neste caso, os códigos são gerados e mantidos sequencialmente pela aplicação, situação mais normal na criação e edição de boletins, sem ser necessária a sua introdução pelo utilizador.

Estas configurações são armazenadas persistentemente num ficheiro de denominado `REVSBallotEditor.cfg`, na mesma directoria onde se encontra o executável da aplicação. Sempre que este ficheiro não estiver presente ou for inconsistente são assumidas as configurações de omissão.

6. Avaliação

Neste capítulo é apresentada a avaliação global do sistema MobileREVS. A primeira parte da avaliação pretende determinar se as propriedades de votação electrónica do REVS, descritas na Secção 1.1.2, continuam a ser respeitadas no sistema MobileREVS. Posteriormente, é realizada a avaliação quantitativa do MobileREVS ao nível dos requisitos apontados na Secção 4.1.

6.1. Propriedades

A primeira parte da avaliação serve para garantir que as propriedades desejadas para os SVE, presentes no REVS, se mantêm no sistema MobileREVS.

6.1.1. Correcção

(1) Um voto não pode ser alterado pois isso iria invalidar a assinatura de todos os Administradores. (2) A eliminação de um dos votos dos Contadores não é uma tarefa trivial, visto que eles podem estar em qualquer servidor e seria necessário eliminar o voto de todos eles. Para além disso, é sempre possível ao utilizador efectuar nova submissão caso tenha o voto guardado persistentemente. (3) Tendo em conta que as assinaturas são publicadas com os votos e podem ser verificadas por qualquer pessoa, é impossível que um voto inválido faça parte da contagem final.

6.1.2. Democracia

(1) A um eleitor apenas pode ser concedida uma assinatura para uma determinada eleição caso este tenha sido incluído na lista de eleitores da mesma. (2) Cada eleitor só pode votar uma vez em cada eleição pois o seu voto apenas é considerado na contagem final caso tenha a assinatura da maioria dos Administradores ($t > n/2$). Assim, torna-se impossível um eleitor adquirir mais do que um voto válido. (3) A obtenção de resultados parciais no MobileREVS só é possível com o conluio da comissão eleitoral, que possui a chave da eleição, e dos Anonimizadores ou Contadores, que possuem os votos cifrados. Assim, se a comissão eleitoral e pelo menos t Administradores forem honestos todos os aspectos da democracia são garantidos.

6.1.3. Privacidade

(1) A única forma de uma autoridade eleitoral conseguir ligar um eleitor a um voto é através de um conluio entre os Administradores, Anonimizadores e Contadores. Enquanto os Anonimizadores se mantiverem honestos é impossível estabelecer qualquer ligação temporal entre o momento da assinatura dos votos e a sua submissão aos Contadores e, como tal, o anonimato é garantido. (2) O sistema não é *Receipt-free*, ou

seja, é possível um eleitor provar que votou de determinada maneira. Como tal, este aspecto da propriedade "Privacidade" não é assegurada.

6.1.4. Verificabilidade

A contagem final dos votos pode ser feita por qualquer pessoa através da verificação das assinaturas nos votos e da sua soma. Cada eleitor pode verificar se o seu voto está correcto, e assume que os outros votos também estão correctos devido à assinatura que eles têm.

6.1.5. Robustez

Disponibilidade

Visto que todos os servidores podem ser replicados o MobileREVS não tem um ponto de falha único. O sistema está disponível enquanto houver um conjunto mínimo de servidores a funcionar correctamente. Esse conjunto mínimo é constituído por um Distribuidor de Boletins, *t* Administradores, e um Anonimizador ou Contador.

Continuação diferida

O eleitor pode recuperar de uma interrupção desde que mantenha em memória os dados únicos do voto, ou seja, o *Commitment* e os factores de cegamento (ver Secção 3.1.2).

Resistência ao conluio

No MobileREVS nenhuma autoridade eleitoral consegue corromper, sozinha, as propriedades do SVE. No entanto, o conluio contra a propriedade da democracia merece especial atenção. Num conjunto de n Administradores e t assinaturas requeridas é necessário o conluio de $n - t + 1$ Administradores para impedir um eleitor de votar, ou seja, à medida que t aumenta torna-se mais fácil impedir o processo de votação. Por outro lado, para simular um voto válido por parte dos Administradores é necessária a cooperação de t deles, o que se torna mais difícil à medida que t aumenta.

Como tal, é necessário um balanceamento ponderado sobre os valores de n e t.

6.2. Requisitos

Nesta secção são avaliados os requisitos considerados para o MobileREVS, apontados na Secção 4.1.

6.2.1. Mobilidade e falhas de energia

Em caso de falha na rede móvel ou falhas de energia (e.g. bateria descarregada) o eleitor pode recuperar o estado do seu voto se o tiver salvaguardado previamente. A

opção de guardar o voto (juntamente com todas as informações do seu estado) está activada por omissão.

O estado do voto é salvaguardado em três partes distintas: (1) após o preenchimento do boletim, prevenindo assim que o utilizador o tenha de voltar a preencher; (2) após obter as assinaturas, quer perfaçam o total de assinaturas requeridas ou não, evitando posteriores comunicações desnecessárias com os Administradores já contactados; e, finalmente, (3) no final do processo de votação, assinalando a submissão com sucesso do voto.

6.2.2. Memória

O espaço total ocupado pelo Módulo Eleitor, aplicação a ser instalada no telemóvel, é de 139 KB. Conforme acordado com a PT Inovação, este é um tamanho considerado admissível visto estar na média de outras aplicações para telemóveis.

Relativamente à memória consumida durante a execução da aplicação, esta vai variando consoante a fase do protocolo de votação do MobileREVS. Tendo em conta que o consumo de memória não depende do dispositivo móvel, optou-se por recorrer à ferramenta de monitorização de memória do Wireless Toolkit 2.2 para a analisar.

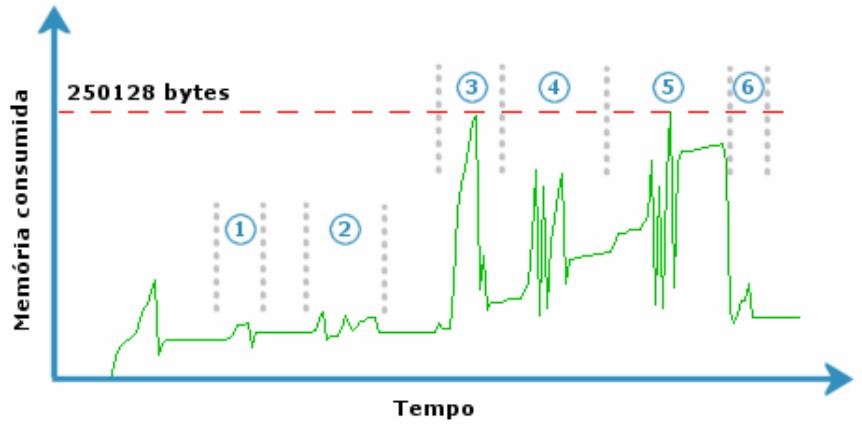

Figura 18 - Memória volátil consumida durante o processo de votação

A Figura 18 representa a evolução da utilização da memória volátil durante um processo de votação com duas assinaturas requeridas e um Anonimizador/Contador. As etapas assinaladas na figura são descritas em seguida:

1. Contacto com o Distribuidor de Boletins para obtenção da lista de eleições;
2. Contacto com o Distribuidor de Boletins para obtenção do boletim de voto;
3. Preparação da votação;
4. Obtenção de uma assinatura junto do 1º Administrador;
5. Obtenção de outra assinatura junto do 2º Administrador;
6. Submissão do voto para o Anonimizador/Contador.

De acordo com a figura, a memória volátil consumida pela aplicação não sofre variação significativa durante o contacto com os Distribuidores de Boletins (1 e 2). Posteriormente, observa-se uma subida abrupta do consumo, até atingir o seu ponto

máximo (3) de aproximadamente 250 KB, situação que ocorre logo após a preparação da votação e antes do envio do voto aos Administradores. Por preparação da votação entenda-se a geração do pacote de votação, do *Commitment* e dos factores de cegamento.

Para cada Administrador envolvido no processo de votação assiste-se a um incremento significativo da memória volátil consumida do telemóvel (4 e 5). Esta situação acontece no momento da resposta de cada um, visto serem retidas todas as assinaturas em memória volátil até ser possível armazená-las em memória persistente. Uma alternativa possível poderia passar pelo armazenamento imediato de cada assinatura, logo após a sua recepção. Porém, os acessos às funções de armazenamento persistente induziriam um atraso significativo no processo. Além disso, a memória volátil consumida pela aplicação em situações mais comuns de configuração do sistema (no exemplo foram usados dois Administradores) é relativamente baixa dados os limites dos telemóveis actuais. No caso do Nokia 6600, por exemplo, são disponibilizados 3 MB de memória volátil para as aplicações Java [FN].

No final, depois de armazenados persistentemente, o voto e as assinaturas dos Administradores são enviados aos Anonimizadores/Contadores (6).

Analisando agora a utilização da memória persistente, esta também apresenta variações significativas. A Tabela 2 revela o espaço ocupado em memória persistente pelo voto, nas fases do processo de votação onde é possível salvaguardá-lo.

# assinaturas requeridas	Memória persistente consumida (bytes)		
	após preenchimento do boletim	após obtenção das assinaturas	após submissão do voto
1	768	511	511
2	1110	654	654
3	1452	799	799
4	1794	934	934

Tabela 2 - Memória persistente consumida durante o processo de votação

Como se pode verificar, o valor máximo da memória persistente utilizada atinge-se na primeira salvaguarda do voto, ou seja, após o preenchimento do boletim. Neste instante é armazenada toda a informação relativa ao estado da votação. Porém, assim que as assinaturas requeridas são obtidas, alguma desta informação (e.g. os factores de cegamento) deixa de ser necessária, não sendo armazenada em memória. É este facto que justifica a diminuição do espaço ocupado persistentemente pelo voto à medida que o processo de votação se aproxima do fim.

Tendo em conta a memória disponível nos telemóveis de hoje em dia, bem como a crescente utilização de cartões de memória, os valores apresentados na Tabela 2 não constituem um factor limitativo na utilização do sistema.

6.2.3. Desempenho

O desempenho da aplicação é outro dos factores críticos do sistema. Conforme foi referido anteriormente, o protocolo de votação exige a cifra de dados, a assinatura dos Administradores, comunicações remotas, entre outras operações de poder computacional elevado.

Na Tabela 3 são apresentados os tempos de execução das diferentes etapas de um processo de votação para os telemóveis Nokia 6600 e Sony Ericsson P900. É usada como referência uma eleição com dois Administradores, um Contador e um boletim de voto com 1 KB. Os valores da tabela expressam-se em milissegundos.

Telemóvel	# assinaturas requeridas	Distribuidor		Administrador	Anonimizador / Contador	Total
		Lista de eleições	Obtenção do boletim	Assinatura	Submissão	
Nokia 6600	**1**	196	281	18510	16500	35487
	2	212	515	28953	16266	45946
Sony Ericsson P900	**1**	32	46	8016	7985	16079
	2	16	47	10281	6890	17234

Tabela 3 - Tempos de execução das diferentes etapas do processo de votação

Existe uma diferença significativa entre os valores obtidos nos dois telemóveis, justificada pelas diferentes características do *hardware* de cada um. O Sony Ericsson P900 possui maior poder computacional que o Nokia 6600, facto que permite que a realização das operações de votação sejam mais rápidas.

Por outro lado, é possível verificar que o aumento do número de assinaturas requeridas não incute um aumento proporcional do tempo de execução da fase de obtenção de assinaturas. Ou seja, de acordo com a tabela duas assinaturas são obtidas em menor tempo que o dobro da obtenção de uma. Esta performance deve-se à utilização de *threads* na comunicação com os Administradores (ver Secção 5.1.2).

6.2.4. Comunicações

O tráfego de dados em comunicações remotas é um dos factores mais importantes para os utilizadores de aplicações móveis, já que são normalmente taxados pelos operadores de telecomunicações móveis.

Para avaliar este requisito configurou-se uma eleição de referência composta por um boletim de voto com 1 KB, um único Contador e exigindo a assinatura de dois Administradores. Os resultados estão expressos na Tabela 4, em termos de dados enviados e recebidos.

Servidor	Etapa	Dados enviados (bytes)	Dados recebidos (bytes)
Distribuidor	**Lista de eleições**	29	103
	Obtenção do boletim	33	1490
Administrador	**1ª Assinatura**	164	134
	2ª Assinatura	164	134
Anonimizador / Contador	**Submissão**	576	2
		966	**1863**
	TOTAL	**2829**	

Tabela 4 - Tráfego de dados em cada etapa do processo de votação

O tráfego total dispendido no processo de votação, para a eleição acima referida, foi de aproximadamente 2,8 KB. Na comunicação com o Distribuidor de Boletins os dados recebidos são variáveis, dependendo do número de eleições disponíveis e do tamanho do boletim de voto. O mesmo acontece com a comunicação com os Anonimizadores/Contadores, onde a quantidade de dados varia com o tamanho do boletim. Relativamente aos Administradores, são trocados 298 bytes por cada assinatura requerida. Estes dados não variam com o tamanho do boletim de voto, já que apenas é enviado um resumo do voto (tamanho fixo).

Em suma, a quantidade de dados trocados vai variar com o tamanho do boletim de voto, o número de assinaturas requeridas e o número total de Administradores e Anonimizadores/Contadores.

Tendo em conta as tarifas actuais dos operadores de telecomunicações móveis portugueses os custos associados ao tráfego gerado no sistema são bastante aceitáveis. O custo da transferência de dados é taxado aos conjuntos de 10 KB de informação, de acordo com os seguintes valores:

- TMN: 0,005€/KB
- Vodafone: 0,0024€/KB
- Optimus: 0,0025/€KB

Para a eleição de referência indicada anteriormente, com dois Administradores, onde são transferidos 2,83 KB de dados, tem-se um custo associado entre 0,024€ (Vodafone) e 0,05€ (TMN), visto não se ter ultrapassado o limite de 10 KB de informação. Rapidamente se conclui que em termos monetários o custo da comunicação via GPRS/UMTS é comportável, não se tornando um entrave à utilização do sistema.

7. Conclusões e Trabalho Futuro

O objectivo do projecto MobileREVS consistiu no desenho e implementação de um sistema de votação electrónica que suporte utilizadores com telemóveis.

Este sistema teve como base o sistema REVS, desenhado para a Internet, e que providencia as características desejadas de um sistema de votação tradicional, como a democracia, privacidade, correcção e verificabilidade (ver Secção 1.1.2). O REVS, além de oferecer as propriedades descritas dos sistemas de votação, pretende providenciar um sistema robusto em termos de disponibilidade, capacidade de continuação diferida e resistência ao conluio (ver Secção 3.1).

A análise efectuada ao REVS e à tecnologia associada aos telemóveis permitiu-nos concluir que seria possível desenvolver o sistema MobileREVS com toda a robustez e segurança do REVS. No entanto, para o desenvolvimento deste sistema, surgiram novos requisitos: mobilidade, espaço ocupado pela aplicação, desempenho e o consumo de energia. Estes necessitam de cuidados acrescidos para que, na prática, o protocolo eleitoral não seja afectado (ver Secção 4.1). Das propriedades dos sistemas de votação electrónica, a privacidade é o requisito que levantou mais problemas devido às escassas técnicas criptográficas oferecidas de base pela plataforma J2ME. Desta forma, foi necessário recorrer à utilização de bibliotecas externas, no caso a Bouncy Castle (*Light Edition*). Foi, também, realizada uma análise a outros mecanismos de segurança que podem ser utilizados em versões posteriores do sistema MobileREVS, como a *Optional Package* SATSA e a utilização das funções criptográficas dos *smart cards* (ver Secção 2.3.3).

O desenho e a implementação do sistema MobileREVS contemplou a criação de uma aplicação para telemóveis, assim como o desenvolvimento de *servlets* para interpretação dos pedidos dos telemóveis e algumas adaptações dos servidores REVS. Destas adaptações destaca-se, principalmente, a adição da autenticação nos Distribuidores de Boletins, visto que a inexistência da mesma se revelou um problema em situações onde os boletins continham informações restritas ou confidenciais.

Adicionalmente, foi implementado um editor gráfico de boletins para o REVS, algo que não existia até ao momento. A aplicação, executável a partir de qualquer PC com a .NET Framework instalada, permite criar novos boletins assim como editar boletins existentes a partir dos respectivos ficheiros XML.

Desta forma, pode-se afirmar que os objectivos do projecto MobileREVS foram alcançados. O sistema desenvolvido oferece uma solução de votação viável e segura para telemóveis, sem abdicar das propriedades requeridas para um processo de votação electrónica (ver Secção 1.1.2).

Relativamente ao sistema MobileREVS podem-se considerar três desenvolvimentos como trabalho futuro: (1) a implementação de uma *release* para utilização da *Optional Package* SATSA; (2) a implementação de uma *release* para utilização da *Optional Package* FileConnection; e (3) a criação de um servidor de entrada no sistema MobileREVS que facilite o escalonamento dos pedidos para os Distribuidores de Boletins.

A implementação da *release* para utilização da *Optional Package* SATSA permitirá o acesso a funções de segurança existentes nos telemóveis, facilitando o acesso a elementos seguros, como os *smart cards*, e a diminuição do tamanho da aplicação. No entanto, esta *release* não foi considerada determinante para este projecto

por três razões: (i) por um lado ainda existem muito poucos telemóveis com esta *Optional Package* na instalação de raiz do J2ME; (ii) por outro lado, o acesso das aplicações aos *smart cards* (no caso, os cartões SIM) não é facilitado pelos operadores de telecomunicações móveis; (iii) finalmente, o tamanho total da aplicação actual, com a biblioteca externa Bouncy Castle (*Light Edition*), não excede o limite máximo considerado aceitável para uma aplicação deste género pela PT Inovação, a entidade patrocinadora deste projecto.

A implementação da *release* para utilização da *Optional Package* FileConnection permitirá usufruir do armazenamento de dados em dispositivos de memória externos, como é o caso dos cartões de memória. Dotar o sistema desta capacidade trará a vantagem de tornar possível a continuação diferida da votação em diferentes telemóveis, usando para tal o cartão de memória com os dados armazenados.

Actualmente a comunicação com os Distribuidores de Boletins é realizada de modo directo com apenas um servidor, sendo necessário um mecanismo de *clustering web* caso se pretenda que os pedidos sejam distribuídos para diferentes Distribuidores de Boletins. A criação de um servidor de entrada para o sistema MobileREVS poderá ser uma opção para o escalonamento dos diferentes pedidos para os Distribuidores de Boletins. Desta forma, obtém-se um melhor balanceamento da carga em ambientes com um número elevado de eleitores. Esse servidor poderia, também, ser usado para mediar toda a comunicação entre o telemóvel e os restantes servidores *web*.

Considera-se, ainda, como desenvolvimento futuro para o "REVS Ballot Editor" a sua integração na aplicação gráfica do Comissário. Assim, todas as operações do Comissário encontrar-se-ão centralizadas na mesma aplicação. O "REVS Ballot Editor" poderá ser usado nas situações em que a criação de boletins de voto esteja a cargo de outros intervenientes que não o Comissário.

8. Referências

[BM02] M. Burmester and E. Magkos. *Towards Secure and Practical E-Elections in The New Era*. Pages 8-9, 2002.
http://thalis.cs.unipi.gr/~emagos/overview_voting_2002.pdf

[CC97] L. Cranor and R. Cytron. *Sensus: A Security-Conscious Electronic Polling System for the Internet*. In Proceedings of the Hawaii International Conference on System Sciences. Wailea, Hawaii, 1997.

[Chaum81] David Chaum. *Untraceable electronic mail, return addresses, and digital pseudonyms*. Communications of the ACM, 24(2):84-88, 1981.

[Chaum82] David Chaum. *Blind signatures for untraceable payments*. Rivest R. L. Chaum, D. and A. T. Sherman, editors. In Proceedings of Advances in Cryptology - CRYPTO '82 (New York), pages 199-203. Plenum Press, 1982.

[CV] CyberVote. http://www.eucybervote.org/main.html

[EA] *Encryption Algorithms*. MyCrypto.net.
http://www.mycrypto.net/encryption/crypto_algorithms.html

[ECPR] *Elisa distributes SIM Card with Citizen Certificate*, Corporate Press Release. Elisa. http://www.elisa.com/english/index.cfm?t=7&o=7120.00&did=12012

[ENEC] Estonian National Electoral Committee. http://www.vvk.ee/engindex.html

[ERS] Electoral Reform Society.
http://www.electoral-reform.org.uk/publications/briefings/briefings.htm

[FN] *Nokia 6600 Device Details*. Forum Nokia. June 16, 2003.
http://www.forum.nokia.com/devices/6600

[GC] Generalitat de Catalunya, Eleições 2003.
http://www.gencat.net/governacio-ap/eleccions/e-votacio.htm

[Giguere02a] Eric Giguère. *Understanding the Connected Limited Device Configuration (CLDC)*. July 30, 2002.

[Giguere02b] Eric Giguère. *Understanding the Mobile Information Device Profile (MIDP)*. August 28, 2002.

[GMR85] S. Goldwasser, S Micali, and C. Rackoff. *The knowledge complexity of interactive proofs*. In Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing, pages 291-305, May 6-8 1985.

[GY02] Guang Yang. *Build Your Own Lightweight XML DOM Parser*. DevX.com. November 22, 2002. http://www.devx.com/xml/Article/10114

[ISCSE] *Integrating the SIM Card into J2ME as a Security Element*, White-Paper. Gemplus S.A., April 2005.

[JCP] *Java Community Process Overview*. http://jcp.org/en/introduction/overview

[JJR02] M. Jakobsson, A. Juels, and R. L. Rivest. *Making mix nets robust for electronic voting by randomized partial checking*. In Proceedings of the 11th USENIX Security Symposium (SECURITY-02), pages 339-353, Berkeley, CA, USA, August 5-9 2002. USENIX Association.

[JMS] *MIME and SMIME: Multipurpose Internet Mail Extensions and Secure MIME*. Javvin. http://www.javvin.com/protocolMIME.html

[JSR68] *Java 2 Platform, Micro Edition (J2ME); JSR 68 Overview*. Sun Microsystems. http://java.sun.com/j2me/overview.html

[JZF03] R. Joaquim, A. Zúquete and P. Ferreira. *REVS - A Robust Electronic Voting System*. IADIS International Journal of WWW/Internet. Vol. 1, N. 2. December 2003.

[LBC] *The Legion of the Bouncy Castle*. http://www.bouncycastle.org/

[MSOA01] A. Monteiro, N. Soares, R. Oliveira e P. Antunes. *Sistemas Electrónicos de Votação*. FCT/UL Technical Report DI-FCULTR-01-9, Outubro de 2001. http://www.di.fc.ul.pt/tech-reports/01-9.pdf

[MY01] David M'Raihi and Moti Young. *E-Commerce Applications of Smart Cards*. Gemplus S.A. Publications, 2001.

[Ortiz05] C. Enrique Ortiz. *The Security and Trusted Services API for J2ME*. Sun Microsystems, March 2005. http://developers.sun.com/techtopics/mobility/apis/articles/satsa1

[PKC] *Introduction to Public-Key Criptography*. Sun Microsystems. http://docs.sun.com/source/816-6154-10/contents.htm#1041372

[PS] *SmartXA – Advanced 16 bit Smart Card Controller*. Philips. http://www.semiconductors.philips.com/products/identification/smartxa/

[RA] *RSA Algorithm*. DI Management Services. http://www.di-mgt.com.au/rsa_alg.html

[SB] *Class BigInteger*. Sun Microsystems. http://java.sun.com/j2se/1.4.2/docs/api/java/math/BigInteger.html

[SCE] *The Smart Card Engine*. Gemplus S.A.. http://www.gemplus.com/techno/chipware

[SCM] *Smart Card: Manufacturers*. Wikipedia. http://en.wikipedia.org/wiki/Smart_card#Manufacturers_of_smart_cards

[Schilcher04] Fabian Schilcher. *Key Management and Distribution for Threshold Cryptography Schemes*. January 19, 2004.

[SKC] *Secret-Key Cryptosystems*. SSH Communications Security. http://www.ssh.com/support/cryptography/algorithms/symmetric.html

[SS] *Secure Sockets Layer*. SearchSecurity.com Definitions. http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci343029,00.html

[UKD] UK-edemocracy. http://www.edemocracy.gov.uk

[WEGE] *ElGamal Encryption*. Wikipedia. http://en.wikipedia.org/wiki/Elgamal

[WEI] *What is Electronic Identity*. Population Register Center. http://www.vaestorekisterikeskus.fi/vrk/home.nsf/pages/8339879C062961E3C2256C93003BC577

[WNM] *What's New in MIDP 2.0*. Sun Microsystems. http://java.sun.com/products/midp/whatsnew.html

[WX] *X.509*. Wikipedia. http://en.wikipedia.org/wiki/X.509

9. Glossário

3DES
Triple DES

AC
Autoridade de Certificação

APDU
Application Protocol Data Unit

API
Application Programming Interface

CDC
Connected Device Configuration

CLDC
Connected Limited Device Configuration

CPU
Control Processing Unit

CSR
Certificate Signing Request

DES
Data Encryption Standard

DOM
Document Object Model

EEPROM
Electrically Erasable and Programmable ROM

GCF
Generic Connection Framework

GPRS
General Packet Radio Service

GSM
Global System for Mobile Communications

HTTP
HyperText Transfer Protocol

HTTPS
Secure HyperText Transfer Protocol

J2EE
Java 2 Enterprise Edition

J2ME
Java 2 Micro Edition

J2SE
Java 2 Standard Edition

JCRMI
Java Card Remote Method Invocation

JVM
Java Virtual Machine

KVM
KiloByte Virtual Machine

LCDUI
Liquid Crystal Display User Interface

MIDP
Mobile Information Device Profile

MIME
Multipurpose Internet Mail Extension

PDA
Personal Digital Assistant

PIN
Personal Identification Number

REVS
Robust Electronic Voting System

RAM
Random Access Memory

RMI
Remote Method Invocation

RMS
Record Management Store

RNG
Random Number Generator

ROM
Read-Only Memory

RPC
Randomized Partial Checking

RSA
Rivest Shamir Adleman

SATSA
Security and Trusted Services API

SHA-1
Secure Hash Algorithm 1

SIM
Subscriber Identity Module

SMS
Short Message Service

SRAM
Static RAM

SSL
Secure Sockets Layer

SVE
Sistema de Votação Electrónica

S/MIME
Secure Multipurpose Internet Mail Extensions

TLS
Transport Layer Security

UAProf
User Agent Profile

UICC
Universal Integrated Circuit Card

UMTS
Universal Mobile Telecommunications System

USIM
Universal Subscriber Identity Module

WTLS
Wireless Transport Layer Security

WAP
Wireless Application Protocol

XML
Extensible Markup Language

Anexo A. Avaliação dos mecanismos de segurança para telemóveis

O sistema MobileREVS tem como objectivo providenciar, ao processo de votação através de telemóveis, toda a robustez e segurança associada ao sistema REVS. O protocolo do MobileREVS requer a utilização dos algoritmos criptográficos 3DES, RSA e SHA-1. Por isso, as soluções indicadas na Secção 2.3.3 são analisadas em termos dos algoritmos providenciados e das características dos mesmos (e.g. o tamanho suportado para as chaves das cifras). É feita excepção ao MIDP já que, ao não oferecer funcionalidades criptográficas de suporte ao cumprimento do protocolo do MobileREVS, não pode ser usado independentemente no desempenho da segurança do sistema, sendo necessário recorrer a outros mecanismos.

1. Módulo *Crypto* dos *smart cards*

Após alguma investigação foi possível determinar que este módulo dos *smart cards* implementa vários algoritmos criptográficos, incluindo os necessários para o projecto: 3DES, RSA e SHA-1. Porém, visto se desconhecer quais os cartões utilizados actualmente pelas operadoras de telecomunicações móveis em Portugal, não foi possível determinar as operações disponibilizadas pelos mesmos nem o seu desempenho, já que se encontram inevitavelmente associadas ao *smart card* utilizado. Alguma informação sobre um processador de *smart card* de referência da Philips pode ser encontrada na Secção 2.3.2.

2. SATSA

Tipo	Algoritmo	Tamanho das chaves (bits)	Tamanho dos blocos (bits)	Output (bits)
Cifra simétrica	DES/3DES	DES: 56 3DES: 112, 168	64	Múltiplo de 64
Resumo	SHA-1	*[não se aplica]*	*[não se aplica]*	160

Tabela 1 – Principais algoritmos e respectivas características da *Optional Package* SATSA

Conforme descrito na Tabela 1, os únicos algoritmos criptográficos oferecidos ao programador pela *Optional Package* SATSA são o DES/3DES e o SHA-1. Na SATSA as funções de cifra assimétrica RSA encontram-se embebidas apenas no contexto da criação e verificação de assinaturas, pelo que as operações básicas de cifra e decifra não estão disponíveis. Deste modo, definitivamente, não é possível cumprir com o protocolo de segurança do REVS para a criação de assinaturas cegas. A solução para controlar este problema requer a utilização de funções RSA externas, disponibilizadas por uma das outras soluções, ou mesmo uma implementação de assinaturas cegas ao nível da aplicação.

Por outro lado, a SATSA, sendo uma *Optional Package* do J2ME, necessita de vir instalada de raiz nos dispositivos pelos seus fabricantes, algo que não é comum nos dias de hoje. Este torna-se, assim, o factor mais limitativo da sua utilização.

3. Bouncy Castle (*Light Edition*)

Tipo	Algoritmo	Tamanho das chaves (bits)	Tamanho dos blocos (bits)	Output (bits)
Cifra simétrica	DES/3DES	DES: 56 3DES: 112, 168	64	Múltiplo de 64
Cifra assimétrica	RSA	até 2048 (múltiplos de 8)	Igual ao tamanho do módulo da chave	Igual ao tamanho do módulo da chave
Resumo	SHA-1	*[não se aplica]*	*[não se aplica]*	160

Tabela 2 – Principais algoritmos e respectivas características da biblioteca Bouncy Castle (*Light Edition*)

A biblioteca Bouncy Castle (*Light Edition*) providencia todos os algoritmos criptográficos necessários para manter o protocolo de segurança definido. A Tabela 2 resume as principais características de cada um deles.

Na Tabela 3 encontram-se sintetizados os algoritmos disponibilizados por cada solução, necessários para a implementação do MobileREVS. Como é possível verificar, apenas a biblioteca Bouncy Castle (*Light Edition*) oferece uma implementação das assinaturas cegas (**), essenciais para o cumprimento do protocolo do MobileREVS. Porém, essa implementação, estando ao nível da aplicação, terá de acompanhar necessariamente o *MIDlet*. Por outro lado, a *Optional Package* SATSA não possibilita a utilização da função de RSA directamente, permitindo o seu uso apenas no contexto da criação de assinaturas digitais. Desta forma, será necessário recorrer a outro mecanismo para a realização das assinaturas cegas.

	Operações elementares			Assinaturas	
	3DES	RSA	SHA-1	Digitais	Cegas
Módulo *Crypto* dos *smart cards*	✓	✓	✓	✓(*)	
SATSA	✓		✓	✓	
Bouncy Castle (*Light Edition*)	✓	✓	✓	✓	✓(**)

Tabela 3 – Resumo dos algoritmos disponibilizados por cada solução
(*) apenas para alguns *smart cards*
(**) embora não estejam incluídas de raiz na biblioteca Bouncy Castle (*Light Edition*) já existe uma implementação de assinaturas cegas com recurso a esta biblioteca

4. Testes realizados

Posteriormente à análise dos algoritmos disponibilizados por cada solução, foram realizados testes de *performance* e de consumo de memória, de maneira a

determinar qual a solução que melhor se adequa ao sistema MobileREVS. Seguidamente, encontram-se apresentados os resultados dos testes sobre diversas plataformas e ambientes em que foram executados. Note-se que a avaliação da utilização do módulo *Crypto* dos *smart cards* não foi submetida a estes testes, visto não possuirmos informação suficiente sobre os cartões actualmente utilizados.

Os testes foram divididos em duas categorias: (1) testes realizados sobre os simuladores, no computador onde o *MIDlet* foi desenvolvido; (2) testes realizados em dispositivos móveis concretos. Os testes incidiram sobre a determinação do tempo de execução das funções criptográficas e do espaço ocupado pelo *MIDlet* de teste na memória do dispositivo móvel.

De modo a cumprir os objectivos definidos foi desenvolvido um simples *MIDlet* de teste que executa uma operação de cifra e decifra de uma mensagem, através do algoritmo 3DES e do RSA. Para o teste do algoritmo SHA-1 foi criado um resumo dessa mesma mensagem e comparado com o resultado esperado, de modo a verificar a sua validação. Escolheu-se, como exemplo de teste, uma mensagem de 50 bytes. Embora pudesse ter outro tamanho consideramos que este é um valor perfeitamente aceitável para uma avaliação dos mecanismos em termos relativos. Para se obter um valor mais preciso acerca do tempo de execução das funções criptográficas, cada teste consistiu em 20 iterações consecutivas com determinação da média dos tempos. O *MIDlet* de teste foi desenvolvido nas configurações CLDC 1.0 e MIDP 2.0, visto estas serem as configurações mais utilizadas hoje em dia nos telemóveis.

Biblioteca	**Espaço total ocupado (KB)**
SATSA	2
Bouncy Castle (*Light Edition*)	24

Tabela 4 – Memória ocupada pelo *MIDlet* de teste para cada uma das soluções

Na Tabela 4 pode-se verificar o espaço total ocupado na memória pelo *MIDlet* de teste, para cada uma das soluções propostas. A utilização da *Optional Package* SATSA diminui consideravelmente esse espaço, já que as funções criptográficas encontram-se embebidas no próprio telemóvel, não acompanhando o *MIDlet*.

No entanto, é importante referir que, como a *Optional Package* SATSA não implementa o algoritmo RSA, não se procedeu ao teste desta função criptográfica, daí haver um decréscimo do espaço ocupado. Porém, se juntarmos a esta *Optional Package* SATSA as classes da biblioteca Bouncy Castle (*Light Edition*) que lidam com RSA, o espaço ocupado em memória seria superior.

Finalmente, as funções criptográficas RSA da biblioteca Bouncy Castle (*Light Edition*) baseiam-se na implementação do tipo de dados `BigInteger`, não suportado no ambiente de execução Java dos telemóveis actuais. Este tipo de dados permite a manipulação de números inteiros da mesma forma que o tipo `Integer`, lidando com números arbitrariamente grandes e oferecendo operações adicionais de aritmética modular, geração de números primos e manipulação de bits, entre outras [SB].

4.1. Simulador J2ME com a *Optional Package* SATSA

O *MIDlet* de teste foi avaliado usando um simulador de telemóveis equipados com J2ME (Sun Wireless Toolkit), cuja arquitectura é composta pelo CLDC 1.0, MIDP 2.0 e a *Optional Package* SATSA. A máquina em que foi executado o simulador é um

Pentium 4 a 3,2GHz, com 1GB de memória RAM. Os resultados desse teste são apresentados na Tabela 5.

Biblioteca	Tempo de execução unitário (ms)
Bouncy Castle (*Light Edition*)	3DES: 2 RSA: 375 SHA-1: menos de 1
SATSA	3DES: 2 SHA-1: menos de 1

Tabela 5 – Resultado dos testes no simulador J2ME

Verifica-se que, a nível do simulador, tanto as funções criptográficas da SATSA como do Bouncy Castle (*Light Edition*) possuem uma *performance* relativa muito semelhante.

Após a execução do *MIDlet* de teste no simulador J2ME, o mesmo foi executado em dispositivos concretos, nomeadamente nos telemóveis Sony-Ericsson P900 e Nokia 6600. Os resultados obtidos dessas execuções encontram-se expressos na Tabela 6 e na Tabela 7, respectivamente.

4.2. Sony-Ericsson P900

Biblioteca	Tempo de execução unitário (ms)
Bouncy Castle (*Light Edition*)	3DES: 17 RSA: 620 SHA-1: 3
SATSA	*[não se encontra implementada]*

Tabela 6 – Resultado dos testes no dispositivo Sony-Ericsson P900

Este dispositivo não possui uma implementação de raiz da *Optional Package* SATSA, pelo que não foi possível executar o *MIDlet* de teste respectivo para comparação dos tempos de execução.

4.3. Nokia 6600

Biblioteca	Tempo de execução unitário (ms)
Bouncy Castle (*Light Edition*)	3DES: 24 RSA: 948 SHA-1: 6
SATSA	*[não se encontra implementada]*

Tabela 7 – Resultado dos testes no dispositivo Nokia 6600

Tal como o anterior, este dispositivo não possui uma implementação de raiz da *Optional Package* SATSA, pelo que não foi possível executar o *MIDlet* de teste respectivo.

Apesar de não ser possível avaliar os resultados da execução do *MIDlet* de teste em SATSA nos telemóveis anteriores, foi possível determinar que a função criptográfica RSA é a que consome mais tempo de execução em relação às funções 3DES e SHA-1, conforme já era esperado (ver Tabelas 5, 6 e 7).

5. Conclusões

A investigação e análise dos mecanismos de segurança apresentados permitiram obter as conclusões que se seguem.

Relativamente à biblioteca Bouncy Castle (*Light Edition*), esta disponibiliza todos os algoritmos de segurança requeridos para o projecto MobileREVS. Como é incluída no *MIDlet suite* oferece portabilidade à aplicação ao não ter dependências com os ambientes J2ME dos telemóveis. Porém, todas as funcionalidades são realizadas ao nível da aplicação, não havendo qualquer aproveitamento das funcionalidades de segurança oferecidas pelos *smart cards*. Outra das desvantagens da utilização desta biblioteca consiste no aumento significativo do tamanho do *MIDlet*.

A *Optional Package* SATSA não permite a utilização das operações de cifra e decifra RSA directamente. Estas, apesar de implementadas, são apenas utilizadas no contexto da criação e verificação de assinaturas digitais. Esta limitação tem algumas implicações no desenvolvimento do projecto MobileREVS, principalmente na manipulação de assinaturas cegas que, definitivamente, deixam de ser possíveis de se realizar. Por outro lado, a *Optional Package* SATSA disponibiliza uma API de comunicação com o cartão, algo que oferece uma mais valia em termos de segurança da informação do processo de eleição. A sua utilização traria algumas vantagens de segurança ao nível do armazenamento seguro e da manipulação de dados sensíveis, pois seriam realizados dentro das barreiras de segurança do cartão. Conforme foi descrito na Secção 2.3.3, outra das vantagens que adviria da utilização das funcionalidades criptográficas do cartão seria a diminuição do espaço total ocupado pelo *MIDlet* no telemóvel. Além disso, a utilização da cifra RSA do cartão (U)SIM permitiria contornar o problema, inicialmente referido, da impossibilidade da utilização de operações RSA directamente ao nível da aplicação, desde que também fossem disponibilizadas operações de aritmética modular. Outra solução, seria a inclusão de funções externas no *MIDlet*, como por exemplo as funções de RSA do Bouncy Castle (*Light Edition*). O principal problema da utilização da *Optional Package* SATSA é que esta só está disponível se os fabricantes desses dispositivos a incluírem na implementação J2ME do telemóvel, o que não é muito usual actualmente. No caso dos telemóveis onde foram desenvolvidos os testes, esta *Optional Package* não se encontra presente.

Como nota importante, o facto de alguns dos mecanismos apresentados oferecerem uma implementação do algoritmo RSA não é suficiente para a realização de assinaturas cegas. Isto porque são também necessárias operações de aritmética modular de grandes números para gerar assinaturas cegas, algo que nem sempre está disponível nas implementações fornecidas. Em todo o caso, a presença de uma implementação do algoritmo RSA é um ponto a favor já que permite uma autenticação mais segura por parte dos eleitores.

Finalmente, não foi possível determinar quais os cartões (U)SIM usados pelas operadoras de telecomunicações móveis portuguesas. Seria importante obter informação sobre o modo de acesso às funções criptográficas oferecidas pelos cartões para efeitos da exploração destas funcionalidades no desenvolvimento do projecto.

Anexo B. MobileREVS: Manual de Instalação e Configuração

1. Introdução

O presente documento serve para descrever o processo de instalação e configuração do sistema MobileREVS.

2. Requisitos do sistema

Para instalar e utilizar o sistema MobileREVS é necessário possuir um telemóvel com as seguintes características:

- J2ME com CLDC 1.0 e MIDP 2.0, ou posteriores
- Serviços de ligação à Internet activada (GPRS ou UMTS)

A execução dos servidores REVS requer a instalação prévia de outro software:

- Java Runtime Environment (JRE), na versão 1.5 ou posterior
  (disponível em http://java.sun.com)
- MySQL Server, na versão 5.0 ou posterior
  (disponível em http://www.mysql.com)

Adicionalmente, é necessário um contentor Java de páginas *web*. É vivamente aconselhada a utilização do contentor:

- Apache Tomcat, versão 5.5 ou posterior
  (disponível em http://tomcat.apache.org)

3. Servidores REVS

Depois da instalação do software anteriormente referenciado deve proceder à instalação dos servidores REVS. Se ainda não está familiarizado com a instalação e a configuração destes servidores é aconselhada a leitura do Manual de Instalação do REVS, disponível em http://www.gsd.inesc-id.pt/~revs. Caso contrário poderá seguir o guia de instalação rápida, apresentado de seguida.

3.1. Ficheiros de configuração

Para cada servidor (Distribuidor de Boletins, Administrador, Anonimizador e Contador) existe um ficheiro de configuração associado, `server.cfg`, com o seguinte conteúdo:

```
SERVER //<endereço_host >/<serviço_RMI>
DATABASE //<endereço_host>/<base_dados>
```

Exemplo para um Distribuidor de Boletins:

```
SERVER //localhost/distributor
DATABASE //localhost/distributor1
```

Estes ficheiros estão pré-configurados para iniciar os servidores do REVS e respectivas bases de dados localmente, ou seja, no endereço *localhost*. Caso pretenda instalá-los noutro endereço deve proceder à sua alteração.

> **NOTA:** A alteração do nome do serviço no campo "SERVER" exige a recompilação dos servidores e geração do ficheiro `revs_servers.jar`.

3.2. Configuração dos servidores

Para cada servidor REVS a instalar deve executar os seguintes passos:

- Criar a base de dados para o servidor em questão com um nome em conformidade com o ficheiro de configuração, descrito na secção anterior (no exemplo, "distributor1")
- Editar o ficheiro `start_server.bat` respectivo e alterar `<CURRENT_DIRECTORY>` para o directório onde o REVS foi instalado (exemplo: "`C:/revs`")

Seguidamente deve ser iniciado o serviço de registo de nomes RMI, digitando o comando "rmiregistry" na linha de comandos. É possível que o caminho para o serviço de registo de nomes RMI não esteja definido nas variáveis de ambiente. Nesse caso deverá executar o comando na subdirectoria "`\bin`" da directoria onde se encontra instalado o JRE.

A próxima etapa consiste na criação de uma conta de utilizador com privilégios de acesso às bases de dados do REVS. Se pretender saber mais informações sobre como adicionar contas de utilizadores à base de dados deve consultar a documentação providenciada em http://www.mysql.com. A título de exemplo são apresentados os passos para criar um utilizador com os dados utilizados por omissão no REVS e com privilégios de administração sobre qualquer base de dados existente no servidor MySQL:

- Abrir uma consola MySQL, entrando com privilégios de administração (por omissão, através do comando "`mysql -u root -p`")
- Executar o comando:

```
GRANT ALL PRIVILEGES ON *.* TO 'REVSuser'@'localhost'
IDENTIFIED BY 'REVSpassDB';
```

A partir deste momento já é possível iniciar cada um dos servidores REVS, executando o ficheiro `start_server.bat` de cada um deles.

Depois de executar o ficheiro anterior terá de indicar o tipo de servidor que está a configurar (Distribuidor de Boletins, Administrador, Anonimizador ou Contador). Deverá, então, introduzir o utilizador e palavra-passe de acesso à base de dados que criou para esse servidor. Os dados são "REVSuser" e "REVSpassDB" para utilizador e palavra-passe, respectivamente. Posteriormente, são-lhe pedidas as palavras-passe de

acesso à *KeyStore* e à *PrivateKey* do servidor. Para ambas deverá introduzir "REVSpass".

A partir deste momento surgirão as opções de menu para o servidor em questão. Note-se que na primeira utilização do servidor é necessário criar as tabelas na base de dados, escolhendo a opção "`C - Create database`". Depois resta apenas popular as tabelas com os dados da eleição, existentes no ficheiro gerado pelo Comissário da eleição, e iniciar o servidor. Para mais informações sobre este processo consulte http://www.gsd.inesc-id.pt/~revs.

4. MobileREVS

Os passos que se seguem descrevem o processo de instalação e configuração para os restantes componentes arquitecturais do MobileREVS. Numa primeira etapa será descrita a configuração do Apache Tomcat. Posteriormente, será apresentado o processo de instalação dos *servlets* e do Módulo Eleitor.

4.1. Configuração do Apache Tomcat

Depois de instalar o Apache Tomcat é necessário configurar a política de segurança associada ao contentor de páginas *web*. A mesma deve ser definida no ficheiro "`java.policy`" (pode encontrar um exemplo deste ficheiro na directoria "`\lib\security`" do JRE) podendo ser ajustada às necessidades de cada um.

Para que o Apache Tomcat possa reconhecer internamente este ficheiro deverá adicioná-lo à lista de opções Java já definidas:

```
-Djava.security.policy=<PATH>\java.policy
```

Em `<PATH>` deverá especificar o caminho para o ficheiro `java.policy`.

Caso o servidor Apache Tomcat já esteja a ser executado é necessário reiniciá-lo para que as alterações surtam efeito.

Seguidamente, e caso pretenda usar comunicações via HTTPS, deverá configurar o Apache Tomcat de modo a este as poder suportar. Para informações mais detalhadas sobre este processo consulte a documentação disponível em http://tomcat.apache.org.

4.2. Instalação dos *servlets*

Para instalar os *servlets* deverá, com o Apache Tomcat iniciado, abrir um browser de acesso à Internet e aceder a http://localhost:8080/manager/html. O porto 8080 é o pré-definido durante a instalação do Apache Tomcat. Caso o tenha alterado deve usar o porto escolhido por si. Os dados de acesso também são definidos durante o processo de instalação. Contudo, caso não os tenha alterado, o utilizador é "admin" e a palavra-passe é deixada em branco.

Depois de aceder à página de gestão do Apache Tomcat deverá instalar os *servlets* através do *upload* do ficheiro `REVS.war`.

4.3. Instalação do Módulo Eleitor

O Módulo Eleitor é composto por um único ficheiro, `MobileREVS.jar`. Este ficheiro pode ser instalado em qualquer telemóvel com os requisitos definidos na secção "Requisitos" deste documento. Para tal, deve utilizar qualquer um dos métodos de transferência de ficheiros oferecidos pelo telemóvel em questão: infravermelhos, *bluetooth*, cabo USB ou acedendo a um servidor *web* que contenha a aplicação.

Nalguns telemóveis poderá ser necessária a instalação prévia do certificado digital para os endereços remotos dos servidores (pelos mesmos meios acima indicados). No entanto, esta instalação é opcional à utilização do sistema, servindo apenas para oferecer maior segurança ao mesmo.

Anexo C. MobileREVS: Manual do Utilizador

1. Introdução

O presente documento serve para descrever na óptica do utilizador o funcionamento do Módulo Eleitor do sistema MobileREVS, a aplicação que se executa no telemóvel do eleitor.

2. Requisitos do sistema

A aplicação foi desenvolvida em J2ME (Java 2 Micro Edition), pelo que deve ser previamente instalada num telemóvel que possua a máquina virtual Java. A máquina virtual Java deve ser constituída pela *Configuration* CLDC na versão 1.0, ou superior, e pelo *Profile* MIDP na versão 2.0. O telemóvel deve, ainda, ter acesso à rede GPRS ou UMTS, normalmente fornecido por um operador de telecomunicações móveis.

3. Notas sobre a primeira utilização

A aplicação utiliza o protocolo HTTP/HTTPS para comunicação na rede. Para garantir a confidencialidade das comunicações é necessário instalar certificados válidos para os servidores do sistema MobileREVS com os quais o telemóvel comunica. Doutra forma, alguns dados transitarão em claro na rede, pelo que é desaconselhada a utilização sem certificados na transmissão de dados sensíveis. Em todo o caso, mesmo nesta situação, os dados relativos à votação do eleitor são sempre transmitidos confidencialmente pela rede.

4. Aplicação

Nesta secção será descrito o modo de utilização das funcionalidades oferecidas pela aplicação.

4.1. Autenticação e configuração do sistema

O primeiro ecrã a surgir após o arranque da aplicação é o ecrã de autenticação (ver Figura 1). Neste ecrã deve introduzir o seu número de eleitor e a palavra-passe, confirmando com a escolha do item “OK” do menu de navegação.

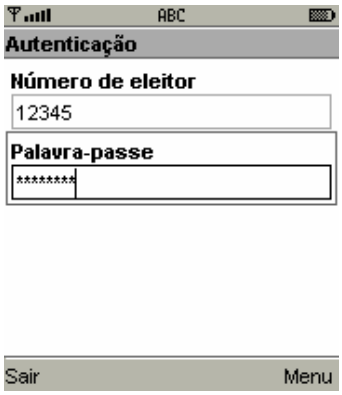

Figura 1 – Ecrã de autenticação

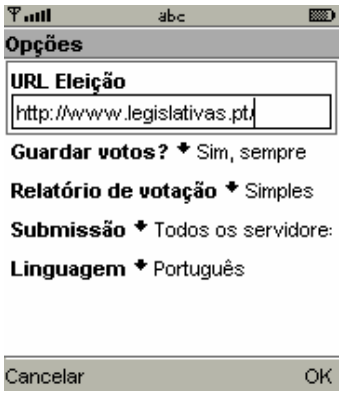

Figura 2 – Ecrã de opções

Também é possível alterar as configurações da aplicação através do item "Opções" (ver Figura 2). Neste ecrã poderá alterar as seguintes configurações:

- O URL público da eleição, ao qual o telemóvel se deverá ligar para proceder à sua votação. As eleições poderão não ser centralizadas, caso em que o URL mudará de eleição para eleição e deverá ser divulgado pela entidade responsável;
- Os votos poderão ser armazenados em memória persistente. Esta característica permitirá retomar a sua votação noutra altura, prevenindo assim que perca o seu boletim preenchido se os servidores estiverem ocupados, p.e.. Esta opção tem três escolhas: "Sim, sempre", para armazenar automaticamente o voto durante o processo de votação; "Não, nunca", para que o voto nunca ser armazenado em memória persistente; "Perguntar", para que escolha se pretende ou não armazenar o seu voto durante o processo de votação;
- O relatório de votação é apresentado no final de cada eleição em que participar. Nesta opção poderá definir se pretende um relatório simples, que indique se a votação foi bem sucedida, ou se pretende um relatório mais detalhado, com informação específica acerca do desenrolar do processo de votação no sistema. Este último requer conhecimento prévio acerca do funcionamento do sistema de votação electrónica;
- A configuração "Submissão" define se o envio do voto deverá ser realizado para todos os servidores de submissão disponíveis ou apenas para um. Neste último caso, o custo da ligação numa rede móvel taxada será inferior dado que a comunicação só será realizada com um dos servidores. Por outro lado, caso se sinta mais seguro com a entrega do seu voto a todos os servidores de submissão disponíveis deverá escolher "Todos os servidores";
- Em "Linguagem" poderá definir dinamicamente a linguagem apresentada pela aplicação. Na versão 1.0 estão apenas disponíveis as línguas Inglesa e Portuguesa.

4.2. Escolha de uma eleição

Depois de autenticado é apresentada a lista de eleições disponíveis, a decorrer no presente momento (ver Figura 3). O eleitor terá, então, de escolher a eleição onde

pretende participar, sendo apresentado o boletim respectivo para preenchimento (ver Secção 4.3.). Nesta lista são omitidas as eleições que verifiquem as seguintes características: (1) o eleitor já completou o processo de votação para essa eleição com sucesso numa outra ocasião; e (2) existe um voto armazenado na memória do telemóvel para essa eleição. Note-se que as eleições cujo processo de votação foi interrompido, e para as quais existe um voto armazenado na memória do telemóvel, surgem na lista. Porém, a escolha de uma destas eleições não apresentará o boletim respectivo, já ter sido preenchido em ocasião prévia. Ao invés será pedida uma confirmação para a retoma do processo de votação, conforme apresentado na Figura 4.

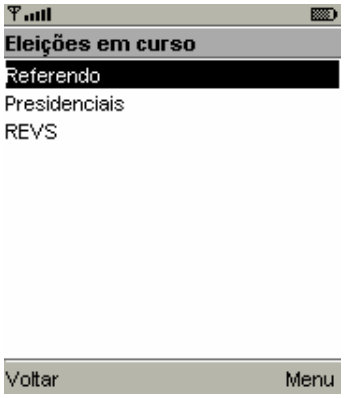

Figura 3 – Ecrã de eleições em curso

A opção "Votos Guardados" permite visualizar os votos armazenados persistentemente na memória do telemóvel. Através deste ecrã é possível visualizar informação específica de cada voto assim como eliminar aqueles que já não são pretendidos, libertando espaço na memória do telemóvel (ver Figura 5).

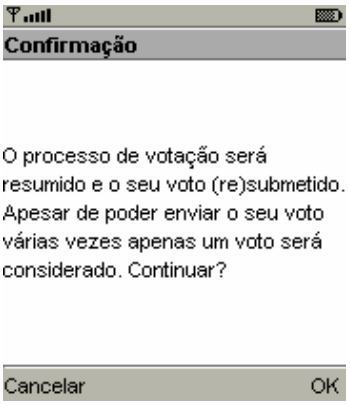

Figura 4 – Ecrã de retoma do processo de votação

Figura 5 – Ecrã de votos guardados

4.3. Preenchimento do boletim

Depois de escolhida a eleição é apresentada uma descrição sobre os pormenores da mesma. Escolhendo "OK" tem-se acesso ao preenchimento do boletim de voto, com a apresentação da primeira questão. O boletim de voto pode ser constituído por várias questões, sendo apresentada uma em cada ecrã. As opções "Próxima" e "Voltar" permitem, respectivamente, passar à próxima questão ou voltar à anterior. Na última questão, a opção "Próxima" dá lugar a "OK", para terminar o preenchimento do boletim.

O ecrã de cada questão é composto pelos seguintes elementos (ver Figura 6):

1. Número da questão
2. Descrição da questão
3. Respostas à questão

O número da questão permite identificar a ordem da questão no boletim de voto. Dependendo do tipo de eleição as questões podem estar agrupadas logicamente. Como tal, é um número composto por dois dígitos separados por um "." (ponto). O primeiro dígito indica o número do grupo; o segundo dígito indica o número da questão nesse grupo.

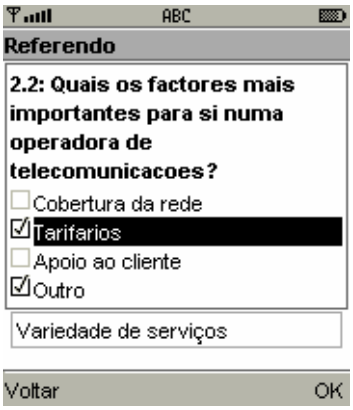

Figura 6 – Ecrã de resposta a uma questão

A questão apresentada poderá ser de resposta exclusiva ou múltipla. As questões de resposta exclusiva só permitirão a escolha de uma das respostas possíveis. As questões de resposta múltipla permitirão escolher qualquer uma das respostas apresentadas. A questão poderá permitir, também, uma resposta aberta. Neste caso, terá de seleccionar a última resposta ("Outro"), escrevendo a sua resposta na caixa de texto respectiva.

4.4. Confirmação das respostas

Ao terminar o preenchimento do boletim de voto será apresentado um ecrã de confirmação das suas respostas (ver Figura 7). Se pretender voltar atrás às questões respondidas deve escolher a opção "Voltar". Para confirmar as suas respostas bastará seleccionar a opção "OK", de forma a submeter o seu voto.

A partir deste ecrã é possível ter acesso a uma previsão das respostas escolhidas, através da opção "Previsão". No ecrã de previsão (ver Figura 8) são listadas as respostas escolhidas para cada questão do boletim. Se pretender alterar alguma das respostas poderá fazê-lo directamente através deste ecrã, seleccionando "Alterar" para a resposta pretendida. Seleccione "OK" para confirmar as alterações.

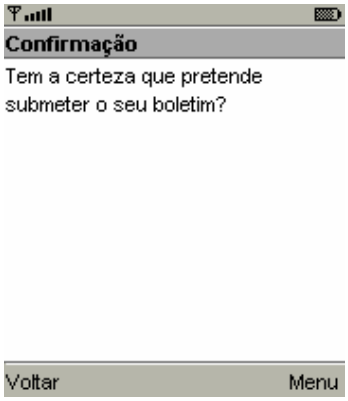

Figura 7 – Ecrã de confirmação

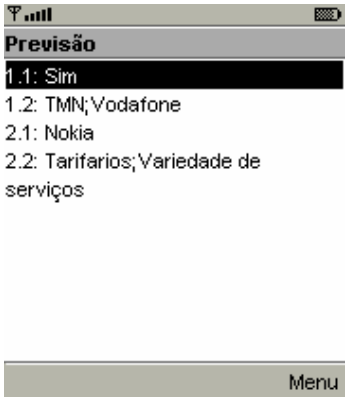

Figura 8 – Ecrã de previsão

4.5. Relatório da votação

Depois de submetido o voto é apresentado um relatório da votação. Este relatório poderá ser do tipo simples (ver Figura 9) ou detalhado (ver Figura 10), de acordo com o que estiver definido nas configurações da aplicação, ver Secção 4.1.

O relatório detalhado informá-lo-á do sucesso ou não do processo de votação. Por outro lado, o relatório detalhado descreverá com detalhe o contacto com todos os servidores do sistema. A sua correcta interpretação está dependente de um conhecimento mais pormenorizado da arquitectura do sistema MobileREVS.

Se a votação não for bem sucedida é-lhe oferecida a oportunidade de tentar novamente, através da opção "Re-submeter Voto". Seleccione "OK" para terminar a votação e voltar ao ecrã de eleições disponíveis.

Figura 9 – Ecrã de relatório de votação simples

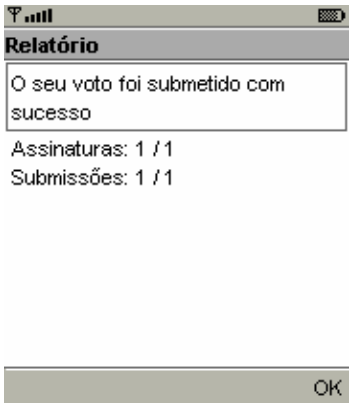

Figura 10 – Ecrã de relatório de votação detalhado

Anexo D. REVS Ballot Editor: Manual do Utilizador

1. Introdução

O presente documento serve para descrever na óptica do utilizador o funcionamento da aplicação "REVS Ballot Editor", um editor de boletins para o sistema electrónico de votação REVS.

2. Requisitos do sistema

A aplicação "REVS Ballot Editor" foi desenvolvida em C# e Windows Forms. Como tal, para a sua utilização é necessário ter instalado o sistema operativo Windows XP juntamente com a ".NET Framework", na versão 1.1 ou superior. A ".NET Framework" pode ser encontrada em http://www.microsoft.com/downloads/details.aspx?displaylang=en&FamilyID=262D25E3-F589-4842-8157-034D1E7CF3A3.

3. Notas sobre a primeira utilização

A aplicação é executada através do ficheiro `REVSBallotEditor.exe`. Este ficheiro deve estar localizado em qualquer directoria de um disco rígido, ou qualquer outro com permissões de escrita. Este requisito, embora não seja obrigatório para o funcionamento correcto da aplicação, é aconselhado devido ao facto de na mesma directoria do ficheiro executável ser armazenado um ficheiro de configurações com as preferências de utilização da aplicação.

Sempre que o ficheiro de configuração não exista na mesma directoria que o executável da aplicação (o que acontece na primeira utilização) é criado um novo com as opções pré-definidas para a aplicação. Assim, se por alguma razão pretender alterar a localização do executável e manter as mesmas configurações deve assegurar que o ficheiro de configurações também é movido para a nova localização.

A inexistência de permissões de escrita na directoria corrente não permite que as preferências de utilização de aplicação sejam armazenadas persistentemente. Desta forma, sempre que a aplicação for reiniciada serão assumidas as preferências por omissão.

4. Aplicação

Nesta secção será descrito o modo de utilização das funcionalidades oferecidas pela aplicação.

4.1. Ecrã inicial

O ecrã principal da aplicação (ver Figura 1) é composto por um menu de navegação contextual e três botões de funcionalidade.

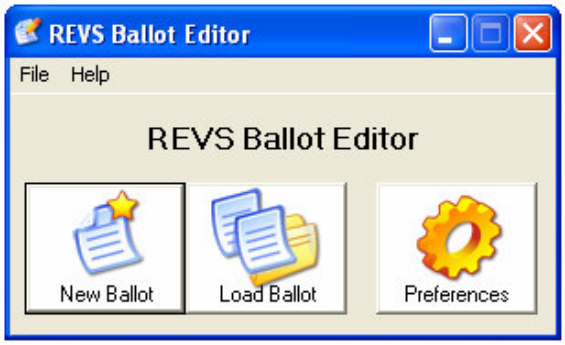

Figura 1 – Ecrã principal

Relativamente aos botões de funcionalidade:

- Em "New Ballot" é facilitada a criação de um novo boletim de voto de raiz;
- O botão "Load Ballot" permite carregar um boletim existente, a partir de um ficheiro em XML, para ser editado na aplicação;
- Através do botão "Preferences" é possível personalizar as opções de configuração da aplicação.

O menu contextual contém dois separadores (ver Figura 2): (1) o separador "File", que contém acesso às funcionalidades acima descritas e a opção "Exit" para sair da aplicação; e (2) o separador "Help". Neste último separador existem as opções: (i) "How to use", uma ajuda básica sobre as funcionalidades principais da aplicação; (ii) "Report bug", que facilita a submissão de bugs detectados pelos utilizadores; e (iii) "About", que apresenta características relativas à aplicação e ao seu desenvolvimento.

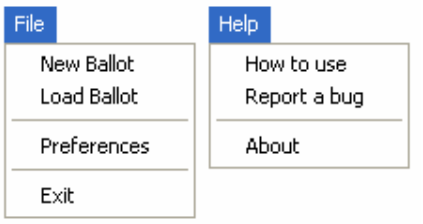

Figura 2 – Separadores do menu contextual

4.2. Editor de boletins

O editor de boletins pode ser invocado em duas situações: quando se pretende criar um novo boletim ou editar um boletim existente. Ambas as funcionalidades são semelhantes, com a diferença de que na edição de um boletim existente o editor de boletins encontra-se à partida preenchido com os dados do boletim a alterar.

O editor de boletins é constituído por quatro grandes áreas, designadas pelas letras de A a D na Figura 3.

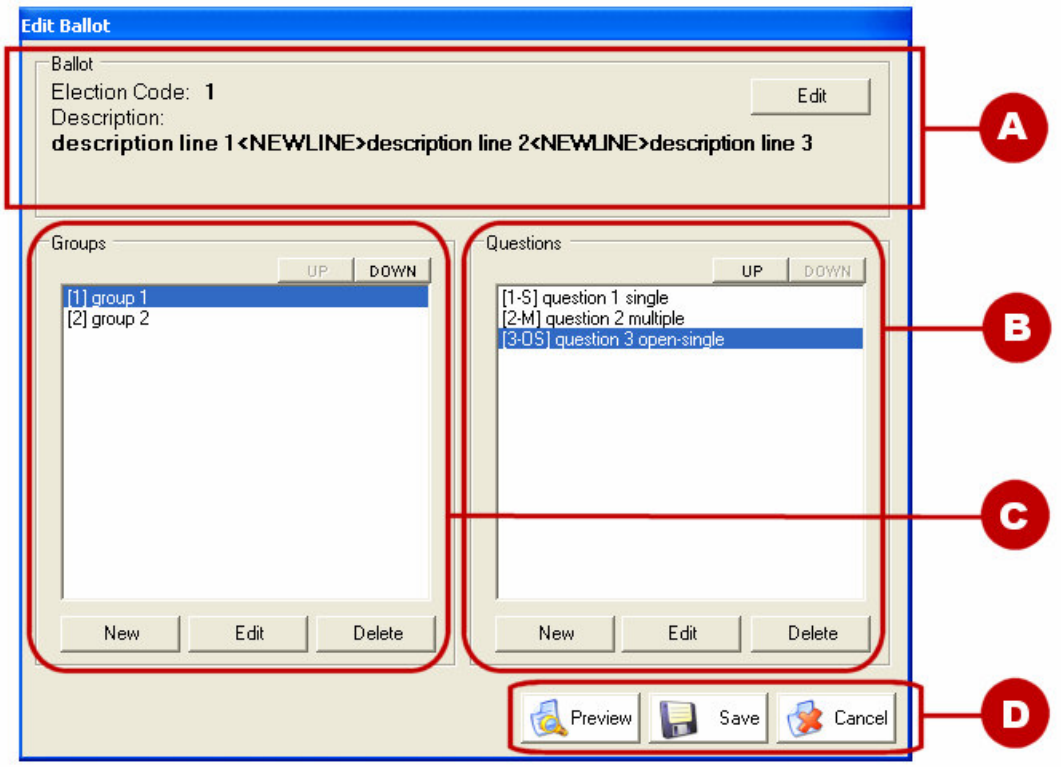

Figura 3 – Editor de boletins

A. Esta área refere-se às informações gerais do boletim, nomeadamente a sua descrição e o código da eleição a que pertence, que podem ser alteradas através do botão "Edit";

B. Nesta área a caixa "Questions" revela as questões relativas ao grupo seleccionado em C, facilitando a sua adição, edição e remoção através dos botões "New", "Edit" e "Delete", respectivamente. Cada entrada é constituída pelo código da questão e o seu tipo (entre parêntesis rectos, separados por um hífen), assim como a sua descrição;

C. Nesta área surgem os diferentes grupos de questões existentes no boletim. Cada entrada é constituída pelo código do grupo (entre parêntesis rectos) e a sua descrição. Os botões "New", "Edit" e "Delete" respectivos permitem, à semelhança dos da área B, adicionar, editar e remover grupos de questões do boletim;

D. Esta área é constituída por três botões de funcionalidades sobre o boletim em questão. Os mesmos permitem, respectivamente da esquerda para a direita da imagem, fazer uma previsão visual do boletim ("Preview"), guardar o boletim persistentemente num ficheiro XML ("Save") e cancelar a edição do boletim ("Cancel").

Ambas as áreas de gestão de grupos e questões possuem dois pequenos botões "UP" e "DOWN". Estes permitem alterar a posição de uma questão ou de um grupo, caso se encontrem respectivamente em B ou C, na lista correspondente.

4.2.1. Edição dos elementos do boletim

Na criação de um grupo novo, ou edição de um existente, surge a janela apresentada na Figura 4. Esta permite alterar as informações relativas ao grupo seleccionado, nomeadamente o seu código e a sua descrição.

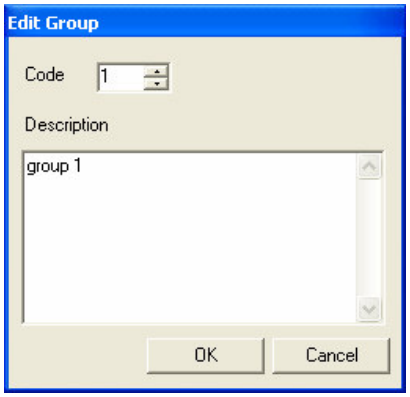

Figura 4 – Ecrã de edição de um grupo

A Figura 5 representa a janela de criação/edição de uma questão. Através da mesma é possível alterar o código, o tipo e a descrição da questão. O tipo da questão pode ser um de quatro valores possíveis: "Single", para questões de resposta exclusiva; "Multiple" para questões de resposta múltipla; e "Open Single" e "Open Multiple" para questões do mesmo tipo que as anteriores com a possibilidade de resposta aberta.

É, também, fornecida uma interface para gerir as respostas da questão, semelhante às anteriores de gestão de grupos e questões. Como tal, são disponibilizados os botões "New", "Edit" e "Delete", para respectivamente criar, editar ou apagar uma resposta, assim como os botões "UP" e "DOWN" para alterar a ordem das respostas na lista.

Note-se que uma questão deverá conter no mínimo duas respostas. Um erro será apresentado no caso desta restrição não ser respeitada.

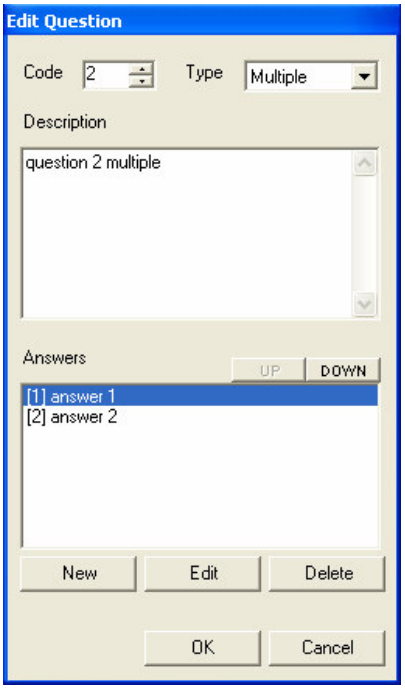

Figura 5 – Ecrã de edição de uma questão

Finalmente, na edição de uma resposta, ou criação de uma nova, é apresentada uma pequena janela onde se pode editar o seu código e descrição (ver Figura 6).

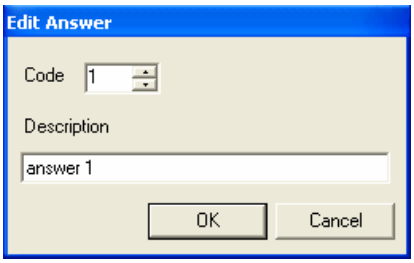

Figura 6 – Ecrã de edição de uma resposta

4.2.2. Pré-visualização do boletim

Através do botão "Preview", referido em 4.2., é possível fazer uma pré-visualização do boletim de voto (ver Figura 7). Esta permite evitar eventuais erros na sua elaboração do boletim, erros que podem não ser detectados na janela de edição de boletins (ver Figura 3).

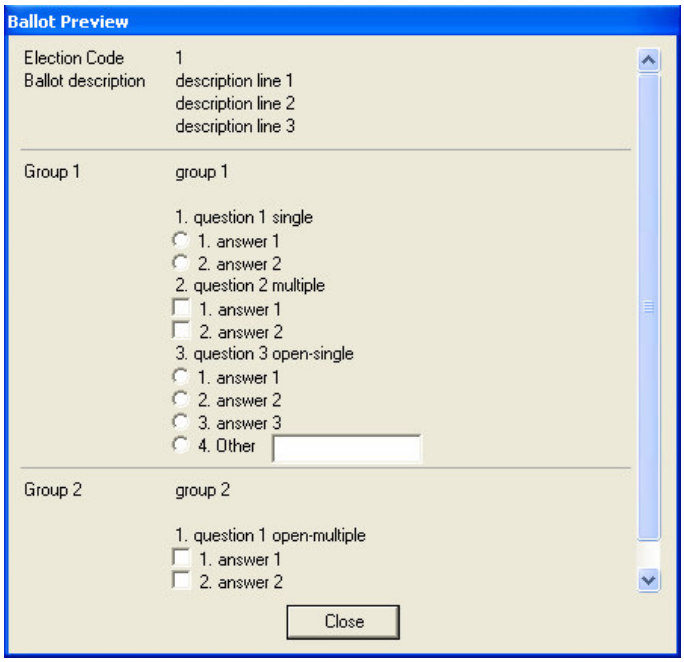

Figura 7 – Ecrã de previsão do boletim

Na janela de pré-visualização são apresentadas as informações gerais do boletim seguidas dos grupos existentes e respectivas questões. Os grupos de questões são separados por uma linha horizontal entre si e surgem barras de deslocamento horizontais e verticais sempre que o aspecto visual do boletim não caiba na janela de previsão, de tamanho fixo. O botão "Close" permite fechar a janela, abandonando a previsão do boletim.

Finalmente, é importante referir que embora os elementos de pré-visualização das questões (botões rádio, caixas de selecção e campos de texto) sejam editáveis, servem apenas para apresentar uma aproximação do aspecto final do boletim, pelo que o seu estado será ignorado nesta janela.

4.3. Preferências

As preferências da aplicação, i.e. as opções de configuração da sua utilização, podem ser acedidas através do botão "Preferences" ou da entrada com o mesmo nome no separador "File" do menu contextual (ver Secção 4.1.). O ecrã da Figura 8 revela as opções que podem ser configuradas para a utilização da aplicação.

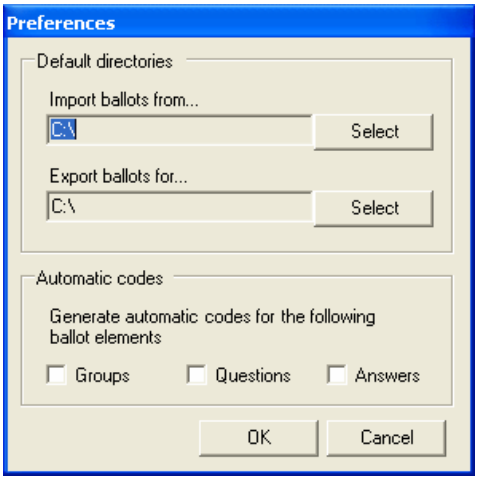

Figura 8 – Ecrã de opções de configuração

O primeiro grupo de opções permite definir as directorias por omissão para a leitura e salvaguarda dos boletins em ficheiros XML. O segundo grupo permite definir códigos automáticos para os grupos, questões e/ou respostas na edição de um boletim. Neste caso, os códigos dos elementos seleccionados são definidos automaticamente pela aplicação sem ser necessária intervenção humana.