UNIVERSIDADE FEDERAL DA BAHIA – UFBA
ESCOLA POLITÉCNICA / INSTITUTO DE MATEMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM MECATRÔNICA

CARLOS CASTELO BRANCO CALDAS NETO

DESENVOLVIMENTO DE UM LEITOR DE CARTÕES INTELIGENTES PARA DISPOSITIVOS MÓVEIS COM COMUNICAÇÃO BLUETOOTH

DISSERTAÇÃO DE MESTRADO
SISTEMAS MECATRÔNICOS

SALVADOR
2011

CARLOS CASTELO BRANCO CALDAS NETO

DESENVOLVIMENTO DE UM LEITOR DE CARTÕES INTELIGENTES PARA DISPOSITIVOS MÓVEIS COM COMUNICAÇÃO BLUETOOTH

Dissertação apresentada ao Programa de Pós-Graduação em Mecatrônica da Escola Politécnica e do Instituto de Matemática, Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre.

Orientadora: Fabíola Gonçalves Pereira Greve

SALVADOR

2011

AGRADECIMENTOS

Agradeço a Deus por sempre ter me dado condições de chegar onde desejei estar. Agradeço, em especial, à minha orientadora, Fabíola Gonçalves Pereira Greve, pela excelente orientação e dedicação empreendida a mim e a esse trabalho que propiciou a sua realização. Agradeço também aos demais professores do PPGM, pelo conhecimento e experiência compartilhados nas aulas, seminários ou mesmo nos contatos a distância.

Gostaria de agradecer aos meus amigos e colegas de curso que de forma direta ou direta contribuíram para a realização deste mestrado. Em especial aos colegas e amigos José Emílio, Ivanoé Rodowanski e Michel Lima que compartilharam muitas madrugadas de estudo e aprendizado.

Agradeço a toda minha família pelo carinho e pelo apoio prestado para que eu pudesse atingir mais este objetivo na vida. Agradeço especialmente meu irmão Marcos Caldas por sempre ter sido prestativo e solícito em me ajudar.

A todos vocês, que de algum modo contribuíram para a realização deste sonho, o meu sincero muito obrigado!

“Se pude ver mais longe foi porque estava sobre os ombros de gigantes”

(Isaac Newton)

RESUMO

Atualmente, diversas transações que envolvem utilização de serviços governamentais, bancários ou comerciais são realizados eletronicamente e assinados de forma digital. No Brasil, desde 2001, documentos assinados com certificados digitais ICP-Brasil têm validade legal e, recentemente, têm se difundido muito devido a ações de fomento realizadas pelo governo que são visíveis em leis, normas e na utilização de documentos como *e-CPF*, *e-CNPJ* e doravante o Registro Único de Identidade Civil (RIC). Os documentos supracitados são cartões inteligentes (ou *smart cards*) que armazenam certificados digitais e, consequentemente, podem ser usados para realizar assinatura digital. Embora leitores de *smart cards* (também conhecidos como CAD – *Card Acceptance Device*) estejam populares e venham sendo utilizados, como é possível constatar, por exemplo, na Receita Federal e na Justiça Superior, a grande maioria os leitores de cartões inteligentes atualmente disponíveis funcionam apenas ligados a computadores pessoais. A grande popularização de *tablets* e telefones celulares inteligentes (ou *smart phones*) tem gerado uma forte tendência de utilização destes dispositivos móveis para a realização de transações eletrônicas como compras, transferências bancárias, envio de *e-mails* e outros. Dessa forma, é desejável assinar documentos digitalmente, através de um certificado digital armazenado em um cartão inteligente, a partir de um dispositivo móvel, como por exemplo, um celular.

O presente trabalho propõe o desenvolvimento do dispositivo denominado SCREAD MOD (*Smart Card Reader for Mobile Devices*), um leitor e escritor portátil de cartões inteligentes (*smart cards*) que funciona sem fios, desenvolvido para ser acoplado e/ou utilizado de forma integrada com objetos inteligentes (celulares, computadores pessoais, *tablets PC's*, etc.), que possam se comunicar utilizando a tecnologia de comunicação sem fios *Bluetooth*. Para demonstrar a viabilidade técnica do protótipo, um exemplo de aplicativo que executa sobre o sistema operacional móvel Android foi desenvolvido.

Palavras-chave: Assinatura Digital; *Smart Phone*; *Smart Card*; Bluetooth; Android; ICP-Brasil.

SUMÁRIO

LISTA DE SIGLAS

AC ..... Autoridade certificadora

ACPR ..... Autoridade Certificadora da Presidência da República

ACT ..... Autoridades de Carimbo do Tempo

AES ..... *Advanced Encryption Standard*

AID ..... *Application Identifier*

AOD ..... *Authentication Objects*

APDU ..... *Application Data Unit*

API ..... *Application Programming Interface*

AR ..... Autoridade de Registro

B2B ..... *Business-to-Business*

B2C ..... *Business-to-Consumer*

C2C ..... *Consumer-to-Consumer*

CAD ..... *Card Acceptance Device*

CAdES ..... CMS *Advanced Electronic Signature*

CRL ..... *Certificate Revocation Lists*

DCE ..... *Data Communication equipment*

DES ..... *Data Encryption Standard*

DH ..... Diffie and Helmman

DTE ..... *Data Terminal Equipment*

EAD ..... Educação a distância

G2C ..... *Government-to-citizen*

ICC ..... *Integrated Circuit Card*

ICP ..... Infraestrutura de Chaves Públicas

IEC ..... *International Electrotechnical Commission*

IEEE ..... *Institute of Electrical and Electronics Engineers*

IETF ..... *Internet Engineering Task Force*

IKE ..... *Internet Key Exchange*

ISO ..... *International Organization for Standardization*

ITU ..... *International Telecommunication Union*

JCRE ..... *Java Card Runtime Environment*

JME ..... *Java Micro Edition*

LCR ..... Lista de Certificados Revogados

NFC ..... *Near Field Communication*

OASIS ..... *Advancing Open Standard for the information society*

OCSP ..... *Online Certificate Status Protocol*

ODF ..... *Object Directory File*

OTP ..... *One Time Pad*

PCI ..... *Peripheral Component Interconnect*

PIN ..... *Personal Identification Number*

PIX ..... *Proprietary Application Identifier Extension*

PKI ..... *Public Key Infrastructure*

PKIX ..... *Public-Key Infrastructure* (X.509)

PrKDFs ..... *Private Key Directory Files*

PuKDFs ..... *Public Key Directory Files*

RGB ..... *Red, Green, Blue*

RIC ..... Registro Único de Identidade Civil

RSA ..... Rivest, Shamir and Adleman

SHA..... *Secure Hash Algorithm*

SKDFs ..... *Secret Key Directory Files*

SO ..... Sistema Operacional

SSL ..... *Secure Socket Layer*

TLS ..... *Transport Layer Security*

TTL..... *Transistor-Transistor Logic*

UART ..... *Universal Asynchronous Receiver Transmitter*

USB OTG..... USB *On-The-Go*

USB..... *Universal Serial Bus*

W3C ..... *World Wide Web Consortium*

WWW ..... *World Wide Web*

XAdES..... XML-DSig *Advanced Electronic Signature*

LISTA DE FIGURAS

LISTA DE TABELAS

CAPÍTULO 1
- INTRODUÇÃO

A evolução dos meios de comunicação eletrônicos, mais especificamente da Internet, gerou uma grande evolução nos serviços de negócio eletrônico. O negócio eletrônico (*e-business*) contempla governo eletrônico (*e-goverment*), comércio eletrônico (*e-commerce*), banco eletrônico (*e-banking*), educação a distância (EAD ou *e-learning*) e outros. O comércio eletrônico pode envolver relações vendedor-consumidor (B2C – *business-to-consumer*), consumidor-consumidor (C2C – *consumer-to-consumer*) ou mesmo relações entre fornecedores (B2B – *business-to-business*) (DEITEL, DEITEL e STEINBUHLER, 2004). Atualmente, os seguimentos citados são muito utilizados e, isso é visível a partir da popularidade de sítios como Submarino, Mercado Livre, Receita Federal, Siciliano e dos sistemas bancários nacionais que são frequentemente mencionados na mídia como os mais avançados do mundo.

Muitos serviços de negócio eletrônico envolvem utilização de dados sensíveis que não podem ser públicos, como por exemplo, números de cartões de crédito, senhas bancárias, dados de rendimentos financeiros e outros. A utilização destes dados sensíveis fez com que o crescimento vertiginoso das transações eletrônicas trouxesse consigo um aumento no número de fraudes eletrônicas. Assim, são necessários mecanismos para se evitar as fraudes, como por exemplo, deve-se evitar que informações sensíveis sejam interceptadas por pessoas não autorizadas, assim como não deve ser possível que uma transação realizada seja negada por uma das partes que se arrependeu etc.

A realização confiável de transações eletrônicas envolve atender a diferentes critérios de segurança que, a depender da necessidade se relacionam com diversos conceitos tecnológicos, como criptografia, distribuição de chaves, assinaturas digitais e outros que devem ser utilizados de forma a atender requisitos legais. Há uma grande discussão a respeito da validade jurídica ou não das transações eletrônicas que envolvem diferentes tipos de tecnologias que não são previstas em lei (GARCIA, 2004). Contudo, através da Medida Provisória 2200-2 (24/08/01), o Governo Federal regulamentou a Certificação Digital e

conferiu a mesma validade jurídica de um documento físico aos documentos eletrônicos assinados com certificados digitais emitidos por Autoridades Certificadoras credenciadas pela ICP-Brasil (CERTISIGN, 2007).

Dentre as diversas formas de realização de assinaturas digitais, a Infraestrutura de Chaves Públicas do Brasil (ICP-Brasil) classifica como mais seguras as que envolvem a utilização de um hardware criptográfico (BRASIL, 2010). Um hardware criptográfico muito difundido, prático e que possibilita o armazenamento de certificados digitais reconhecidos pelo ICP-Brasil é o cartão inteligente (ou *smart card)*. Há diversos tipos de cartões inteligentes e os cartões inteligentes de contato que são compatíveis com os padrões especificados pelo ICP-Brasil, atualmente, devem atender à norma internacional ISO/IEC 7816.

Figura 1 - e-CPF / e-CNPJ

Fonte: http://www.receita.fazenda.gov.br/atendvirtual/LeiautesDefault.htm em 08/03/2011

Há diversos exemplos de cartões inteligentes que armazenam certificados digitais pertencentes à cadeia ICP-Brasil utilizados. Em 2004 a Receita Federal do Brasil começou a emitir os documentos e-CPF's e e-CNPJ's para cidadãos e empresas (SERPRO, 2004). A Figura 1 contém modelos de certificados digitais emitidos pela Receita Federal do Brasil, onde a parte (a) da figura contém um modelo de certificado digital e-CPF, a parte (b) contém um e-CNPJ e a parte (c) contém o fundo comum aos dois certificados anteriores. Diversas instituições bancárias, como por exemplo, Itaú, Caixa Econômica Federal, Bradesco, Bank Boston Brasil e Banco do Brasil possibilitam autenticação através de cartões e-CPF/e-CNPJ.

Outro exemplo de cartão inteligente é o Registro de Identidade Civil (RIC). O RIC tem previsão de começar a ser emitido ainda em 2011. Contém um número único válido em

todo o território nacional (o número do RIC) e reúne dados de outros documentos, como RG (Registro Geral), CPF (Cadastro de Pessoas Físicas), Título de Eleitor, PIS (Programa de Integração Social), Pasep (Programa de Formação do Patrimônio do Servidor Público), Carteira de Trabalho e Carteira Nacional de Habilitação (PORTAL BRASIL, 2010). Em seus campos de inscrição, o cartão do RIC mostra o nome, sexo, nacionalidade, data de nascimento, foto, filiação, naturalidade, assinatura, impressão digital do indicador direito, órgão emissor, local de expedição, data de expedição e de validade do cartão. Existe ainda um campo de observações optativo que pode trazer outras informações, como tipo sanguíneo e se a pessoa é doadora ou não de órgãos (PORTAL BRASIL, 2010).

Diversos outros serviços governamentais atualmente disponíveis utilizam cartões inteligentes e certificados digitais ICP-Brasil, dentre os quais podem-se destacar: consulta e acompanhamento da Situação Fiscal das Pessoas Físicas e Jurídicas com o e-CPF do responsável legal perante a Receita Federal; elaboração de Procurações Eletrônicas; adoção da Nota Fiscal Eletrônica, dos Livros Fiscais Eletrônicos e do Livro Diário Eletrônico; apresentação de assinatura e firma reconhecida em cartório do Documento Básico de Entrada no CNPJ junto a Receita Federal (DBE) àqueles que utilizam a certificação digital; acesso ao Siscomex com certificado digital e-CPF, obrigatório a partir de 2008; envio eletrônico de documentos (e-Doc) referentes a processos que tramitam nas Varas do Trabalho dos 24 TRTs e no TST, através da Internet, sem a necessidade da apresentação posterior dos documentos originais; acesso ao sistema de Nota Fiscal Eletrônica com e-CNPJ para as Pessoas Jurídicas e com e-CPF para os benefícios fiscais das Pessoas Físicas e outros (CERTISIGN, 2007).

É possível perceber que os cartões inteligentes já são muito utilizados e difundidos em todo mundo; para que os cartões possam funcionar em conjunto com um computador necessitam de leitores de cartões inteligentes compatíveis. A grande maioria dos leitores de cartões inteligentes, também chamados de *Card Acceptance Device (*CAD) comercializados são destinados à utilização em computadores pessoais, o que em termos práticos, limita a sua aplicabilidade à presença de um computador pessoal.

Após uma pesquisa de mercado, identificou-se que, no Brasil, os fornecedores de leitores de cartões inteligentes, dentre os quais se incluem Gemalto, CIS Eletrônica, Nonus,

atualmente, apenas comercializam leitores de cartões inteligentes compatíveis com a interface USB que devem trabalhar em conjunto com um PC. A Figura 2 contém leitores de cartões inteligentes dos três fabricantes citados.

Figura 2 - Leitores de cartões inteligentes comumente comercializados no Brasil
Fonte: Nonus Smarthome: www.nonus.com.br (acesso em 04/01/2011).
GEMPC Twin USB: www.gemaltodobrasil.com.br (acesso em 04/01/2011).
Argos Mini II: www.ciseletronica.com.br (acesso em 04/01/2011).

Os cartões inteligentes, quando utilizados por um computador através de um leitor de cartões inteligente conectado, propiciam um meio legal e eficiente para se realizar transações eletrônicas de forma segura, pois são capazes de armazenar certificados digitais ICP-Brasil. Uma restrição presente, no entanto, está no fato de que os leitores de *smart card* comumente comercializados no Brasil apenas são compatíveis com computadores pessoais e não permitem que os *smart cards* sejam utilizados com uma classe de dispositivos que há algum tempo têm ganhando uma aplicabilidade muito grande na realização de transações eletrônicas: os celulares inteligentes (*smart phones)*.

Os *smart phones* são telefones móveis que trazem uma série de recursos que os equiparam em muitos aspectos aos computadores pessoais. Os *smart phones* possuem sistema operacional, memória para armazenamento de dados, possibilitam a instalação de jogos e outros programas de propósito geral. Também são capazes de processar filmes em alta resolução, servir como roteadores *wireless,* acessar a Web e dispõem da tecnologia *Bluetooth* que possibilita a comunicação sem fio com outros celulares e/ou computadores pessoais. Contudo os *smart phones* ainda possuem recursos limitados de processamento, de memória e de alimentação energética (utilizam baterias) quando comparados a um computador pessoal. Dessa forma o desenvolvimento de software para *smart phone* envolve algumas preocupações

relacionadas ao uso de recursos que não estão presentes no desenvolvimento de software para computadores pessoais.

Diversos exemplos confirmam a tendência de expansão dos serviços de comércio eletrônico, anteriormente realizados apenas por computadores, para os smart phones. Por exemplo, o Banco do Brasil e o Itaú desenvolveram aplicações específicas para o iPhone; em janeiro de 2011 o Itunes App Store ultrapassou a marca de 10 bilhões de downloads[1]; em 2010 o Android Market Place ultrapassou a marca de 100.000 aplicativos e mais de 1 bilhão de downloads[2]; no fim de 2010 a Cielo (empresa líder em soluções de pagamento eletrônico na América Latina[3]) firmou uma parceria com a operadora de telefonia Oi e o Banco do Brasil para desenvolver um sistema de pagamento via celular através de mensagens SMS com um potencial de mercado estimado em três milhões e meio de usuários[4].

É possível perceber que os exemplos da aplicabilidade dos celulares inteligentes como mecanismos para processamento de negócio eletrônico são diversos. Dessa forma, entende-se que a construção de um leitor portátil de cartões inteligentes que atua de forma integrada a um celular, compartilhando com este a principal vantagem de um celular em relação a um computador pessoal que é a mobilidade, é algo desejável.

1.1 - OBJETIVOS

O presente trabalho propõe o desenvolvimento do dispositivo denominado SCREAD MOD (***S**mart **C**ard **Re**ader for **Mo**bile **D**evices*), um leitor e escritor de cartões inteligente (*smart cards*) desenvolvido para ser acoplado e utilizado de forma integrada com *smart phones* que utilizem o sistema operacional Android e possam se comunicar utilizando a tecnologia de comunicação sem fios *Bluetooth*. Por possuir comunicação *Bluetooth,* o

[1] http://www.edibleapple.com/itunes-app-store-reaches-10-billion-downloads/
2http://www.mobilecrunch.com/2010/07/15/android-hits-100000-apps-and-a-billion-downloads/
3 http://www.cielo.com.br/portal/cielo/conheca-a-cielo/quem-somos.html
[4] http://www.cielo.com.br/portal/cielo/servicos/banco-do-brasil-e-cielo-anunciam-parceria-com-a-oi-para-operar-o-servico-de-mobile-payment.html

dispositivo SCREAD MOD não terá sua utilização restrita a celulares inteligentes e poderá ser utilizado em conjunto com computadores pessoais ou *Tablets PCs* que sejam compatíveis.

O dispositivo a ser desenvolvido inclui circuitos para leitura/escrita de *smart cards* de contato, circuitos para transmissão/recepção de dados utilizando a tecnologia *Bluetooth* e a programação de um microcontrolador que irá gerencia a integração entre os componentes. Paralelamente será desenvolvido um *software* para um celular que utiliza SO *Android* que deverá ser capaz de estabelecer uma conexão e realizar uma assinatura digital através do envio e recebimento de comandos que o *smart card* é capaz de interpretar, isto é, comandos descritos na norma ISO 7816. Como será utilizado um circuito compatível com a especificação ICP-Brasil o dispositivo também será compatível com certificados digitais ICP-Brasil.

O objetivo geral deste trabalho deverá ser atingido a partir dos seguintes objetivos específicos:

- Elaboração de uma análise que permite comparar as tecnologias de acesso ao meio sem fio WiFi, IrDa, *Bluetooth* e NFC (*Near Field Communication*) para justificar a escolha da tecnologia *Bluetooth*;
- Desenvolvimento de um protótipo funcional do dispositivo proposto;
- Desenvolvimento de um software aplicativo desenvolvido para o sistema operacional *Android* que realiza assinatura digital de documentos utilizando chaves privadas e certificados digitais armazenados em cartões inteligentes para demonstrar o funcionamento do protótipo através da assinatura digital de um documento;

1.2 - TRABALHOS RELACIONADOS

Após uma pesquisa de mercado foram encontrados dois modelos de leitores de *smart card Bluetooth*: um modelo fabricado pela Research In Motion e outro fabricado pela HID Global. Algumas patentes

Na segunda metade de 2007 a loja virtual Amazon começou a comercializar um leitor de *smart cards Bluetooth* denominado Blackberry Smart Card Reader Bluetooth fabricado pela empresa Research In Motion (RIM). O BlackBerry Smart Card Reader é projetado para conectar *smart phones* BlackBerry com suporte a *Bluetooth* e computadores com suporte a *Bluetooth* (RESEARCH IN MOTION, 2007).

O leitor fabricado pela RIM - visto na Figura 3(a) – atualmente é compatível apenas com computadores que executam a plataforma Windows e, tem como característica interessante o fato de que é possível utilizá-lo conectado à porta USB, interface pela qual o dispositivo também é recarregado. Desde o fim de 2009 os leitores são compatíveis com smart cards que atendem à especificação ISO 7816 que são os modelos atualmente utilizados nos leitores vendidos no Brasil; atualmente o dispositivo custa aproximadamente U$ 200,00.

Figura 3 - Leitores smart card Bluetooth

Fonte: (a) Site http://us.blackberry.com/ataglance/security/products/smartcardreader/, acessado em 11/2010.
(b) Site www.hidglobal.com, acessado em 04/2011.

Em 2011 a HID Global, empresa especializada em tecnologias de soluções de identificação e validação que utilizam *smart cards*, anunciou o leitor de *smart cards* OMNIKEY 2061 Bluetooth Reader - Figura 3 (b). O produto ainda não é comercializado, mas segundo especificações do fabricante o item é compatível apenas com computadores que utilizam a plataforma Windows, porém a empresa está trabalhando para tornar o produto compatível com computadores que utilizem as plataformas Linux ou MacOS e também para alguns celulares que utilizem a plataforma Windows Mobile. O dispositivo OMNIKEY 2061 também é recarregável pela porta USB e é compatível com o padrão ISO 7816.

Após a identificação de produtos comerciais, foi realizada uma busca na base de patentes nacionais e não foi encontrado nenhum leitor de cartões inteligentes sem fio. Uma busca na base de patentes internacionais permitiu identificar patentes parecidas, dentre as quais se destacam as patentes: CA2541741A1- *Portable Smart Card Reader Having Secure Wireless Communications Capability*; US2009276626A1-*Portable Smart Card Reader Having Secure Wireless Communications Capability*; CN2624295Y - *A wireless mobile IC card reader*; DE20320080U1 - *Card reader for mobile equipment such as a telephone, uses an infra red or Bluetooth link*. Não por acaso, dentre as citadas, as patentes que mais se aproximam do objetivo deste trabalho são as patentes CA2541741A1 e US2009276626A1 que pertencem à Research In Motion e descrevem dispositivos que se assemelham ao dispositivo Blackberry Smart Card Reader Bluetooth.

1.3 - ESTRUTURA DA DISSERTAÇÃO

A estrutura desta dissertação está organizada conforme segue: o Capítulo 2 contém a fundamentação teórica necessária ao entendimento de como é possível se obter transações seguras recorrendo a conceitos relacionados à segurança da informação, assinatura digital, certificados digitais e outros. O Capítulo 3 descreve normas e padrões existentes que são aplicados no contexto de certificados digitais, cartões inteligentes e avaliação de segurança de *software* e *hardware*. O Capítulo 4 descreve tecnologias de comunicação sem fio que permitem comunicação entre dois dispositivos e as compara. O Capítulo 5 detalha as tecnologias de comunicação que incluem UART e USB. O capítulo 6 descreve o funcionamento de cartões inteligentes compatíveis com a norma ISO/IEC 7816. No Capítulo 7 o projeto do um leitor de cartões inteligentes *Bluetooth* é desenvolvido em conjunto com sistemas de *software* desenvolvidos para executar em um celular inteligente que utiliza o sistema operacional Android permitem validar o protótipo desenvolvido. Finalmente, no Capítulo 9 a conclusão é detalhada bem como as sugestões para trabalhos futuros.

CAPÍTULO 2
- COMUNICAÇÕES SEGURAS

Há diversos cenários em que a comunicação realizada entre duas entidades necessita de mecanismos de segurança. Por exemplo, em transações eletrônicas de compras dados sensíveis que incluem números de cartões de crédito, dados bancários e outros tipos de dados pessoais devem ser sigilosos e visíveis apenas às partes envolvidas na transação.

Em outros cenários pode haver a necessidade de que os dados envolvidos na comunicação não sejam indevidamente alterados, pois uma alteração indevida nos dados comunicados pode prejudicar pelo menos uma das partes envolvidas. Por exemplo, uma empresa A pode interceptar um orçamento enviado por uma empresa concorrente B (que disponibiliza seus preços publicamente), aumentar os valores descritos em tal orçamento e fazer com que um possível cliente desista de comercializar com a empresa B.

É possível perceber que nos exemplos supracitados existem duas necessidades diferentes que objetivam o mesmo fim: manter a comunicação segura. As necessidades que tornam a comunicação segura variam de acordo com o contexto, pois em um cenário o sigilo é algo essencial enquanto que em outro cenário é essencial que a informação não seja indevidamente alterada ou que seja possível identificar com certeza quem é o emissor de uma determinada informação.

As diversas necessidades que tornam a comunicação segura são comumente referenciadas na literatura como serviços. Nesta dissertação, a expressão “serviços” será também referenciada pela expressão “propriedade”. Dessa forma, o serviço que provê sigilo a uma comunicação, por exemplo, poderá ser referenciado como uma propriedade da comunicação na qual ela ocorre de forma sigilosa.

Neste capítulo serão apresentadas as propriedades que a literatura relaciona à segurança das comunicações e, posteriormente, as técnicas que objetivam alcançar as diversas propriedades de segurança.

2.1 – PROPRIEDADES PARA UMA COMUNICAÇÃO SEGURA

Uma comunicação segura depende da capacidade de se manter segura a informação envolvida no processo de comunicação. As propriedades (também conhecidas como serviços) de segurança que devem ser garantidas para a informação variam com o contexto e as necessidades do negócio. As propriedades ou critérios mais comumente citados pela literatura são: confidencialidade, integridade e autenticidade. Essas propriedades se relacionam com outras como disponibilidade, não repúdio e autorização (ou controle de acesso).

Disponibilidade e controle de acesso são extensões mais recentes da noção de comunicações seguras (KUROSE e ROSS, 2006). Confidencialidade, autenticação, não repúdio de mensagens e integridade são considerados componentes fundamentais da comunicação segura há bastante tempo. São definições dos conceitos supracitados:

- **Autorização (ou controle de acesso):** políticas de níveis de acesso das pessoas à informação e é definida de acordo com os critérios específicos de cada Organização (ISO/IEC, 2005).

- **Autenticidade (ou autenticação):** é o processo pelo qual se pode provar a identidade de uma pessoa (KUROSE e ROSS, 2006). No mundo físico isso é realizado através de uma assinatura, uma impressão digital, uma conferência de um documento ou uma combinação de métodos.

- **Confidencialidade (ou sigilo ou privacidade):** apenas usuários autorizados podem ter acesso à informação; numa comunicação seriam apenas o remetente e o destinatário (KUROSE e ROSS, 2006). Cofres, alarmes, seguranças e outros são mecanismos utilizados para garantir a confidencialidade no mundo físico.

- **Disponibilidade:** a informação deve estar disponível quando necessária (ISO/IEC, 2005). No mundo físico, várias cópias de um documento podem ser dispostas em locais diferentes para que se possa garantir disponibilidade do mesmo.

- **Integridade:** a informação não pode ser alterada (ou danificada) por alguém ou algo que não disponha de autorização. Em uma comunicação a informação deve chegar ao destinatário do mesmo modo que foi enviada pelo remetente (KUROSE e ROSS, 2006). No mundo físico alguns mecanismos como papel especial, rubrica em todas as mudanças, autenticação de cópias são utilizados para garantir a integridade da informação.

- **Não repúdio (ou irretratabilidade):** prova o acontecimento de um fato ou ação. Um receptor pode provar que um remetente designado enviou de uma mensagem (KUROSE e ROSS, 2006) ou assinou um documento. Não é possível garantir a irretratabilidade sem garantir a autenticidade. Diversos mecanismos não digitais são utilizados atualmente, como assinaturas autenticadas em cartório, testemunhas, gravações e outros.

Conforme citado anteriormente, é possível haver segurança sem a necessidade de se atender a todas as propriedades. Os critérios utilizados para escolher uma propriedade ou outra podem ser operacionais, políticos ou legais. Todas as propriedades supracitadas são interligadas, mas podem afetar uma organização de forma diferenciada. Por exemplo, pode não ser essencial que sistemas de banco eletrônico disponham de disponibilidade 24 horas para alguns tipos de serviços, mas é essencial que quando disponível o sistema tenha algum mecanismo que garanta a autenticidade do utilizador do serviço. Por isso é essencial entender exatamente qual propriedade é desejada em cada contexto. Diversas técnicas podem ser empregadas para se garantir as propriedades de autenticação, sigilo, integridade, não repúdio e disponibilidade e controle de acesso. A seguir serão vistas técnicas que possibilitam alcançar as propriedades de sigilo, integridade, autenticação e não repúdio. Autorização e disponibilidade são temas que não serão tratados nesse texto.

2.2 – CRIPTOGRAFIA

A palavra criptografia (*kriptós* = escondido, oculto; *gráphein* = escrita) deriva do grego e significa “escrita secreta”. A criptografia foi muito utilizada na segunda guerra mundial pelas forças armadas dos EUA permitindo o entendimento da mensagem apenas ao

seu destinatário. Contudo, sua origem é muito mais antiga, já sendo utilizado pelos egípcios quase 2000 anos antes de cristo (MORKEL e ELOFF, 2004).

A criptografia é o principal mecanismo utilizado para se garantir o sigilo em uma comunicação (KUROSE e ROSS, 2006). Diversos cenários de comunicação que requerem sigilo podem ser reduzidos à situação em que há um Emissor querendo enviar uma mensagem sigilosa a um Receptor e, há possivelmente um Intruso querendo interceptar a informação que não lhe diz respeito. Como existem diversos tipos de canais de comunicação e não se podem fazer suposições sobre a inviolabilidade ou não do canal, faz-se necessário utilizar técnicas de segurança que permitam uma comunicação sigilosa entre o Emissor e o Receptor.

A Figura 4 contém um modelo de comunicação que descreve um cenário em que há uma comunicação sigilosa. Há um Emissor (*Sender*) que deseja enviar uma mensagem (*Message*) para um Destinatário (*Recipient*). O Emissor utiliza algum tipo de técnica secreta (*Secret Information*) para aplicar uma transformação na mensagem original gerando uma mensagem segura (*Secure Message*). A mensagem segura trafega por um canal não confiável, onde há um Intruso (*Opponent*) que possivelmente é capaz de ler a mensagem. Após a mensagem chegar ao destinatário, este deve ser capaz de fazer a transformação inversa e obter a mensagem original. A técnica secreta foi entregue ao destinatário por uma terceira pessoa confiável (*Trusted third party*).

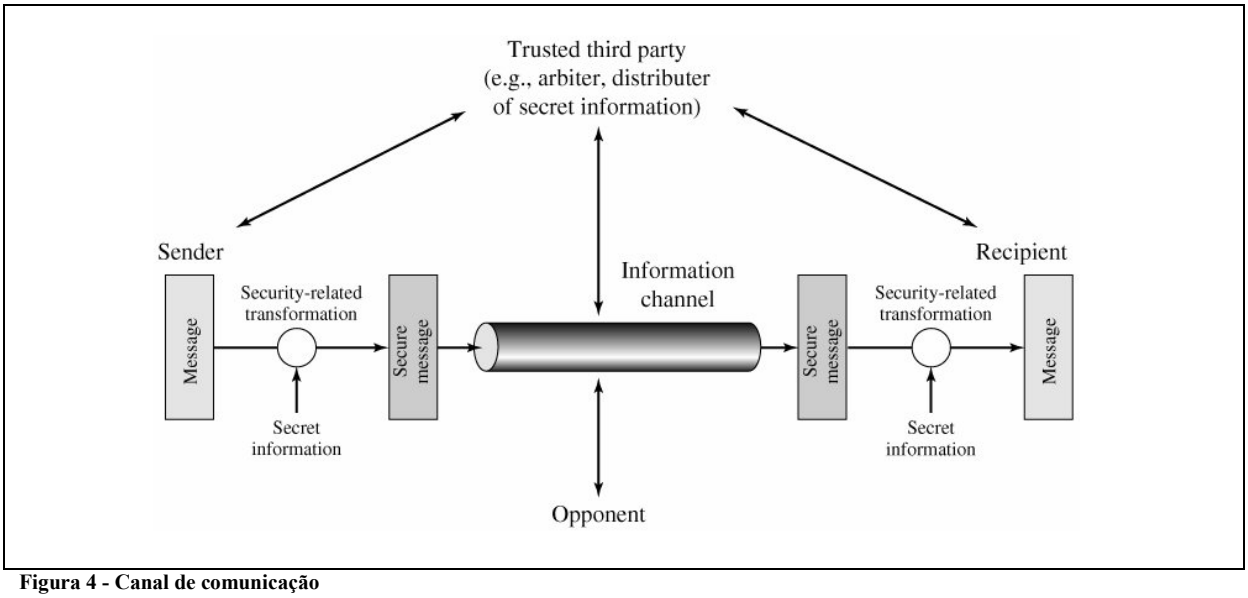

Figura 4 - Canal de comunicação

Fonte: (STALLINGS, 2005)

Uma mensagem não criptografa é conhecida como texto aberto, texto claro ou texto simples (*plain text*) enquanto que a mensagem criptografada é conhecida como texto cifrado (*cipher text*) (KUROSE e ROSS, 2006). A técnica ou algoritmo utilizado para criptografar uma mensagem é também conhecido como criptograma (DEITEL, DEITEL e STEINBUHLER, 2004).

A criptografia pode ser realizada de dois modos: *criptografia de códigos* e *criptografia de cifras*. Na criptografia de cifra há uma transformação caractere por caractere ou de bit por bit, sem levar em conta a estrutura linguística da mensagem. Em contraste, um código substitui uma palavra por outra palavra ou símbolo. Os códigos não são mais utilizados atualmente (TANENBAUM, 2003).

Os antigos criptogramas que trabalhavam com cifras eram classificados em criptogramas de substituição e criptogramas de transposição. Em um criptograma de substituição, toda a ocorrência de uma determinada letra é substituída por outra diferente; por exemplo, se todo "a" for substituído por um "b", todo "b" por um "c" etc., a palavra "segurança" seria criptografada como "tfhvsbodb". Em um criptograma de transposição a ordem das letras é modificada; por exemplo, alternando-se as letras da palavra "segurança", onde as letras de posição ímpar criam a primeira palavra no texto cifrado e as restantes criam a segunda; assim "segurança" seria criptografada como "sgrnç euaa". (DEITEL, DEITEL e STEINBUHLER, 2004)

Entre as cifras de substituição mais antigas se destacam: o *Atbash*, utilizada pelos hebreus para criptografar o livro bíblico de Jeremias substituindo a primeira letra pela última, a segunda letra pela penúltima e assim sucessivamente (MORKEL e ELOFF, 2004); e a cifra de César (*Caese cipher*) atribuída a Júlio César em que o a se torna D, b se torna E, c se torna F, ... e z se torna C (KUROSE e ROSS, 2006).

É possível perceber que a segurança utilizada na Cifra de César, no *Atbash* ou em outras técnicas antigas se baseia no fato de que uma pessoa não autorizada não conhece ou não é capaz de identificar o algoritmo utilizado, por isso o algoritmo deveria ser secreto e conhecido apenas por pessoas autorizadas. Manter o algoritmo de criptografia secreto e disponível apenas para os envolvidos é conhecido como *segurança por obscuridade*.

Desenvolver um algoritmo que simplesmente “embaralha” um dado pode gerar alguns problemas. Por exemplo, se o algoritmo for descoberto, o sigilo de todas as mensagens cifradas com aquele algoritmo está comprometido; paralelamente deverá haver um grande esforço em atualizar equipamentos, além de desenvolver, difundir e treinar todos os envolvidos no novo algoritmo. A abordagem conhecida como *criptografia com uso de chaves* possibilita a alteração instantânea de futuros criptogramas sem que se torne necessário atualizar equipamentos, desenvolva-se um novo algoritmo ou gaste-se tempo com treinamentos.

Na criptografia com uso de chaves há uma função (ou algoritmo) que recebe e criptografa um texto simples a partir de um parâmetro chamado de chave. Será gerado um criptograma diferenciado para cada chave passada como parâmetro; o método que decriptografa o texto cifrado também precisará da chave para obter novamente o texto simples original.

É um consenso atual entre os especialistas que segurança por obscuridade é uma opção ruim, pois como citado anteriormente, caso o algoritmo seja descoberto o impacto é muito grande. Esse consenso hoje é conhecido como *Princípio de Kerckhoff*, princípio em que “todos os algoritmos devem ser públicos; apenas as chaves são secretas” (TANENBAUM, 2003).

Existem dois tipos de algoritmos de criptografia que utilizam chaves: *algoritmos de chave simétrica* e *algoritmos de chave assimétrica (ou chave pública)*. No primeiro caso a mesma chave é utilizada para criptografar e para decriptografar o texto simples. No segundo caso existem duas chaves; uma é utilizada para criptografar e a outra é utilizada para decriptografar. Ambos os modelos serão detalhados a seguir.

2.2.1 - CRIPTOGRAFIA SIMÉTRICA

Os algoritmos de criptografia simétrica são também chamados de algoritmos de chave secreta. Os algoritmos dessa classe utilizam a mesma chave para codificação e decodificação, conforme pode ser percebido pela Figura 5.

A criptografia simétrica pode ser realizada, basicamente, por algoritmos de fluxo - trabalham com o texto um byte por vez – e algoritmos de bloco que trabalham sobre blocos de tamanho fixo (ex.: 128 bits). Como exemplos de algoritmos muito difundidos destacam-se os algoritmos RC4 e RC5 que trabalham com cifras de fluxo e os algoritmos que trabalham com cifras de bloco *Data Encryption Standard* (DES), *Triple Data Encryption Algorithm* (TDEA ou 3DES) e *Advanced Encryption Standard* (AES).

Cifras de fluxo são na maioria das vezes mais rápidas e geralmente usam muito menos código que as cifras de bloco. A cifra de fluxo RC4 é pelo menos duas vezes mais rápida que a mais rápida cifra de bloco. RC4 pode ser escrita com talvez 30 linhas de código. A maioria das cifras de bloco requerem centenas de linhas de código (BURNNET e PAINE, 2001).

Um problema associado às cifras de fluxo é o fato de que estas se baseiam na técnica OTD (*one-time pad*) e, para que a técnica OTP tenha eficácia é essencial que as chaves utilizadas no processo criptográfico não possam ser utilizadas novamente (BURNNET e PAINE, 2001). Por exemplo, caso uma empresa armazene dados sigilosos de seus clientes no banco de dados, o dado de cada cliente deveria ser criptografado com uma chave diferente, tornando a gerência de chaves muito mais complexa do que se fosse utilizado uma única chave.

Dessa forma, cada tipo de criptografia simétrica (fluxo ou bloco) tem uma indicação que depende da necessidade. Se a chave necessita ser reutilizada é indicado a criptografia com cifras de bloco; se a velocidade é fundamental e as chaves não forem reutilizadas é indicado a utilização de cifras de fluxo. No primeiro caso o AES é recomendado por questões de padronização de mercado (BURNNET e PAINE, 2001).

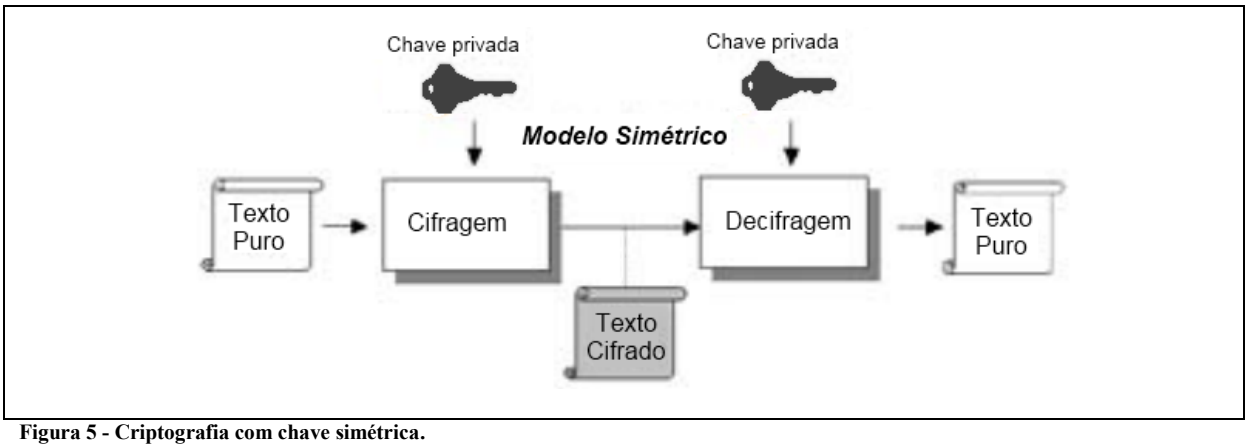

Figura 5 - Criptografia com chave simétrica.
Fonte: (MORENO, PEREIRA e CHIARAMONTE, 2005).
Nota: Adaptador pelo autor.

Para criptografia simétrica funcionar as duas partes envolvidas na comunicação (emissor e receptor) necessitam compartilhar a mesma chave e a chave deve ser protegida de um possível intruso que possa interceptar a troca de chaves (vide Figura 4); dessa forma, o problema da troca de informações confidenciais foi reduzido à troca de chaves confidenciais. Consequentemente, a troca de chaves frequente é desejável para limitar a quantidade de dados comprometidos caso um intruso seja capaz de identificar a chave (STALLINGS, 2005); adicionalmente, deve-se desenvolver uma política eficiente de distribuição de chaves.

DISTRIBUIÇÃO DE CHAVES

A distribuição de chaves é o fator determinante na eficácia da criptografia simétrica. A **Figura 4** contém um modelo de comunicação que envolve quatro atores: Emissor, Receptor, Intruso e uma terceira pessoa confiável. Este modelo será utilizado para derivar as quatro possibilidades de distribuição de chaves que serão descritas em sequência. Ao invés dos nomes Emissor e Receptor, estes serão referenciados como A e B, uma vez que ambos podem assumir os papéis de remetente e destinatário; por questão de simplificação a terceira pessoa confiável será referenciada como C.

A primeira possibilidade é a pessoa A entregar a chave fisicamente à pessoa B. A e B podem se comunicar sempre utilizando a mesma chave e, frequentemente trocarem as chaves fisicamente para aumentar o grau de segurança. No cenário atual em que velocidade é um fator essencial e as limitações geográficas não existem no mundo virtual esta técnica pode ser

muito dispendiosa, pois provavelmente não é interessante uma pessoa ter que viajar para outra localidade (possivelmente um continente) simplesmente para ter que trocar a chave e manter a comunicação segura com outra.

A segunda possibilidade é o Emissor e Receptor trocarem a chave fisicamente uma única vez e, posteriormente trocarem a chave através de mensagens criptografadas com a chave anteriormente utilizada. Esta possibilidade traz um ganho considerável em relação à primeira possibilidade, mas ainda assim mantém o inconveniente fato de ter que haver pelo menos um contato físico entre partes que possivelmente estarão muito distantes. Caso uma chave seja descoberta será necessário um novo contato físico.

A terceira possibilidade seria uma terceira pessoa confiável (por exemplo, uma entidade governamental) selecionar uma chave e entregar aos envolvidos no processo, ou um dos envolvidos escolher uma chave e solicitar que a terceira pessoa a entregue. Esta hipótese é muito utilizada por instituições bancárias que enviam a senha de seus clientes pelos Correios. Esta solução é financeiramente melhor que a segunda possibilidade, mas ainda assim é demasiadamente lenta e, ainda que uma entidade governamental de um país seja cem por cento confiável a de outro país pode não ser e isso inviabilizaria a confidencialidade.

Uma quarta possibilidade é haver uma conexão criptografada de A para C e outra conexão criptografada entre B e C. Dessa forma seria possível solicitar eletronicamente uma chave e obter uma resposta em tempo real. Essa solução resolveria o problema da latência entre a troca de chaves, mas ainda assim sua segurança utiliza como pressuposto o fato de C ser uma instituição confiável e isso é um problema grande nas comunicações que envolvem diferentes países ou países que mantenham algum tipo de censura.

Em 1976, dois pesquisadores, chamados Diffie e Hellman, publicaram pela primeira vez o conceito de Criptografia com Chaves Públicas (STALLINGS, 2005). É um conceito que resolve os problemas das trocas de chaves presentes na criptografia simétrica.

2.2.2 - TROCA DE CHAVES EM UM MEIO INSEGURO

Em 1976 dois pesquisadores chamados Diffie e Helmman provaram que é possível realizar a troca de chaves secretas utilizando um meio inseguro. Para isso utilizaram propriedades matemáticas que se baseiam na ideia de que é computacionalmente possível calcular determinadas funções, mas é computacionalmente inviável calcular a sua inversa (STALLINGS, 2005). Em (COUTINHO, 2007), mostra-se que, mesmo utilizando os computadores mais poderosos da atualidade, seriam necessários zilhões de anos para fatorar um número primo de 100 algarismos que é um valor comumente utilizado em uma das técnicas que permite a troca de chaves em um meio inseguro.

O algoritmo Diffie-Hellman, por exemplo, resolve o problema da troca de chaves em um canal inseguro baseado no problema do cálculo de logaritmos discretos. Os algoritmos criptográficos de chave pública, conhecidos como *Elliptic Curve Cryptography* (ECC) e RSA, também são muito utilizados com a finalidade de se realizar troca de chaves sobre um canal inseguro. Enquanto este utiliza conceitos associados aos problemas da fatoração de números primos, o outro se baseia no problema do cálculo de logaritmos discretos de curvas elípticas. Nesta seção será apresentado o algoritmo Diffie-Hellman.

ALGORITMO DIFFIE-HELMAN (DH)

O algoritmo Diffie-Hellman é também chamado de Método do Acordo de Diffie-Hellman, Acordo de Chave Exponencial ou simplesmente DH. Diffie-Hellman é descrito na RFC 2631[5] da *Internet Engineering Task Force(IETF*) e na norma PKCS#3[6].

Diffie-Hellman resolve o problema do compartilhamento de chaves, uma vez que possibilita a dois usuários trocarem uma chave secreta ao longo de um meio inseguro, sem segredos prévios (RSA LABORATORIES).

Para se entender o funcionamento do algoritmo DH, primeiro deve-se entender o conceito de raiz primitiva. Uma raiz primitiva *a* de um número primo *p* é definida como uma de suas potências $a^i \bmod p$ ($1<=i<=p-1$) que gera todos os inteiros que variam de *1* a *p-1*. Dessa forma $a \bmod p$, $a^2 \bmod p$, ..., $a^{p-1} \bmod p$ são números inteiros distintos e consistem em

[5] http://www.ietf.org/rfc/rfc2631.txt
[6] Descrito na seção Normas e Padrões

inteiros que variam de *1* a *p-1* permutados (STALLINGS, 2005). Por exemplo, sendo a= 3 e p=7 dizemos que *a* é uma raiz primitiva de p, pois k variando de 1 a 6 (*p*-1) temos os números 1,2,3,4,5,6 conforme pode ser visto na Tabela 1:

Tabela 1 - Raiz primitiva do número 7

I	3^i	3^i mod 7
1	3	3
2	9	2
3	27	6
4	81	4
5	243	5
6	729	1

Para qualquer número inteiro b e uma raiz primitiva de um número primo p, é possível achar um único expoente *i* tal que $b \equiv a^i \ (mod\ p)$ onde 0<=*i*<=*p*-1. O expoente *i* é referenciado como o logaritmo discreto de b para a base a mod pe é expresso como $dlog_{a,p}$ (b). (STALLINGS, 2005)

Supondo que dois usuários A e B desejem trocar informações sigilosas por um canal inseguro, DH mostra-se como uma solução que permite que ambos os usuários possam trocar uma chave secreta por um canal conhecido. A chave secreta compartilhada pode ser utilizada pelos usuários A e B para gerar criptogramas a partir de algoritmos criptográficos simétricos possibilitando uma comunicação sigilosa. A forma como as chaves são trocadas é descrita a seguir.

Os usuários A e B concordam e definem dois números públicos – que podem ser lidos por um intruso sem afetar a finalidade do algoritmo - *a* e *q*, onde *q* é um número primo e *a* é uma raiz primitiva de q.

Inicialmente o usuário A escolhe aleatoriamente um número inteiro $X_A < q$ e calcula $Y_A = a^{X_A}\ mod\ q$. Similarmente, o usuário B escolhe aleatoriamente um número inteiro $X_B < q$ e calcula $Y_B = a^{X_B}\ mod\ q$. Os números X_A e X_B serão, respectivamente, mantidos privados pelos usuários A e B; de forma contrária, os números Y_A e Y_B serão enviados ao outro usuário. A partir dessas informações o usuário A calcula a chave K_A como $K_A = (Y_B)^{X_A}\ mod\ q$ e o usuário B calcula a chave K_B como $K_B = (Y_A)^{X_B}\ mod\ q$. Pelas regras da

aritmética modular é possível concluir que KA = KB (STALLINGS, 2005) e ambas as chaves serão chamadas apenas de K. Este cenário é descrito na Figura 6 e simulado com valores reais na Tabela 2.

Tabela 2 - Valores aplicados no cenário da troca de chaves utilizando Diffie Hellman

Fonte: elaborado pelo autor (2011)

Valores públicos	
Número primo q = 353	Raiz primitiva a = 3
Valores calculados por A	Valores calculados por B
XA = 97	XB = 233
YA = 3^{97} mod 353 = 40	YB = 3^{233} mod 353 = 248
$K_A = (Y_B)^{X_A} \ mod \ q$ = 248^{97} mod 353 = 160	$K_B = (Y_A)^{X_B} \ mod \ q$ = 40^{233} mod 353 = 160

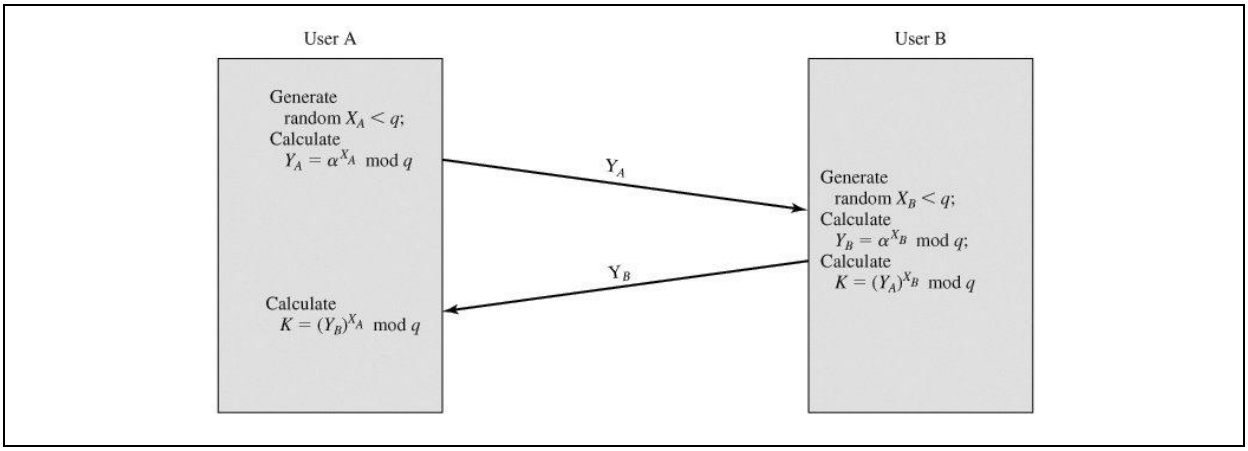

Figura 6 - Algoritmo Diffie-Helman

Fonte: (STALLINGS, 2005)

O resultado é que ambos os usuários A e B trocaram um valor secreto (chave K). Como X_A e X_B são privados, um intruso possui apenas as variáveis q, α, Y_A, e Y_B para tentar decifrar a chave. Dessa forma um intruso é forçado a calcular um logaritmo discreto para determinar a chave K. Por exemplo, para determinar a chave privada do usuário B um intruso deve ter que calcular inicialmente X_B a partir da fórmula $X_B = dlog_{a,q} (Y_B)$ e posteriormente calcular o valor de K da mesma forma que B calculou.

A segurança da troca de chaves de Diffie-Helman reside no fato de que, enquanto é fácil calcular um exponencial módulo número primo, é extremamente difícil calcular logaritmos discretos. Para números primos grandes, calcular os logaritmos discretos é considerado inviável (STALLINGS, 2005).

O Algoritmo de Diffie-Hellman é utilizado no protocolo IKE (*Internet Key Exchange*) (BURNNET e PAINE, 2001) e, indiretamente no protocolo IPSec que utiliza o protocolo IKE (TANENBAUM, 2003).

2.2.3 - CRIPTOGRAFIA ASSIMÉTRICA

Os algoritmos de criptografia assimétrica são também chamados de Algoritmos de Chave Pública. Os algoritmos de criptografia assimétrica utilizam duas chaves distintas e inversamente relacionadas, uma para criptografar (chave pública) e outra chave para decriptografar (chave privada) (vide Figura 7).

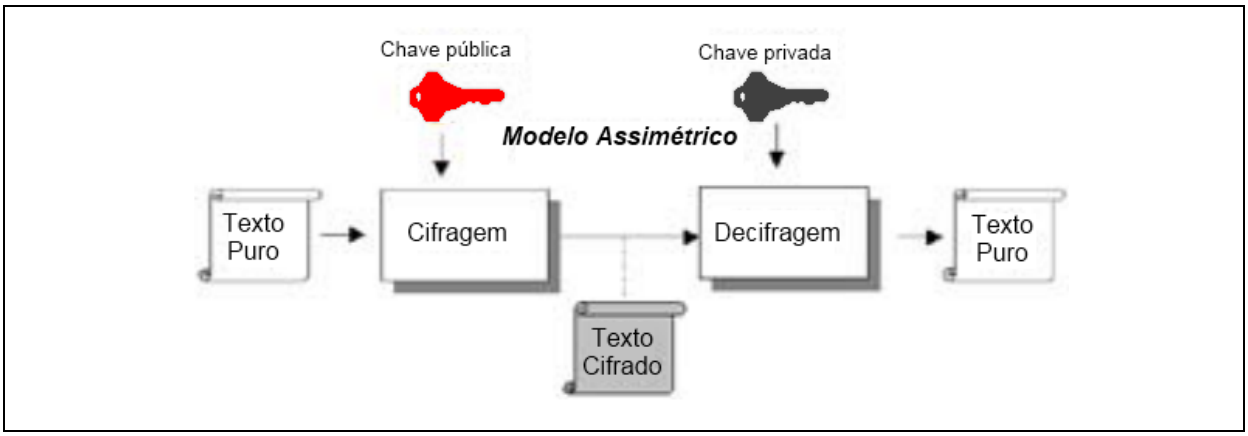

Figura 7 - Criptografia com chave assimétrica
Fonte: (MORENO, PEREIRA e CHIARAMONTE, 2005)
Nota: Adaptado pelo autor

A característica importante da criptografia assimétrica é o fato de que é computacionalmente inviável determinar a chave de decriptografia (chave privada) a partir do algoritmo e/ou a partir da chave de criptografia (chave pública) (STALLINGS, 2005). Isso torna possível que haja comunicação por um canal violável e mantenha-se o sigilo das mensagens.

Em uma situação que envolva troca de mensagens entre duas entidades A e B, ambos podem enviar sua respectiva chave pública por um canal violável. A partir do momento que A receber a chave pública de B, é possível criptografar uma mensagem utilizando a chave pública de B e enviar para B a mensagem criptografada; somente a chave privada de B (e que apenas B dispõe) é capaz de decriptografar a mensagem enviada por A. De forma recíproca,

caso B queira enviar uma mensagem para A, a lógica da comunicação é a mesma. Essa característica da criptografia assimétrica resolve o problema das distribuições de chaves presente na criptografia simétrica.

O fato de que as chaves públicas e privadas são relacionadas permitiu que a implementação de alguns algoritmos adicionasse outros benefícios à criptografia assimétrica. O algoritmo RSA, por exemplo, possibilita que a informação criptografada com a chave privada possa ser decriptografada com a chave publica e vice-versa (STALLINGS, 2005). Isso possibilita obter, além da propriedade Sigilo, as propriedades Autenticação e Não Repúdio.

Como exemplo de uma situação em que se obtém as propriedades Autenticação/Não Repúdio é possível imaginar que uma entidade A utiliza a sua chave privada para criptografar uma mensagem que é enviada à entidade B e, após receber a mensagem, a entidade B utiliza a chave pública de A para decriptografar; caso o conteúdo seja um conteúdo válido isso significa que a mensagem foi criptografada com a chave privada de A e consequentemente foi a entidade A que enviou a mensagem. Esse mecanismo pode funcionar como uma *Assinatura Digital*.

O exemplo citado anteriormente garante autenticidade mas não garante sigilo uma vez que outra pessoa pode ter acesso à chave pública de A e, será capaz de ler a mensagem. É possível obter, além de autenticação, o sigilo utilizando as chaves das duas entidades envolvidas na comunicação. A Figura 8 descreve um cenário em que um Emissor A realiza dois passos: (1) utiliza a sua chave privada (PrKA) para criptografar um texto puro X originando o texto cifrado Y e (2) criptografa Y com a chave pública da entidade B (PuKB) gerando o texto cifrado Z e, posteriormente envia Z para a entidade B. Após a entidade B receber o valor Z realiza os passos inversos: (2) utiliza sua chave privada (PrKB) para decifrar Z originando o texto Y e (1) utiliza a chave pública de A (PuKA) para decifrar Y obtendo o texto puro original X. É possível perceber que os passos de numeração 2 realizam o sigilo enquanto que os passos de numeração 1 são os passos necessários para se obter a autenticidade.

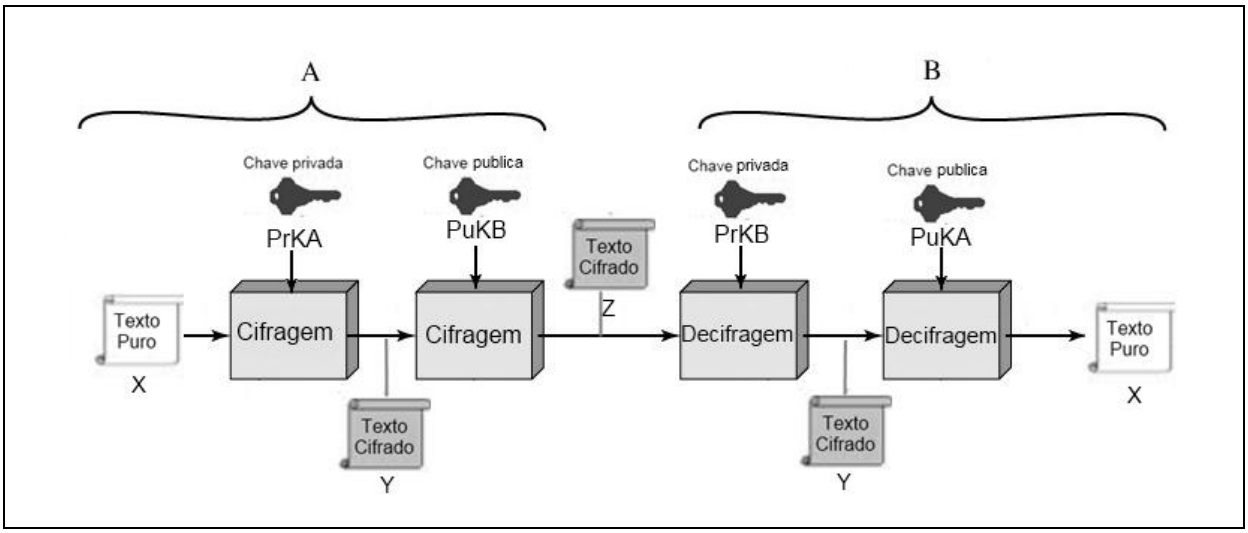

Figura 8 - Autenticação e Sigilo com Criptografia Assimétrica
Fonte: (MORENO, PEREIRA e CHIARAMONTE, 2005)
Nota: adaptado pelo autor

Deve-se destacar que algoritmos de chave pública resolvem o problema da autenticidade, mas têm uma complexidade computacional elevada. Sistemas de chave pública dependem da utilização de algum tipo de função matemática inversível e a complexidade do cálculo dessas funções pode não crescer linearmente com o número de bits na chave, mas crescer mais rapidamente do que isso (STALLINGS, 2005). O resultado prático disso é que "algoritmos de chaves públicas são lentos enquanto algoritmos simétricos podem criptografar e decriptografar dados de forma muito rápida" (BURNNET e PAINE, 2001).

Algoritmos de elevada complexidade computacional tornam-se problemáticos em situações onde a velocidade é essencial ou há um grande volume de dados a ser criptografado. A utilização de algoritmos de chave pública em dispositivos que não possuem grande poder computacional ou possuem restrições energéticas, como por exemplo, celulares ou *smart cards*, pode levar a um gasto excessivo de tempo e/ou gasto excessivo da bateria.

O algoritmo criptográfico para chave assimétrica mais famoso é o RSA. Foi desenvolvido em 1977 por Ron Rivest, Adi Shamir e Leonard Adleman. Contudo existem outros algoritmos criptográficos que utilizam o conceito de chaves públicas entre eles pode-se citar o algoritmo El Gamal e o Algoritmo de Curvas Elípticas.

A Tabela 3 cita as principais diferenças entre as técnicas de criptografia simétrica e a criptografia assimétrica:

Tabela 3 - Comparativo entre criptografia simétrica e assimétrica
Fonte: (STALLINGS, 2005)
Nota: adaptado pelo autor

Criptografia simétrica	Criptografia assimétrica
Necessita para funcionar	
Um mesmo algoritmo com a mesma chave é utilizado para criptografia e decriptografia.	Um mesmo algoritmo é usado para criptografar e decriptografar com um par de chaves, uma para criptografar e a outra para decriptografar.
O emissor e o destinatário devem compartilhar o algoritmo e a chave.	O emissor e o receptor devem ter cada um, um par de chaves correspondentes.
Necessário para manter a segurança	
A chave deve ser mantida secreta.	Uma das duas chaves deve ser mantida secreta.
Deve ser impossível, ou pelo menos, impraticável decifrar a mensagem se não houver nenhuma outra informação disponível.	Deve ser impossível ou pelo menos impraticável decifrar a mensagem se não houver nenhuma outra informação disponível.
Conhecimento sobre o algoritmo mais exemplos de texto cifrado deve ser insuficiente para determinar a chave.	Conhecimento sobre o algoritmo e o conhecimento de uma das chaves e exemplos de textos cifrados devem ser insuficientes para determinar a outra chave.
Desempenho	
Rápido	Lento

2.2.4 - CRIPTOGRAFIA POR HARDWARE

Soluções de criptografia em hardware normalmente são realizados a partir de aceleradores criptográficos e dispositivos *Hardware Security Module* (HSM) que incluem *tokens* de autenticação, cartões inteligentes e dispositivos biométricos. Os algoritmos criptográficos implementados em hardware implicam em ganho de velocidade e os implementados em software em software implicam em ganho de flexibilidade (TANENBAUM, 2003). Contudo, soluções em hardware além de se obter velocidade propiciam diversos ganhos de segurança conforme será visto adiante.

CRIPTOGRAFIA A PARTIR DE CIRCUITOS ELÉTRICOS SIMPLES

As transposições e substituições podem ser implementadas com circuitos elétricos simples. Em (TANENBAUM, 2003) é demonstrado um circuito elétrico que realiza permutação de bits denominado caixa P. A Figura 9(a) mostra a caixa P que realiza uma transposição em uma entrada de 8 bits. Se os 8 bits forem designados de cima para baixo

como 01234567, a saída dessa caixa P específica será 36071245. Com uma fiação interna adequada, pode-se criar uma caixa P para executar qualquer transposição praticamente na velocidade da luz, pois nenhuma computação é envolvida, apenas a propagação e sinais. Esse projeto segue o princípio de *Kerckhoff*: o atacante sabe que o método geral é permutar os bits. O que ele não sabe é qual bit fica em cada posição, e isso é a chave (TANENBAUM, 2003).

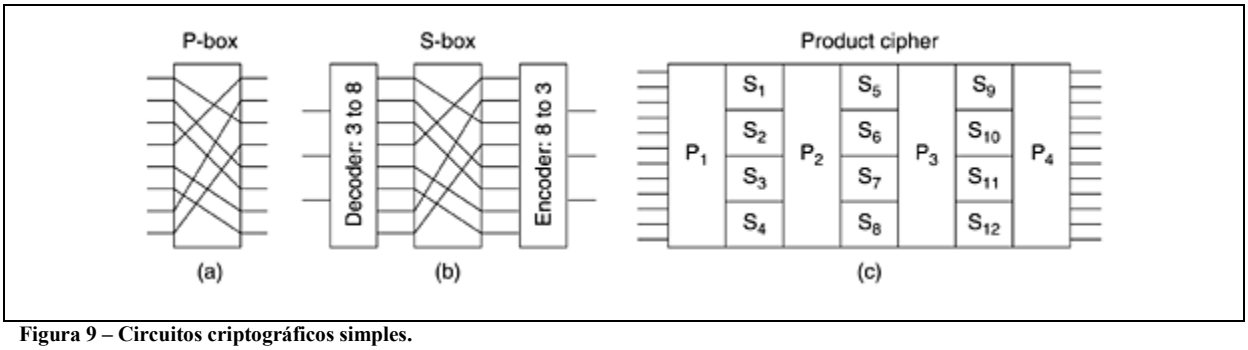

Figura 9 – Circuitos criptográficos simples.
Fonte: (TANENBAUM, 2003)

Em (TANENBAUM, 2003) também é demonstrado um circuito elétrico (caixas S) utilizado para realizar substituições de bits. Tal circuito contém 3 estágios e é representado na Figura 8.6(b). É possível imaginar o estágio 1 (*Decoder*) como um multiplexador que seleciona uma das oito trilhas de saída do primeiro estágio e a define como 1 e todas as demais como 0; o segundo estágio é uma caixa P que irá permutar o resultado dos bits do primeiro estágio; o terceiro estágio (*Encoder*) pode ser entendido como um demultiplexador que irá codificar a trilha selecionada novamente em três bits. Por exemplo, se for introduzido o número 5 (101 em binário), no primeiro estágio será selecionado a sexta trilha; o segundo estágio irá permutar o bit da sexta posição para a posição 8; o terceiro estágio irá demultiplexar o binário 10000000 gerando o correspondente número 7 (111 em binário). É possível notar que fazendo o caminho inverso obtém-se novamente o número 5. De acordo com o circuito exposto, se os oito números octais 01234567 fossem introduzidos um após o outro, a sequencia de saída seria 24506713, ou seja, cada número seria substituído por outro.

A capacidade real desses elementos básicos se torna aparente quando dispomos uma série inteira de caixas em cascata para formar uma cifra de produto, como mostra a Figura 8.6(c). Nesse exemplo, 12 linhas de entrada são transpostas (isto é, permutadas) pelo primeiro estágio (P1). Teoricamente, seria possível fazer com que o segundo estágio fosse uma caixa S

que mapeie um número de 12 bits em outro número de 12 bits. No entanto, tal dispositivo necessitaria de $2^{12} = 4096$ fios cruzados em seu estágio intermediário. Em vez disso, a entrada é dividida em quatro grupos de 3 bits, sendo que cada um deles é substituído de forma independente dos outros. Apesar de ser menos genérico, esse método ainda é mais eficiente. Através da inclusão de um número de estágios suficientemente grande na cifra de produto, a saída pode ser transformada em uma função excessivamente complicada da entrada. (TANENBAUM, 2003)

ACELERADORES CRIPTOGRÁFICOS E MÓDULOS DE SEGURANÇA EM *HARDWARE* (HSM)

Aceleradores de criptografia funcionam como coprocessadores matemáticos que normalmente implementam vários algoritmos e protocolos. Aceleradores de criptografia oferecem um grande aumento de velocidade reduzindo a carga sobre a CPU do sistema e, podem ser acoplados a um barramento de expansão do computador, como por exemplo, barramento PCI (*Peripheral Component Interconnect*). (BURNNET e PAINE, 2001)

Figura 10 - Acelerador criptográfico Sun Crypto Accelerator 6000
Fonte: http://reviews.cnet.com/i-o-cards/sun-crypto-accelerator-6000/1707-3019_7-33595776.html (acessado em 01/02/2010)

Outra razão para a popularidade de aceleradores de criptografia é o fato de que estes possuem certificações de qualidade. O Instituto Nacional de Padrões e Tecnologia (NIST) do governo americano, por exemplo, já certificou diversos deles. A certificação de cada aparelho depende das garantias que foram implementadas durante a fabricação. (BURNNET e PAINE, 2001). A Figura 10, por exemplo, mostra um acelerador criptográfico que possui certificação

FIPS-2[7] nível 3 e é capaz de realizar conexões SSL suportando RSA, DH, 3DES e outros algoritmos criptográficos.

HSM (*Hardware Security Module*) é um tipo de processador criptográfico seguro orientado para o gerenciamento de chaves digitais, acelerando processos de criptografia em termos de assinaturas digitais e fornecendo autenticação forte para acesso às chaves críticas para aplicativos de servidor. Também podem atuar como aceleradores criptográficos, mas têm como objetivo principal a proteção de chaves criptográficas de alto valor. Os dispositivos HSM podem variar de simples *tokens* USB, *smart cards* ou a *hardwares* de processamentos poderosos e tão seguros que até mesmo uma variação térmica pode implicar na exclusão de todas as chaves armazenadas; a Figura 11 contém um dispositivo HSM utilizado pelo ITI (Instituto Nacional de Tecnologia da Informação) que atende ao padrão de segurança FIPS-2 nível 4 e se enquadra nesta situação.

Figura 11 - HSM utilizado na AC Raiz ICP-Brasil
Fonte: http://kryptus.com/site/index.php?option=com_content&task=view&id=12&Itemid=32

PROCESSADORES COM SUPORTE A INSTRUÇÕES CRIPTOGRÁFICAS

Conforme já visto, a necessidade de criptografia tem tornado cada vez mais comum não apenas para grandes empresas, mas também para usuários domésticos. Objetivando tornar o uso da criptografia mais rápida para aplicações comuns, alguns processadores de propósito geral destinados a usuários domésticos, recentemente, passaram a implementar instruções específicas de criptografia.

Recentemente a Intel passou a implementar o conjunto de instruções conhecido como AES-NI ou *AES New Instructions* em alguns processadores das famílias i5 (ex.: Intel Core i5-

[7] O padrão FIPS-2 será comentado no Capítulo 3 - Normas e padrões

661) e i7 (ex.: Intel Core i7-920), possibilitando a utilização do algoritmo AES em nível de instrução do processador. A AMD pretende implementar esse conjunto de instruções a partir da família de processadores conhecida como Bulldozer; espera-se que estejam à venda ainda em 2011.

A Microchip, uma das principais empresas que desenvolvem microcontroladores no mundo, desde 2004 comercializa os microcontroladores PIC12F635 e PIC16F636 que são microcontroladores de 8 bits que possuem um periférico criptográfico integrado chamado KEELOQ. O KEELOQ é uma tecnologia proprietária que provê uma solução completa para autenticação e segurança remota de sistemas a partir de um algoritmo criptográfico próprio e é atualmente utilizada por sistemas automobilísticos de grandes empresas como Chrysler, Fiat, GM, Honda, Toyota, Volvo, Volkswagen Group, Jaguar e outras.

2.4 - ENVELOPE DIGITAL

É sabido que os algoritmos de criptografia simétricos são muito mais rápidos que os algoritmos de criptografia assimétricos, porém o sigilo em uma comunicação que utiliza criptografia simétrica depende de uma Distribuição de Chaves segura e, isso não é algo fácil de ser realizado. Por outro lado a criptografia assimétrica resolve o problema da Distribuição de Chaves, mas é centenas de vezes mais lenta[8].

É possível a realização de uma comunicação sigilosa utilizando as técnicas de criptografia simétrica e assimétrica aproveitando o melhor de cada uma das técnicas. Para uma comunicação que envolva criptografia duas coisas são necessárias: a chave de criptografia e o conteúdo (texto puro) da mensagem. O conteúdo da mensagem é, via de regra, muito menor do que a chave de criptografia, dessa forma utiliza-se a técnica lenta (criptografia assimétrica) para criptografar a chave e, a técnica rápida (criptografia simétrica) para criptografar o conteúdo da mensagem.

[8] Em (BURNNET e PAINE, 2001) é citado que "dependendo da plataforma, alguns algoritos simétricos podem operar a velocidades de 10MB, 20MB, 50MB (e às vezes mais por segundo). Em contraste, um algoritmo de chave pública opera provavelmente a 20KB a 200KB por segundo, dependendo do algoritmo, plataforma e outros fatores."

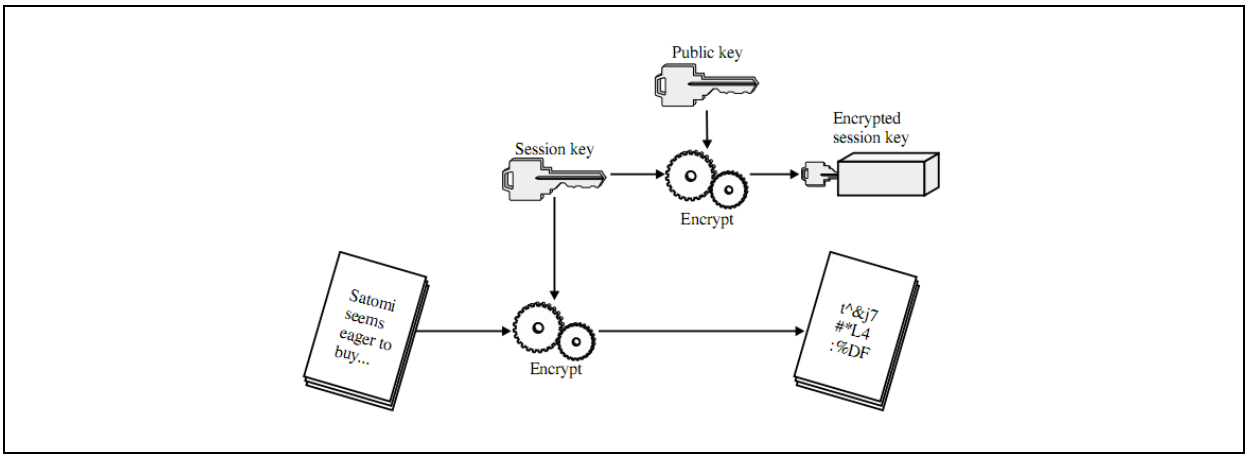

Figura 12 - Envelope Digital 1
Fonte: (BURNNET e PAINE, 2001)

A Figura 12 descreve um cenário de utilização de um Envelope digital. Um Emissor cria uma chave aleatória denominada Chave de Sessão (*Session Key*) que poderá ser utilizada apenas em um intervalo de tempo ou armazenada para se comunicar única e exclusivamente com o mesmo Receptor. O Emissor criptografa um documento com um algoritmo simétrico e utiliza como chave a Chave de Sessão; posteriormente a Chave de Sessão é criptografada com um algoritmo de criptografia assimétrico sendo utilizado como chave a Chave Publica (*Public Key*) do Receptor.

O processo de criptografar os dados (texto puro) utilizando uma chave de criptografia simétrica e criptografar a chave de criptografia simétrica utilizando um algoritmo de chave pública é denominado Envelope Digital (BURNNET e PAINE, 2001).

2.5 - SUMÁRIO DE MENSAGEM (*MESSAGE DIGEST*)

Os envelopes digitais permitem que sejam trocadas informações sigilosas e que seja possível verificar a identidade do emissor da mensagem (autenticidade). Contudo os envelopes digitais, quando utilizados da forma descrita aqui, não garantem a integridade da mensagem e não garantem o não repúdio, uma vez que as mensagens trocadas são criptografadas com uma chave simétrica.

Os métodos de assinatura possibilitam autenticação e não repúdio; os envelopes digitais reúnem, adicionalmente às assinaturas, o sigilo. Em geral, a autenticação é necessária,

mas o sigilo não. Como a criptografia é lenta, em geral as pessoas preferem enviar textos simples assinados (TANENBAUM, 2003).

A garantia da integridade de uma mensagem independentemente do sigilo pode ser obtido a partir de uma função *hash* unidirecional que extrai um trecho qualquer do texto simples e, a partir dele, calcula uma *string* de tamanho fixo.

O propósito de uma função *hash* H é produzir uma "impressão digital" de um arquivo, mensagem ou um bloco de dados qualquer. Para ser útil a função *hash* deve possuir as seguintes propriedades: (STALLINGS, 2005)

- A função pode ser aplicada a um bloco de dados de qualquer tamanho.
- A função produz um resultado de tamanho fixo.
- A função deve ser relativamente fácil de ser executada para qualquer dado, tornando a implementação tanto por software quanto por hardware prática.
- Para um dado valor h, é computacionalmente inviável procurar x tal que H(x) = h. Essa propriedade é referenciada na literatura como propriedade de caminho único.
- Para um bloco de dados x qualquer, é computacionalmente não factível obter y tal que H(x) = H(y). Isso é algumas vezes referenciado como "resistência a ataques de colisão".
- É computacionalmente infactível encontrar um par (x,y) tal que H(x) = H(y). Isso é às vezes referenciado como forte resistência a colisões.

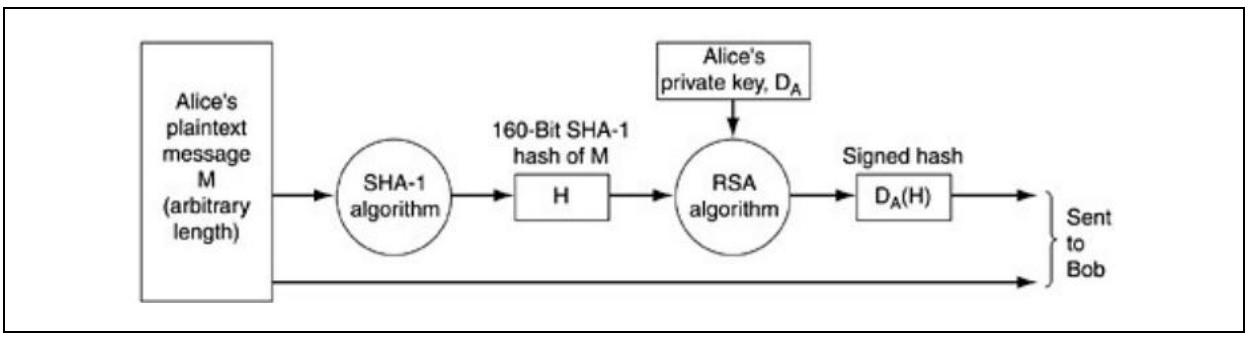

Figura 13 – Sumário de mensagens utilizando SHA-1 e RSA para assinar mensagens não secretas

Fonte: (TANENBAUM, 2003)

A Figura 13 mostra um cenário de utilização de Sumário de Mensagens. Inicialmente um emissor (Alice) aplica a função de *hash* a um texto simples M e, obtém o *hash* da mensagem H; após obter o *hash* Alice criptografa o *hash* H com sua chave privada. Quando o receptor (Bob) receber a mensagem utilizará a chave pública de Alice para obter o *hash* H; posteriormente poderá calcular um *hash* H2, utilizando o texto puro que recebeu e o mesmo algoritmo. A partir da comparação entre H e H2 o receptor pode identificar se a mensagem está íntegra (caso H seja idêntico a H2) ou se foi alterada. O cenário descrito contempla os serviços de autenticidade e não repúdio, uma vez que, é possível identificar se o emissor (Alice) assinou ou não a mensagem enviada.

O termo utilizado para descrever uma situação em que duas mensagens produzem o mesmo resumo é chamado de colisão. Uma colisão ocorre quando uma segunda mensagem produz o mesmo resumo de uma mensagem anterior, ou quando duas mensagens produzem o mesmo sumário. Isso é possível porque o tamanho de um sumário de mensagem é relativamente pequeno; por exemplo; para um *hash* que produz um resultado de 16 bytes ou 128 bits existem 2^{128} possibilidades (BURNNET e PAINE, 2001). Essa característica pode ser utilizada para se realizar o que se chama de ataques de colisão onde é possível encontrar uma mensagem que produza um *hash* equivalente.

Existem diversos algoritmos de *hash*, mas três deles são os mais utilizados: MD2, MD5 e SHA-1 (BURNNET e PAINE, 2001). Atualmente o algoritmo SHA-2 é utilizado como substituto natural[9] do SHA-1 e vem sendo aconselhado em detrimento dos demais por não ser susceptível a ataques de colisão. Como o algoritmo SHA-2 permite diferentes parâmetros (tamanho de resultado, tamanho de bloco, etc.) ele é também referenciado a partir do número de *bits* do resumo calculado, ou seja, quando o SHA-2 gera 256 *bits* de resultado ele é também referenciado como SHA-256, quando gera 512 *bits* de resultado é também referenciado como SHA-512 e assim sucessivamente. A Tabela 4 apresenta algumas diferenças entre esses algoritmos.

[9] Espera-se que em 2012 seja lançada a versão SHA-3.

Tabela 4 - Algoritmos de sumário de mensagens

Fonte: http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf (acessado em 14/03/2011)

Nota: Adaptado pelo autor

Algoritmo	Tamanho resultado	Tamanho do bloco	Complexidade ataque de colisão
MD2	16 bytes	16 bytes	2^{63}
MD5	16 bytes	64 bytes	$2^{20.96}$
SHA-1	20 bytes	64 bytes	2^{51}
SHA-224	28 bytes	64 bytes	-
SHA-256	32 bytes	64 bytes	-
SHA-384	48 bytes	128 bytes	-
SHA-512	64 bytes	128 bytes	-

2.6 - CERTIFICADO DIGITAL

A partir da utilização de algoritmos de criptografia assimétricos é possível se obter sigilo, autenticidade e não repúdio entre um receptor e um emissor. Um envelope digital utiliza os algoritmos simétricos e assimétricos para permitir obter sigilo, autenticidade e não repúdio de uma forma mais eficiente. Os sumários de mensagem possibilitam obter assinaturas digitais (autenticidade e não repúdio) e validar a integridade de forma eficiente. A partir da composição desses conceitos é possível obter sigilo, autenticidade, não repúdio e integridade de forma eficiente desde que se saiba qual é a chave pública na parte envolvida na comunicação.

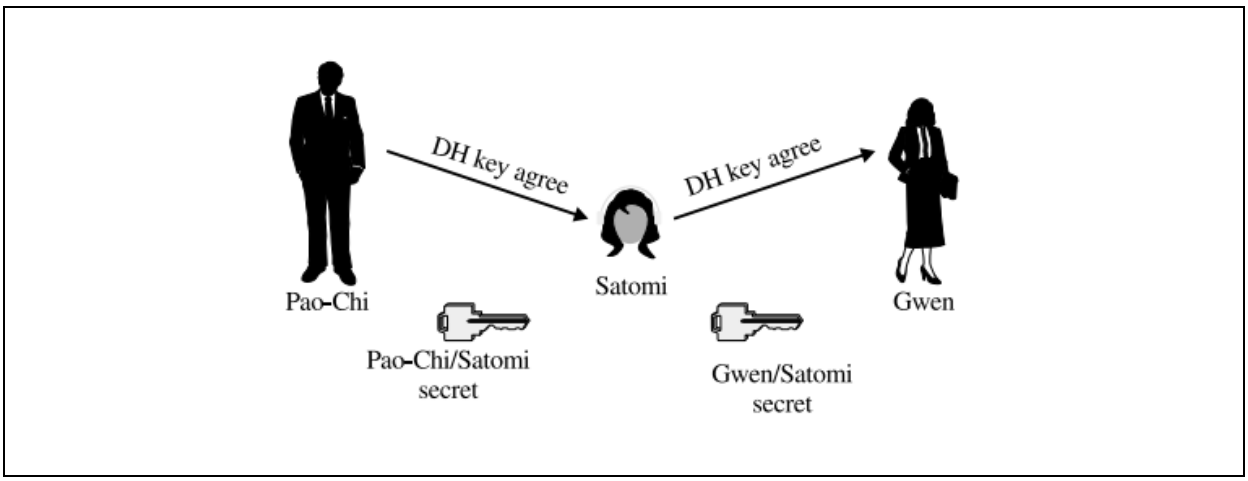

Figura 14 - Ataque de homem do meio (*man-in-the-middle attack*) ou ataque brigada de incêncio

Fonte: (BURNNET e PAINE, 2001)

A comunicação por criptografia de chaves públicas está sujeita ao ataque conhecido como ataque do homem do meio. A Figura 14 descreve um cenário em que Pao-Chi deseja se comunicar com Gwen e a intrusa Satomi deseja obter acesso ao conteúdo da comunicação. Esse cenário é descrito pelos passos abaixo:

- Pao-Chi criptografa uma mensagem com a chave pública de Gwen (que na verdade é da intrusa Satomi);
- Satomi intercepta e lê a mensagem;
- O impostor criptografa a mensagem recebida com a verdadeira chave de Gwen e envia a mensagem para ela;
- Gwen pode ler a mensagem com sua chave privada;
- Nem Pao-Chi, nem Gwen ficarão sabendo que estão sendo espionados e a comunicação entre eles não é mais confidencial.

Na distribuição simétrica um intruso deve interceptar a troca de chaves para ter acesso ao conteúdo; na criptografia assimétrica o intruso deve criar uma chave nova e se passar por uma das partes na comunicação atuando como um intermediário. Sendo assim, é possível perceber que o problema da distribuição de chaves presente na criptografia simétrica persiste, ainda que de forma diferenciada, na distribuição assimétrica. Outro problema presente na criptografia assimétrica é o fato de que se a chave privada de uma determinada instituição (ou pessoa) for descoberta por um intruso, este poderá assinar documentos como se fosse aquele. O conceito de certificado digital objetiva solucionar o problema da distribuição de chaves públicas e da validade das assinaturas.

Um certificado digital é um documento digital emitido por uma Autoridade Certificadora (também chamada de Autoridade de Certificação - um terceiro confiável que pode ser uma instituição financeira, por exemplo). O certificado digital inclui o nome do sujeito (companhia ou indivíduo que está sendo certificado), a chave pública do sujeito, um número serial, uma data de validade, a assinatura da autoridade de certificação responsável e quaisquer outras informações relevantes. A autoridade de certificação assina o certificado,

criptografando a chave publica do sujeito e um valor *hash* da chave pública deste sujeito, por meio da própria chave privada. (DEITEL, DEITEL e STEINBUHLER, 2004).

No Brasil pode-se destacar como autoridades certificadoras o SERASA, a Receita Federal, AC-Jus (Autoridade Certificadora da Justiça), ACPR (Autoridade Certificadora da Presidência da República), Certsign e outras.

Os certificados digitais são muito utilizados pelos *browsers*. Quando um browser acessa um sistema bancário, por exemplo, utiliza o protocolo HTTPS que permite uma conexão segura a um site e utiliza o conceito de certificado digital. O *browser* é pré-carregado pelo fabricante com um conjunto de certificados digitais de autoridades certificadoras consideradas confiáveis, podendo o usuário adicionar ou remover certificados; a Figura 15 exibe a tela do Internet Explorer 8 que possibilita configurar os certificados digitais.

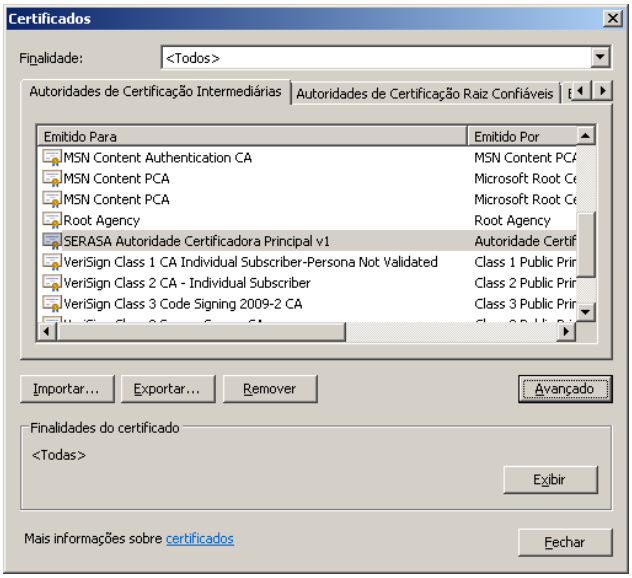

Figura 15 - Tela de Certificados e Autoridades de certificação no Internet Explorer 8
Nota: Elaborado pelo autor (2011)

O certificado digital por si só não resolve todos os problemas. Por exemplo, um intruso pode criar um certificado digital assinado com sua própria chave pública fazendo-se passar por uma Autoridade Certificadora. Dessa forma, para que um site ou documento possua um certificado digital confiável o certificado deve ser emitido por uma autoridade certificadora confiável sob o ponto de vista do usuário. Na prática, toda a confiança se baseia

na ideia de uma rede de confiança que é implementada através de infraestrutura de chaves públicas.

2.7 - INFRAESTRUTURA DE CHAVES PÚBLICAS (ICP)

A Autoridade Certificadora (AC) é a instituição confiável que deverá validar a identidade do dono de um certificado digital através de sua assinatura. Isso significa que se a autoridade certificadora assina um certificado a AC validou a identidade do sujeito utilizando suas próprias políticas de validação e, cabe ao usuário, confiar naquela AC ou não. No mundo real confia-se em diversas instituições públicas, como por exemplo, o Detran que emite a Carteira Nacional de Habilitação ou, a Receita Federal que emite o CPF e assim por diante.

No mundo virtual, da mesma forma que no mundo real, existem diversas instituições "confiáveis". Cabe destacar o termo confiável porque o mesmo é altamente subjetivo e dependente dos critérios adotados por cada tipo de usuário. Uma instituição financeira brasileira, por exemplo, pode não confiar em um certificado assinado por uma empresa russa por não conhecê-la, contudo, esta mesma empresa pode ser considera altamente confiável por cidadãos daquele país.

Ainda que uma determinada AC seja considerada confiável em todo o mundo, faz-se necessário a existência de diversas instituições. Uma única AC entraria em colapso sob a carga e também seria um ponto central de falha. Uma solução possível poderia ser a existência de várias AC's, todas administradas pela mesma organização e todas usando a mesma chave privada para assinar certificados. Embora isso pudesse resolver o problema da carga e da falha, há um novo problema: o vazamento de chaves.

Se houvesse dezenas de servidores espalhados pelo mundo, todos com a chave privada da AC, a chance de que a chave privada fosse roubada ou sofresse algum outro tipo de vazamento seria bastante aumentada. Tendo em vista que o comprometimento dessa chave arruinaria a infraestrutura de segurança eletrônica do mundo, a existência de uma única AC central é muito arriscada (TANENBAUM, 2003). Os motivos expostos levaram à criação de

uma infraestrutura de chaves públicas (PKI – *Public Key Infrastructure*) baseada na ideia de rede de confiança.

A *Public Key Infrastructure* – PKI (ou Infraestrutura de chave pública) integra a criptografia de chave pública aos certificados digitais e às autoridades de certificação para autenticar as partes de uma transação (DEITEL, DEITEL e STEINBUHLER, 2004), ou dito de outra forma, uma infraestrutura de chaves públicas (ICP ou PKI) é um conjunto de hardware, software, pessoas e políticas, e procedimentos necessários para criar, gerenciar, armazenar, distribuir e revogar certificados digitais baseados em criptografia assimétrica.

O objetivo principal para desenvolver um ICP é possibilitar a aquisição segura, conveniente e eficiente de chaves públicas (STALLINGS, 2005). Como exemplos de Infraestrutura de chaves públicas é possível citar o modelo PKIX desenvolvido pelo IETF e o modelo ICP-Brasil regulamentado pela através da Medida Provisória 2200-2 (24/08/01).

2.7.1 ICP-BRASIL

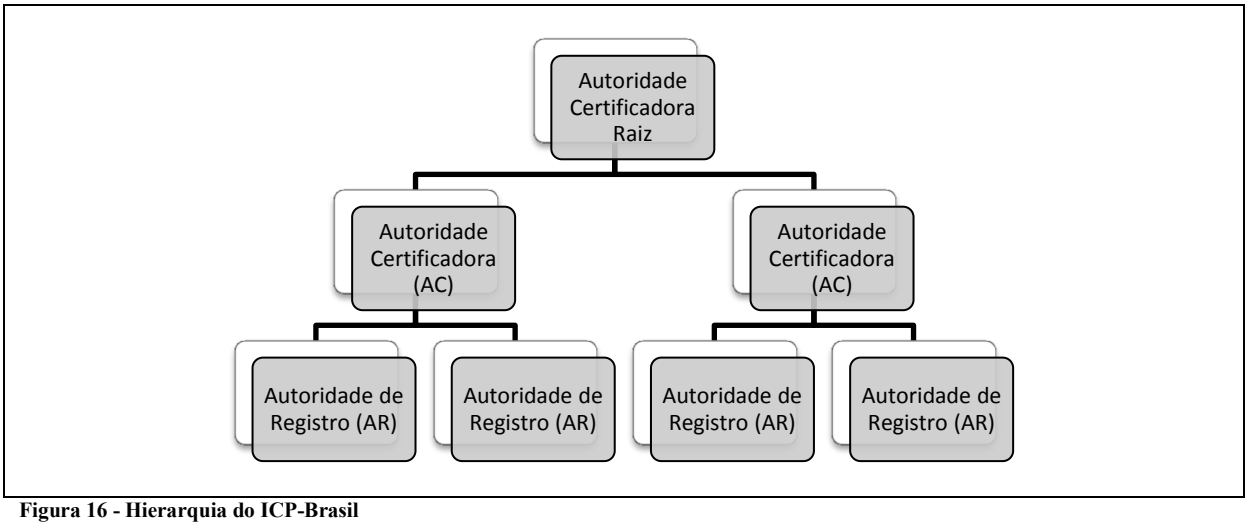

Figura 16 - Hierarquia do ICP-Brasil
Fonte: elaborado pelo autor (2011)

A Figura 16 mostra a estrutura hierárquica de autoridades do ICP-Brasil. No topo da hierarquia há uma autoridade certificadora denominada Autoridade Raiz tendo como subordinadas as Autoridades Certificadoras (AC) e estas tendo como subordinadas as Autoridades de Registro (AR).

A AC Raiz é o ITI (Instituto Nacional de Tecnologia da Informação) e é responsável por executar as políticas e as normas técnicas e operacionais para os certificados brasileiros, portanto, ela tem como principal função expedir, distribuir, gerenciar e fiscalizar os certificados das autoridades certificadoras que estão abaixo do seu nível hierárquico (AC's e AR's). A AC é composta por entidades credenciadas pela AC-Raiz para emitir certificados digitais associando as chaves criptográficas aos seus respectivos donos. A Autoridade de Registro (AR) é composta por entidades vinculadas operacionalmente a uma determinada Autoridade Certificadora (AC); sua função é identificar e cadastrar os usuários em postos de atendimento, onde os mesmos possam comparecer, e a partir daí encaminhar as solicitações de certificados para uma AC. (CERTISIGN, 2007)

O certificado da AC Raiz é auto assinado. O certificado de cada uma das AC's intermediária é assinado pela AC imediatamente superior, o certificado da entidade final é assinado pela AC imediatamente superior na cadeia e assim sucessivamente. Embora o ICP-Brasil seja composto por três níveis hierárquicos uma infraestrutura de chaves públicas pode conter quantos níveis hierárquicos o modelo julgar necessário.

A Autoridade Certificadora mantém um repositório (também chamado de diretório) com todos os certificados emitidos, pois para validar uma assinatura pode ser necessário acessar um certificado muito tempo depois que ele tiver expirado. Os diretórios também podem ser utilizados para armazenar a LCR-Lista de Certificados Revogados (ou CRL-*Certificate Revocation Lists*), pois a autoridade que concede um certificado pode decidir revogá-lo, entre outros motivos, porque a pessoa ou organização que possui o certificado cometeu algum abuso, a chave privada foi exposta ou, pior ainda, se a chave privada da CA foi comprometida. Dessa forma, as aplicações que aceitam certificados de uma determinada AC devem verificar a lista de certificados revogados. OCSP (*Online Certificate Status Protocol*) é um protocolo utilizado para obter a lista de certificados revogados de uma ICP.

A partir[10] de novembro de 2009 a Microsoft fechou um acordo com o ITI passando a distribuir junto com seus programas a cadeia de certificados ICP-Brasil. Isso significa que o Windows reconhece a Autoridade Certificadora Raiz (ITI) como uma entidade confiável e,

[10] http://www.microsoft.com/latam/presspass/brasil/2009/novembro/sites_seguros.mspx

dessa forma certificados assinados por uma Autoridade Certificadora subordinados ao ITI, como por exemplo SERPRO, são considerados certificados confiáveis pelo Microsoft Windows e pelo Internet Explorer.

CARIMBO DE TEMPO

O certificado digital, junto com as técnicas de criptografia e *hash* podem garantir a autenticidade, integridade, sigilo, autenticidade e não repúdio. Contudo há diversas situações em que a validade jurídica de um documento assinado está atrelada à data e à hora que foi assinado. Por exemplo, uma seguradora de automóveis, por má fé, pode se negar a pagar o conserto de um carro alegando que o seguro foi assinado posteriormente a um acidente. Outras situações incluem prazos para executar atos processuais definidos em lei, transações envolvendo compra de ações, validade de atestados médicos e assim por diante.

É sabido que uma mensagem assinada pode conter como conteúdo qualquer texto, incluindo uma data e hora. O problema de o próprio emissor da mensagem especificar a data e hora que a mesma foi escrita reside no fato de que o assinante pode utilizar uma data retroativa para obter ganhos pessoais ou simplesmente porque a data/hora do seu computador encontra-se errada. Para evitar essa situação faz necessário um mecanismo confiável de carimbo de tempo.

Um carimbo do tempo aplicado a uma assinatura digital ou a um documento prova que ele já existia na data incluída no carimbo do tempo. Os carimbos de tempo são emitidos por terceiras partes confiáveis, as Autoridades de Carimbo do Tempo (ACT's), cujas operações devem ser devidamente documentadas e periodicamente auditadas pela própria AC-Raiz da ICP-Brasil (BRASIL, 2010).

O carimbo agrega ao conteúdo do resumo do arquivo eletrônico (*hash*) a data/hora emitida por uma ACT – Autoridade de Carimbo do Tempo. O *Time-Stamp Protocol* (TSP) definido pela RFC 3161 especifica como deve ser a comunicação existente entre um assinante e uma ACT-Autoridade de Registro de Tempo.

São ACT's reconhecidas pelo ICP-Brasil Certisign, Autoridade de Carimbo do Tempo Brasileira de Registros (ACT BR), ACT Notarial, BRy ACT entre outras.

CERTIFICADOS DIGITAIS

Os certificados digitais podem ser armazenados em diferentes mídias, assim como criptografados com diferentes algoritmos com diferentes tamanhos de chave. Cada um desses requisitos poderá influenciar na segurança da conferência da validade do certificado. Como diferentes tipos de negócio têm diferentes necessidades de segurança e custo, o ICP-Brasil especificou diferentes tipos de certificado para atender a diferentes necessidades.

A ICP-Brasil classifica os certificados digitais em três tipos: tipo A, tipo S e tipo T. Os certificados do tipo S são direcionados para atividades sigilosas e podem ser classificados em S1, S2, S3 e S4. Os demais tipos (A e T) são direcionados para fins de assinatura digital, identificação e autenticação. Os certificados tipos A podem ser classificados em A1, A2, A3 e A4 enquanto que os certificados do tipo T se classificam em T3 e T4. A Tabela 5 resume as principais características que diferenciam os tipos de certificados.

Tabela 5 – Tabela Comparativa com Requisitos Mínimos por Tipo de Certificado
Fonte: (BRASIL, 2010)

Tipo	Chave Criptográfica			Validade máxima (anos)	Freqüência de emissão de LCR (horas)	Tempo limite revogação (horas)
	Tamanho (bits)	Gerado por	Mídia Armazenadora			
A1 e S1	1024	SW	Repositório (ex.: disco rígido) protegido por senha e/ou identificação biométrica, cifrado por software.	1	6	12
A2 e S2	1024	SW	Cartão Inteligente ou Token, ambos sem capacidade de geração de chave e protegidos por senha e/ou identificação biométrica.	2	6	12
A3 e S3	1024	HW	Cartão Inteligente ou Token, ambos com capacidade de geração de chave e protegidos por senha e/ou identificação biométrica, ou hardware criptográfico aprovado pelo CG da ICP-Brasil.	3	6	12

A4 e S4	2048	HW	Cartão Inteligente ou Token, ambos com capacidade de geração de chave e protegidos por senha e/ou identificação biométrica, ou hardware criptográfico aprovado pelo CG da ICP-Brasil.	3	6	12
T3	1024	HW	Hardware criptográfico aprovado pelo CG da ICP-Brasil.	3	6	12
T4	2048	HW	Hardware criptográfico aprovado pelo CG da ICP-Brasil.	3	6	1

ASSINATURA DIGITAL

Uma assinatura eletrônica representa um conjunto de dados, em formato eletrônico, que é anexado ou logicamente associado a outro conjunto de dados, também em formato eletrônico, para conferir-lhe autenticidade ou autoria. A assinatura eletrônica, portanto, pode ser obtida por meio de diversos dispositivos ou sistemas, como *login*/senha, biometria, impostação de *Personal Identification Number* (PIN) etc. (ITI, 2010)

Um dos tipos de assinatura eletrônica é a assinatura digital, que utiliza um par de chaves criptográficas associado a um certificado digital. Uma das chaves – a chave privada – é usada durante o processo de geração de assinatura e a outra – chave pública, contida no certificado digital – é usada durante a verificação da assinatura. O conjunto de normativos da ICP-Brasil trata, apenas, das assinaturas digitais geradas no âmbito da ICP-Brasil. Os demais tipos de assinaturas eletrônicas estão fora do seu escopo. (ITI, 2010)

No contexto das normas ICP-Brasil, as assinaturas digitais são produzidas com a utilização de chaves criptográficas privadas associadas a certificados digitais ICP-Brasil, seguindo os passos (ITI, 2010):

a) o signatário gera um resumo criptográfico de um documento eletrônico;

> *b) o signatário cifra o resumo criptográfico com sua chave privada, associada a uma chave pública constante do seu certificado digital, gerando a assinatura digital;*
>
> *c) o documento eletrônico e a assinatura digital ficam associados para futura validação.*

Um possível destinatário deve verificar a legitimidade da assinatura digital, que de forma simplificada, pode ser realizada pelos seguintes passos: (ITI, 2010)

> *a) o documento eletrônico e a assinatura digital associada são disponibilizados para o verificador, juntamente com o certificado digital do signatário.*
>
> *b) o verificador calcula novamente o resumo criptográfico do documento eletrônico;*
>
> *c) o verificador decifra a assinatura digital com a chave pública do signatário, contida no certificado digital, obtendo o resumo criptográfico gerado e cifrado pelo signatário no momento da assinatura;*
>
> *d) o verificador compara os resumos criptográficos obtidos nos passos b) e c). Se forem iguais, significa que o documento eletrônico esta íntegro e que é possível identificar o signatário por meio do certificado digital. Caso contrário, a assinatura digital é inválida.*

Há diversos padrões para arquivos assinados digitalmente, porém o ICP-Brasil especificou que documentos assinados devem utilizar o padrão CAdES (CMS Advanced Electronic Signature) ou o padrão XAdES (XML-DSig Advanced Electronic Signature) (ITI, 2010). Ambos serão comentados no Capítulo 3 – Normas e padrões.

2.8 - PROTOCOLOS DE COMUNICAÇÃO SEGUROS

Diversos protocolos de segurança são utilizados atualmente para permitir uma comunicação segura entre dois pares nas diversas camadas do modelo OSI. Entre os mecanismos de comunicação segura mais utilizados destacam-se os protocolos SSL (*Secure Socket Layer*) atualmente também chamado de TLS (*Transport Layer Secure*), os protocolos IPSec, VPN (Virtual Private Network) e outros. Nesta seção será descrito o SSL por ser o protocolo mais comumente utilizado nas aplicações Web.

2.8.1 - *SECURE SOCKET LAYER* (SSL)

A *Secure Socket Layer (*SSL*)* é uma camada de comunicação responsável por promover uma comunicação segura entre dois *sockets* (mecanismo que fornece um meio de comunicação entre dois processos) estando situada entre a camada de aplicação e a camada de transporte na pilha de protocolos TCP/IP. É também referenciada como um protocolo que objetiva prover privacidade e integridade utilizando mecanismos de criptografia para comunicações realizadas em redes IP como a Internet. A SSL admite uma variedade de algoritmos e opções distintas incluindo, por exemplo, a presença ou a ausência de compactação e a possibilidade de escolha de algoritmos criptográficos a serem utilizados (TANENBAUM, 2003).

A SSL foi desenvolvida em 1996 pela Netscape Communications Corp e, posteriormente submetida à IETF para padronização. O resultado foi a TLS (Transpor Layer Security), descrita na RF 2246. As mudanças feitas na SSL foram relativamente pequenas, mas suficientes para a SSL versão 3 (descrita de forma simplificada nesta seção) e a TLS não conseguirem interoperar. A versão TLS também é conhecida como SSL versão 3.1. (TANENBAUM, 2003) Atualmente a TLS está na versão 1.2 (chamada de SSL versão 3.3) e é descrita pela RFC 5246.

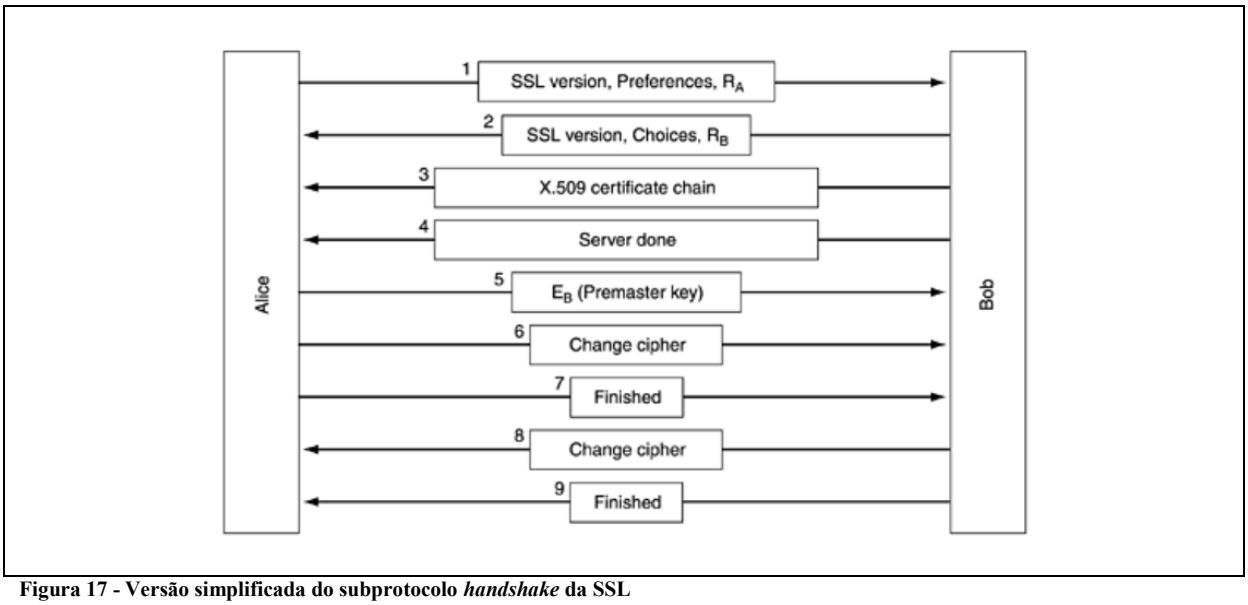

Figura 17 - Versão simplificada do subprotocolo *handshake* da SSL
Fonte: (TANENBAUM, 2003)

SSL é composto por dois subprotocolos, um para estabelecer uma conexão segura (*Handshake Protocol*) e outro para usá-la (*Record Protocol*).

A Figura 17 demonstra um cenário do funcionamento do subprotocolo que objetiva estabelecer uma conexão segura entre duas pessoas chamadas, respectivamente de Alice e Bob. Inicialmente Alice envia uma solicitação a Bob para estabelecer uma conexão; a solicitação especifica a versão de SSL que Alice tem e suas preferênias com relação aos algoritmos de compactação e de criptografia e contém também um número aleatório RA (também chamado nonce RA) que será utilizado posteriormente.

Na mensagem 2 Bob faz uma escolha entre os diversos algoritmos que Alice pode admitir e envia seu próprio *nonce* RB. Em seguida, na mensagem 3, ele envia um certificado contendo sua chave pública. Se esse certificado não for assinado por alguma autoridade conhecida, ele também envia uma cadeia de certificados que pode ser seguida de volta até chegar a uma autoridade original. Todos os navegadores, inclusive o de Alice, são pré-carregados com uma lista de chaves pública. Dessa forma, se Bob puder estabelecer uma cadeia ancorada em uma dessas chaves, Alice será capaz de verificar a chave pública de Bob.

Posteriormente, Bob pode enviar algumas outras mensagens (como uma solicitação do certificado de chave pública de Alice). Ao terminar, Bob envia a mensagem 4 para dizer a Alice que é a vez dela.

Alice responde escolhendo ao acaso uma chave pré-mestre de 384 bits e a envia para Bob, codificada com a chave pública de Bob (mensagem 5). A chave de sessão real usada para codificar os dados é derivada da chave pré-mestre combinada com ambos os *nonces* de modo complexo. Depois que a mensagem 5 é recebida, Alice e Bob são capazes de calcular uma chave de sessão. Por essa razão, Alice informa a Bob que ele deve passar para a nova cifra (mensagem 6) e também que ela concluiu o subprotocolo de estabelecimento de conexão (mensagem 7). Posteriormente, através das mensagens 8 e 9, Bob confirma as mensagens de Alice.

Na maioria dos casos práticos Alice sabe quem é Bob, mas Bob não sabe quem é Alice pois a maioria dos usuários ainda não possuem certificados digitais. Dessa forma, a primeira mensagem de Bob pode ser uma solicitação para Alice se conectar usando um *login* e uma senha estabelecidos anteriormente (TANENBAUM, 2003). Muitas aplicações Web que necessitam de segurança utilizam esse mecanismo, como por exemplo, as aplicações de e-commerce. Deve-se destacar que o protocolo de comunicação padrão para navegação na *World Wide Web* (WWW) é o protocolo HTTP. Quando utilizando em conjunto com o protocolo SSL o browser normalmente referencia essa junção como protocolo HTTPS.

Para aumentar a segurança, deve-se solicitar no passo 1 - estabelecimento da comunicação (Figura 18) – a utilização de algoritmos criptográficos mais fortes com chaves maiores, mesmo que isso incorra em uma demora um pouco maior na comunicação. Alguns browsers permitem habilitar ou não o uso de determinados algoritmos criptográficos em conexões seguras. A Figura 18 mostra a tela de configuração do navegador Opera que permite configurar os algoritmos criptográficos que serão utilizados em uma conexão SSL.

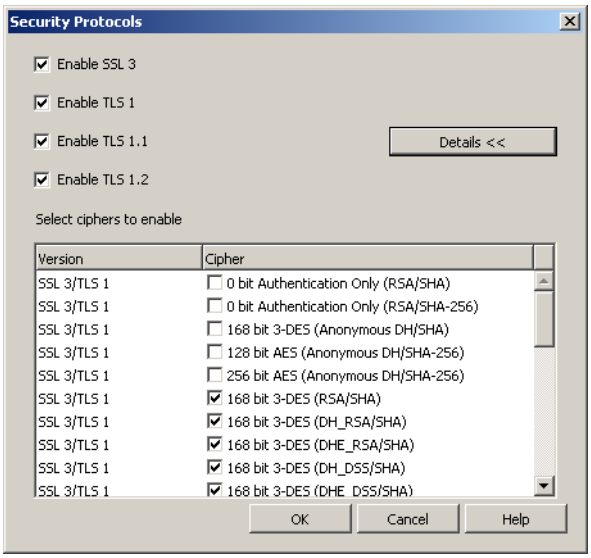

Figura 18 - Configuração de algoritmos criptográficos no browser Opera
Fonte: Elaborado pelo autor (2011)

2.9 - MECANISMOS DE SEGURANÇA X PROPRIEDADES DE SEGURANÇA

Foram vistos diversos mecanismos de segurança que incluem criptografia simétrica e assimétrica, envelope digital, sumário de mensagens, certificados digitais e protocolos de comunicação. Esta seção apresenta um resumo, através da Tabela 6, da relação entre cada um desses mecanismos e sua aplicabilidade para se alcançar as propriedades relacionadas à segurança em uma comunicação.

Tabela 6 - Relação entre técnicas e propriedades de segurança da informação
Fonte: elaborado pelo autor (2011)

	Autenticidade	Sigilo	Integridade	Não repúdio
Assinatura digital	x		x	x
Criptografia simétrica		x	*	
Criptografia assimétrica	x	x	*	
Envelope digital	x	x		
Sumário de mensagem			x	
TLS – *Transport Layer Sercurity*	*	x	x	

* A criptografia prover integridade é pouco comumente encontrado na literatura. Vários autores não citam essa possibilidade pelo fato de esse não ter sido o intuito original da criptografia, contudo é possível se obter integridade a partir da criptografia utilizando-se protocolos específicos. Por exemplo:

- **Protocolo 1:** o texto é concatenado com si mesmo e todo o texto é criptografado. O destinatário pode decriptografar todo o texto, dividir ao meio e comparar as duas partes.

- **Protocolo 2:** duas cópias do texto plano são criptografadas com duas chaves diferentes e ambos os textos cifrados são decriptografados pelo destinatário.

Ambas as hipóteses gerariam uma redundância maior que a utilização de um sumário de mensagens, contudo estariam menos sujeitos a ataques de colisão.

* O protocolo TLS – *Transport Layer Security* somente provê autenticidade em ambos os lados da comunicação (usuário e servidor) quando ambos os lados possuem certificados digitais. Foi visto anteriormente que na prática é comum que apenas o servidor possua certificados digitais, contudo paulatinamente os usuários estão começando cada vez mais a utilizar seus próprios certificados digitais.

Embora certificados digitais possam ser utilizados tanto pelo cliente quanto pelo servidor, o protocolo TLS não pode ser considerado um protocolo que provê Não Repúdio. Isto se deve ao fato de que os certificados digitais e as respectivas chaves públicas das partes envolvidas não são utilizados para assinar cada mensagem enviada trocada na comunicação. Por questões de desempenho, as chaves públicas são utilizadas com o intuito de se realizar uma troca de chave de sessão que será utilizada para trocar mensagens criptografadas com algoritmos criptográficos simétricos (similar ao conceito de envelope digital).

CAPÍTULO 3
- NORMAS E PADRÕES

A troca de informações entre computadores, segura ou não, é possível graças aos protocolos de comunicação que são especificações que normatizam o modo como a comunicação é realizada. Padrões possibilitam que diferentes sistemas possam trocar informações entre-si, mesmo quando têm que lidar com diversas tecnologias que podem envolver diferentes arquiteturas de hardware, diferentes sistemas operacionais, múltiplos algoritmos criptográficos, assinaturas e outros. Dessa forma, a interoperabilidade exige estrita adesão a um formato acordado padrão para dados transferidos (JR., 1993).

A necessidade para definição de padrões é algo inegável, mas deve-se haver uma preocupação para que a padronização possa ser abrangente o suficiente para que possa englobar todos os casos ao qual ela se propõe e deve ser limitada o suficiente para não enrijecer demais a sua utilização e impossibilitar um uso eficiente. No artigo que trata sobre os padrões PKCS (JR., 1993), o autor responde à pergunta "O que necessita ser padronizado?" sob a óptica da criptografia de chaves públicas e também comenta sobre a necessidade de padrões para: assinatura digital, envelope digital, certificação digital e acordo de chaves. A maioria dos padrões citados nesta seção terá como foco aqueles que tratam desses quatro itens.

Nas seções que seguem diversos padrões serão citados. A próxima seção é dedicada a citar padrões que especificam características físicas e interfaces de utilização de dispositivos biométricos e *smart cards* que são cada vez mais utilizados como dispositivos de segurança. Em seguida haverá uma seção dedicada a padrões relacionados ao uso de criptografia e certificados digitais seguida por uma seção específica que comenta padrões de avaliação de segurança para produtos e serviços.

3.1 - PADRÕES DE SMART CARDS E LEITORES BIOMÉTRICOS

PADRÕES DE SMART CARDS

Os *smart cards* têm sido muito utilizados em diversos cenários. Empresas de ônibus e restaurantes, por exemplo, utilizam o *smart card* para armazenar um determinado crédito que é debitado no momento que o mesmo é utilizado. O Governo Federal do Brasil tem utilizado os *smart cards* como dispositivos de identificação digital que provê mecanismos de criptografia e armazenamento de certificados digitais. Atualmente há diversos padrões de especificação de *smart cards* que detalham as propriedades físicas, características de comunicação, e identificadores de aplicação dos chips e dados. Quase todos os padrões que tratam de *smart card* de contato se referem à norma ISO 7816, partes 1, 2 e 3 como referência básica. A norma ISO 7816 será abordada com detalhes no Capítulo 6.

A interoperabilidade entre os sistemas de cartões é realizada em diferentes níveis: do cartão, dos terminais de acesso (leitores), das redes, dos sistemas proprietários dos provedores de cartão. Entre outros padrões que contém especificações pertinentes à tecnologia *smart cards* pode-se citar as normas ISO 7810, 7811, e 7813 que padronizam o tamanho dos cartões, a qualidade do plástico, o posicionamento dos contatos, as frequências utilizadas em cartões sem contato entre outros.

A especificação EMV (*Europay, MasterCard and VISA*), baseada nas normas ISO/IEC 7816 (*smart cards* de contato) e ISO / IEC 14443 (*smart cards* sem contato), define características físicas e em nível de aplicação para processamento de transações financeiras. Vale ainda destacar o Java Card que é um conjunto de especificações para rodar um subconjunto da linguagem Java em *smart cards* e JCRE (*Java Card Runtime Environment*) que é utilizado para gerenciar as operações de comunicação com *smart cards* em *applets java*.

Global Platform (GP) é uma associação não comercial que objetiva criar e promover especificações da tecnologia de *smart card*, incluindo especificações para os cartões, componentes de hardware para *smart card* e sistemas (SILVA, 2007). Entre as especificações destaca-se o *Global Platform Specifications Card* que define os componentes do cartão, conjunto de comandos, sequência de operação e interfaces; a especificação engloba hardware, sistema operacional, fornecedores etc. Pode ser aplicável para qualquer tipo de implantação da indústria.

A *Global Platform* possui uma gama de especificações extremamente abrangente que inclui assuntos como troca de mensagens, comunicações via TCP/IP, utilização de *Webservices* (padrões W3C e OASIS) para suportar *Global Platform System Messaging Specification* e outros.

PADRÕES DE DISPOSITIVOS BIOMÉTRICOS

A biometria ainda é muito pouco utilizada se comparada a outros mecanismos de autenticação, como por exemplo, o *token*. Isso se deve ao fato de seus dispositivos serem muito caros e ao fato de ser uma tecnologia além de mais complexa mais recente. Atualmente, os mecanismos de identificação biométricos utilizados se baseiam no reconhecimento da impressão digital, reconhecimento facial, reconhecimento da voz, reconhecimento da íris, reconhecimento de DNA e outros.

Embora o estudo da biometria seja relativamente recente, existe um consórcio denominado *BioAPI Consortium* que atualmente conta com mais de 120 organizações e que provê uma interface de programação de alto nível denominado BIOAPI - *Biometric Application Program Interface*. Tal interface provê um modelo genérico de autenticação biométrica de alto nível com objetivo de prover interoperabilidade entre diversas aplicações que utilizam tecnologia de biometria (SILVA, 2007).

A BioAPI encontra-se atualmente na versão 2.0 e derivou diversos padrões ANSI/IEC como exemplo os padrões ISO/IEC 19784, ISO/IEC 24708, ISO/IEC 24709 além do padrão ANSI INCITS 358. É possível encontrar diversos dispositivos de reconhecimento biométrico são compatíveis com a BioAPI. ANSI X9.84-2003 Biometric Information Management and Security for the Financial Services Industry.

3.2 - PADRÕES RELACIONADOS A CRIPTOGRAFIA E CERTIFICAÇÃO DIGITAL

X.509

Em 1988 a ITU criou e aprovou um padrão para certificados digitais denominado X.509 (TANENBAUM, 2003) que é o padrão mais adotado atualmente (SILVA, 2007). É baseado na utilização de criptografia de chave pública e assinaturas digitais e utiliza o mecanismo de assinatura digital para exigir a utilização de uma função *hash*. A norma não obriga a utilização de um algoritmo específico para criptografia ou para assinatura, mas recomenda o RSA.

Os certificados são codificados com o uso da ASN.1 (*Abstract Syntax Notation* 1) da OSI, que pode ser considerada uma *struct* em C, exceto por sua notação muito peculiar e extensa (TANENBAUM, 2003). O X.509 foi aprovado pela IETF e atualmente o X.509 encontra-se na versão 3.0. A Figura 19 (SILVA, 2007) mostra a evolução das versões do padrão X.509.

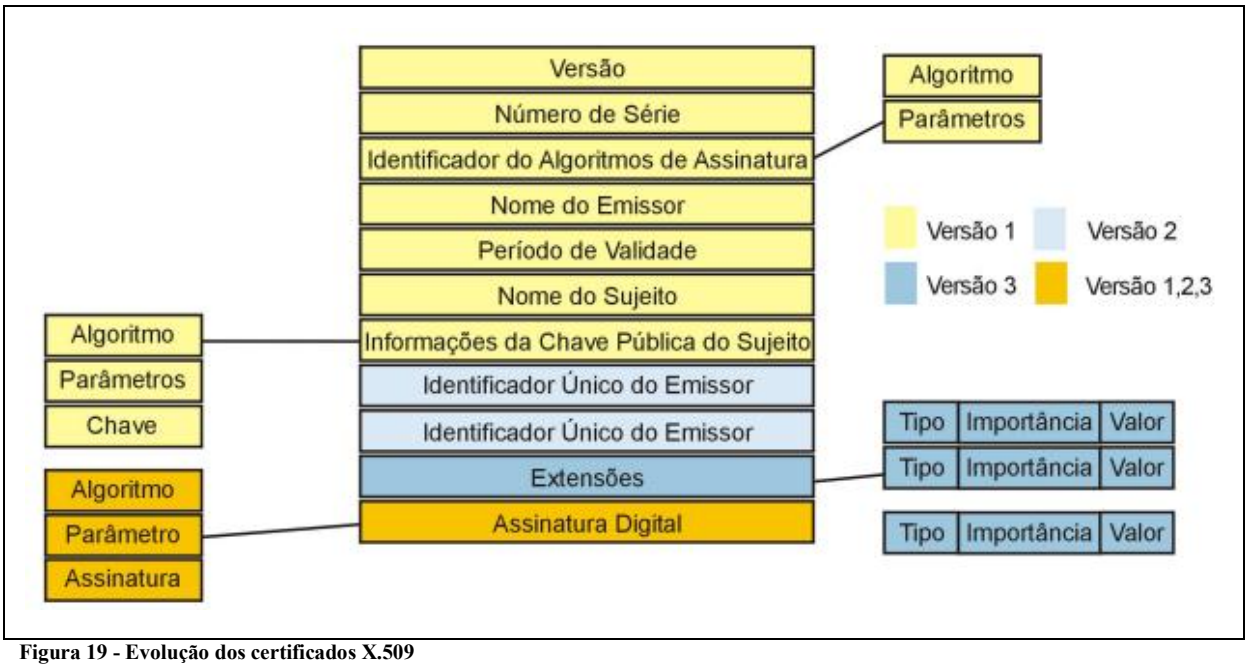

Figura 19 - Evolução dos certificados X.509
Fonte: (SILVA, 2007)

É relevante destacar que o padrão X.509 especifica como o conteúdo do certificado deve ser escrito, mas não especifica como o conteúdo dessa escrita deve ser codificada e armazenada em arquivos. Dois padrões de codificação comumente utilizados para codificar e armazenar os arquivos são os padrões DER (*Distinguished Encoding Rules*) e PEM (*Privacy Enhanced Mail*). A diferença básica entre os dois é que o primeiro utiliza uma codificação binária e o último utiliza uma codificação em modo texto com caracteres imprimíveis.

PUBLIC KEY CRYPTOGRAPHY STANDARDS (PKCS)

A *Public-Key Cryptography Standards* (PKCS) são especificações produzidas pelos Laboratórios RSA em cooperação com desenvolvedores de sistemas seguros de todo o mundo com o propósito de acelerar a implantação de criptografia de chave pública (SILVA, 2007). Publicado pela primeira vez em 1991 como resultado das reuniões com um pequeno grupo de pioneiros da tecnologia de chave pública, os documentos PKCS tornaram-se amplamente referenciados e implementadas. Contribuições a partir da série PKCS tornaram-se parte de muitos formais e normas de fato, incluindo os documentos ANSI X9, PKIX, SET, S / MIME, e SSL (RSA LABORATORIES). Pertencem ao conjunto PKCS (RSA LABORATORIES):

- **PKCS#1:** ***RSA Cryptography Standard***. Fornece recomendações para a implementação de criptografia de chave pública baseado no algoritmo RSA, abrangendo os seguintes aspectos: primitivas criptográficas; esquemas de criptografia, sistemas de assinatura digital com apêndice, sintaxe ASN.1 para representar as chaves e para identificar os esquemas.

- **PKCS#3:** ***Diffie-Hellman Key Agreement Standard***. Descreve o método para implementação de chaves no Diffie-Hellman. Esse padrão destina-se a ser utilizado em protocolos que estabelecem conexões seguras.

- **PKCS#5:** ***Password-Based Encryption Standard***. Provê recomendações para o desenvolvimento de criptografia baseada em senha cobrindo aspectos como funções de derivação de chaves, sintaxe ASN1.1 para identificar a técnica e outros. Destina-se a ser utilizado em aplicações de propósito geral de computadores e sistemas de comunicação que queiram proteger informações sensíveis como chaves privadas, como em PKCS#8.

- **PKCS#6:** ***Extended-Certificate Syntax Standard***. Descreve a sintaxe para os certificados estendidos, que consiste em um certificado e um conjunto de atributos, coletivamente, assinado pelo emissor do certificado. Destina-se a estender o processo de certificação para além de emitir uma chave pública certificar outras informações sobre a determinada entidade.

- **PKCS#7:** ***Cryptographic Message Syntax Standard***. Descreve uma sintaxe geral para os dados serem criptografados e aplicados em assinaturas digitais e envelopes digitais. Permite que atributos arbitrários, tais como a assinatura de tempo, para ser autenticado, juntamente com o conteúdo de uma mensagem, e provê outros atributos, como respectivas assinaturas para ser associado com uma assinatura. Um caso degenerado da sintaxe fornece um meio para divulgação de certificados e listas de revogação de certificado. É utilizado para prover mensagens seguras em S/MIME e é compatível com PEM. Versão atual 1.5.

- **PKCS#8:** ***Private-Key Information Syntax Standard***. Descreve a sintaxe utilizada para armazenamento de chaves privadas, incluindo uma chave privada de um algoritmo de chave pública e um conjunto de atributos. Recomenda a utilização de algoritmos criptográficos descritos em PKCS#5.

- **PKCS#9:** ***Selected Attribute Types***. Define os tipos de atributos selecionados para uso em PKCS#6 certificados prorrogado, PKCS#7 assinado digitalmente mensagens, PKCS # 8 informações de chave privada, e PKCS#10 pedidos de assinatura de certificado. Versão atual 2.0.

- **PKCS#10:** ***Certification Request Syntax Standard***. Descreve uma sintaxe para codificar requisições de certificados, incluindo o nome da pessoa que requisita o certificado e sua chave pública e possivelmente um conjunto de atributos que são assinados pela entidade requisitante.

- **PKCS#11:** ***Cryptographic Token Interface Standard***. Especifica uma API chamada Cryptoky (*Cryptographic Token Interface*), para dispositivos que guardam informações de criptografia e executam funções criptográficas. É uma interface para token criptográfico que segue uma abordagem simples baseada em objetos para obter independência tecnológica (qualquer tipo de dispositivo) e compartilhamento de recursos (acesso a múltiplas aplicações de diversos dispositivos).

- **PKCS#12:** ***Personal Information Exchange Syntax Standard***. Especifica um formato portátil para armazenamento ou transporte de chaves privadas de um usuário, certificados e outros.

- **PKCS#13:** ***Elliptic Curve Cryptography Standard***. Padrão ainda em desenvolvimento que objetiva tratar aspectos relativos à utilização de criptografia que utiliza o algoritmo baseado em curvas elípticas, como por exemplo, parâmetros e geração de chaves, validação, sintaxe ASN.1, entre outros.

- **PKCS#15:** ***Cryptographic Token Information Format Standard***. Estabelece um padrão que permite aos usuários utilizar *tokens* criptográficos para identificar-se a múltiplas aplicações de reconhecimento de padrões, independentemente da Cryptoki (ou de outra interface de *token*).

OUTROS PADRÕES

- **ANSI X9.31-1998**: contém especificações para o algoritmo RSA. O padrão cobre especificamente o gerenciamento manual e automático de chaves usando na criptografia assimétrica e simétrica para a indústria nos serviços financeiros de venda no atacado; (SILVA, 2007)

- **ANSI X9.62-1998**: contém especificações para o algoritmo de assinatura ECDSA. (SILVA, 2007)

- **O FIPS 186-2**: especifica o conjunto de algoritmos utilizados para geração e verificação de assinaturas digitais. Esta especificação relaciona três algoritmos especificamente: *Digital Signature Algorithm* (DAS), RSA, e o algoritmo para assinatura digital de curvas elípticas (ECDSA); (SILVA, 2007)

- **FIPS 197**: O *Advanced Encryption Standard* (AES) é um algoritmo de criptografia simétrica aprovado pela FIPS. (SILVA, 2007)

3.3 - PADRÕES PARA AVALIAÇÃO DE SEGURANÇA

FIPS 140

O FIPS 140 é um padrão desenvolvido pelo governo americano em conjunto com o governo do Canadá que certifica módulos criptográficos, especificando por exemplo, quais algoritmos de criptografia e de *hash* podem ser usados e como as chaves de criptografia devem ser geradas e gerenciadas em módulos criptográficos.

A segunda versão FIPS 140-2 é a última versão final do FIPS-140 e possui 4 níveis de garantia de segurança que variam de 1 a 4, sendo o nível mais baixo o menos rigoroso e o nível 4 o mais rígido. Cada nível é uma especialização do nível anterior, dessa forma o nível 2 contém o nível 1, o nível 3 contém o nível 2 e assim sucessivamente. Desde 2009 está em processo de desenvolvimento um rascunho para terceira versão que tende a incluir o nível 5 de segurança.

Os níveis de segurança do FIPS 140-2 especificam:

- **Nível 1**: oferece o menor nível de confiabilidade e impõe requisitos básicos de segurança para módulos criptográficos. Permite que os componentes de software de um módulo criptográfico sejam executados em um sistema computacional de propósito geral com um sistema operacional não avaliado. Essas implementações podem ser apropriadas para aplicações de segurança onde controles, como segurança física, segurança de rede, e os procedimentos administrativos são prestados fora do módulo.

- **Nível 2**: adiciona requisitos de segurança físicos ao módulo criptográfico para evitar a adulteração do *hardware* (*tamper-evidence*) através de revestimentos invioláveis ou selos, fechaduras resistentes em tampas removíveis e outros mecanismos que impossibilitem alcançar fisicamente os parâmetros de segurança (CSP's) dentro do módulo.

- **Nível 3**: adiciona requisitos para resistir à violação do hardware (tornando o acesso à informação contida no chip mais difícil), garantir a autenticação e garantir uma separação física e lógica entre as interfaces cujos parâmetros críticos de segurança entrem ou saiam do módulo. Destina-se ter uma alta probabilidade de detectar e

responder às tentativas que dão acesso físico direto, e uso ou modificação do módulo criptográfico. Zerar todos os CSP's (*Criptographic Service Providers*) é um dos mecanismos de proteção realizado quando é detectada a violação do hardware.

- **Nível 4**: faz com que os requisitos de segurança física sejam mais rigorosos, impondo maior resistência contra ataques externos. Fornecem um envelope completo de proteção em torno do módulo criptográfico com o intuito de detectar e responder a todas as tentativas não autorizadas de acesso físico. Qualquer violação física tem uma alta probabilidade de ser detectado, resultando na exclusão imediata de todos os provedores de texto simples. O nível 4 é tão rigoroso que até mesmo a variação de condições ambientais como temperatura e voltagem podem ser consideradas tentativas de violação.

Os testes para certificação FIPS 140 são feitos por laboratórios independentes que foram previamente certificados para tal propósito. Os resultados são encaminhados para o NIST2 (*National Institute of Standards and Technology*) nos Estados Unidos e para o *Canadian Security Establishment* (CSE) no Canadá para uma validação independente (SILVA, 2007).

Assim como na certificação Common Criteria, o processo seguido para o FIPS 140 vem de padrões rigorosos tanto para os testes do módulo quanto para o laboratório que conduz os testes, combinados com a validação independente dos resultados dos testes.

ISO/IEC 15408 (COMMOM CRITERIA)

O Common Criteria é um padrão internacional voltado para a área se segurança da informação na computação que é designado para ser usado como base para a avalição das propriedades de segurança dos produtos de TI (ISO/IEC, 2009). O padrão disponibiliza documentos que visam integrar os interesses dos consumidores, dos desenvolvedores e fornecedores de produtos e serviços e avaliadores e certificadores da norma.

O Common Criteria não provê uma lista de requisitos de segurança do produto, ou recursos que os produtos devem conter. Ao invés disso, provê uma camada base na qual fabricantes podem especificar os seus próprios requisitos de segurança ou implementar ou aclamar sobre os atributos de segurança de seus próprios produtos (SILVA, 2007). O processo de certificação/validação dos resultados poderá atestar se as propriedades asseguradas pelo fabricante são ou não de fato garantidas.

São conceitos importantes do Common Criteria:

- ***Target of Evaluation (TOE)***: produto ou sistema sujeito a avalição.
- ***Protection Profile (PP)***: uma declaração de dependente de implementação de segurança necessidades específicas identificadas por um TOE.
- ***Security Target (ST)***: documento que identifica requisitos de segurança relevantes para um propósito particular. Uma declaração dependente da implementação de necessidades de segurança específica identificada para uma TOE.
- ***Security Functional Requirements (SFRs)***: uma tradução dos objetivos de segurança da TOE em um nível de abstração mais detalhado. Provê uma descrição exata do que será avaliado. Os objetivos de segurança (ST) são normalmente especificados em linguagem natural, a tradução em uma linguagem padronizada propicia uma descrição mais exata das funcionalidades da TOE.
- ***Security Assurance Requirements (SARs)***: descreve como o TOE deve ser avaliado utilizando uma linguagem padronizada que auxilia na criação de uma descrição precisa e sem ambiguidades.
- ***Evaluation Assurance Level (EAL)***: são níveis de garantia de segurança que especificam o rigor a qual o TOE foi submetido no processo de avaliação. Cada EAL correspondente a um pacote requisitos de garantia de segurança (SAR) sendo divididos em sete níveis:
    - EAL1 - Funcionalmente testado (*Functionally Tested*)

- EAL2 - Estruturalmente testado (*Structurally Tested*)
- EAL3 - Metodicamente testados e verificados (*Methodically Tested and Checked*)
- EAL4 - Metodicamente projetados, testados e revisados (*Methodically Designed, Tested, and Reviewed*)
- EAL5 – Projetado semiformalmente e testados (*Semiformally Designed and Tested*)
- EAL6 - Projeto verificado semiformalmente e testado (*Semiformally Verified Design and Tested*)
- EAL7 - Projeto verificado formalmente e testado (*Formally Verified Design and Tested*).

O Common Criteria já é bastante difundido mundialmente sendo utilizado por diversos produtos comerciais; por exemplo, o MAC OS Server a partir da 10.3.6 adiquiriu algumas certificações Common Criteria e, o Windows 2003 Server atingiu a certificação EAL 4. Atualmente o Common Criteria está na versão 3.1 sendo dividido em três livros (ISO/IEC, 2009):

- **ISO 15408-1:** ***Introduction and General Model***. Define conceitos e princípios gerais de avaliação da segurança de TI e apresenta um modelo geral de avaliação. Esta parte também apresenta estruturas para expressar os objetivos de TI, segurança de seleção e definição de requisitos de TI, segurança e para a escrita de alto nível de especificações para produtos e sistemas. Além disso, fornece a utilidade de cada parte do CC, em termos de cada um dos públicos-alvo.

- **ISO 15408-2:** ***Security Functional Requirements***. Estabelece um conjunto de componentes funcionais de segurança como uma forma padronizada de expressar os requisitos de segurança para produtos e sistemas de TI. O catálogo é organizado em classes, famílias e componentes.

- **ISO 15408-2 - *Security Assurance Requirements***. Estabelece um catálogo com um conjunto de componentes de segurança que pode ser usado como uma forma padronizada de expressar os requisitos de garantia para produtos de TI e sistemas. Part 3 também definie critérios para Perfis de Proteção (PP) e Objetivos de Segurança (STs). Apresenta 7 níveis de Garantias de Níveis de Avaliação (EALs - *Evaluation Assurance Levels*), que são pacotes predefinidos de componentes de garantia que compõem a escala de CC para a confiança na avaliação da segurança de produtos e sistemas.

CAPÍTULO 4 - TECNOLOGIAS DE ACESSO AO MEIO SEM FIO

Existem diferentes tipos de redes sem fio que variam em tecnologia e aplicação, sendo possível classificá-las de acordo a dispersão geográfica em WPAN's (*Wireless Personal Area Network*), WLAN's (*Wireless Local Area Network*), WMAN's (*Wireless Metropolitan Area Network*) e WWAN's (*Wireless Wide Area Network*).

WPAN's são redes de comunicação sem fio que envolvem dispositivos portáteis e móveis, como PC's, PDA's, periféricos, celulares e outros eletrônicos, permitindo com que se comunicarem e interajam uns com os outros. As características dos dispositivos que compõem as redes PAN's são alcance pequeno, pouco consumo de energia, baixo custo, interagem dentro de um espaço de uso pessoal.

WLAN's possuem alcance médio e envolvem dispositivos com restrições energéticas não críticas e podem estar dispostos em uma área superior a apenas um espaço de uso pessoal. As WLAN's são muito utilizadas em domicílios e empresas normalmente conectando computadores e notebooks e possibilitando o acesso à internet e a dispositivos fixos como impressoras.

WMAN's têm a abrangência de uma cidade e podem interligar diversas LAN's ou WLAN's. A abrangência de WWAN é ilimitada e pode englobar múltiplas cidades, países ou até mesmo continentes.

O foco deste trabalho está direcionado para o desenvolvimento de um dispositivo cuja aplicabilidade se enquadra como uma rede pessoal sem fio WPAN. Dessa forma serão tratadas apenas as tecnologias de comunicação sem fio que sejam relevantes nesse contexto. As tecnologias que possibilitam comunicação sem fio compatíveis com dispositivos móveis mais difundidas são Wi-Fi (IEEE 802.11[11]), Bluetooth (IEEE 802.15.1), IrDA e NFC.

[11] O padrão 802.11 é destinado a redes WLAN, contudo está presente na maioria dos dispositivos móveis inteligentes e, por esta razão, será descrito.

Contudo existem outras tecnologias que podem ser citadas como, por exemplo, HomeRF, RFID, ZigBee (IEEE 802.15.4) e UWB-Ultra-Wideband (IEEE 802.15.3).

4.1 - IRDA STANDARD

Em 1993 diversos líderes dos setores de telecomunicações e computação - entre eles Motorola, ACTiSYS, Microsoft, Sony, Nokia, Apple, AT&T, Compaq, Intel, HP e outros - fundaram a Infrared Data Association (IrDA). A associação especifica a comunicação física e padrões de protocolo para a troca de dados em redes de curto alcance através de luz infravermelha através do padrão também denominado IrDA.

A especificação IrDA inclui padrões que envolvem diversas camadas. Entre os padrões pode-se citar IrPHY (*Infrared Physical Layer Specification*), IrLAP (*Infrared Link Access Protocol*), IrLMP (*Infrared Link Management Protocol*), IrCOMM (*Infrared Communications Protocol*), Tiny TP (*Tiny Transport Protocol*), OBEX (*Object Exchange*), IrLAN (*Infrared Local Area Network*) and IrSimple e IrFM (Infrared financial messaging).

As conexão físicas de dispositivos IrDA são muito simples. Dispositivos entram apontam uns para os outros com uma amplitude máxima de 30 graus, com no máximo um metro de cone, em suma, eles são apontados um para o outro em escala relativamente estreita. Um dispositivo (o principal) inicia a descoberta do dispositivo, e se um dispositivo remoto (o secundário) é detectado, ele pode iniciar uma conexão. Os dois dispositivos desfrutam de uma conexão simples, ponto-a-ponto na taxa mais rápida de dados suportados por ambos os dispositivos. Geralmente, a conexão é fechada quando a sessão está concluída. A comunicação IrDA ocorre em *half-duplex* com taxa máxima de transmissão de 4Mb/s (DIVINEY, 2003). Extensões à versão inicial permitem taxas de transferência de até 4 ou 16 Mbps. O tempo de conexão entre dois dispositivos IrDA é aproximadamente igual a 0,5 segundos. (ORACLE, 2011)

SEGURANÇA

O IrDA não implementa nenhum mecanismo com fim de prover segurança. Contudo a limitação física imposta pela distância máximo entre 2 dispositivos e o ângulo máximo entre eles pode ser entendido como um mecanismo de segurança simples (MULLER, 2000) e que a depender do contexto pode até ser eficiente.

4.2 - *BLUETOOTH* (IEEE 802.15)

Bluetooth é uma especificação de redes sem fio tradicionalmente utilizada em redes locais (LAN - *Local Area Network*) e redes pessoais (PAN - *Personal Area Network*). É regulamentada por um consórcio denominado *Bluetooth Special Interest Group* (SIG) que foi fundado em 1998 e engloba grandes empresas como IBM, Ericson, 3Com, Microsoft e outras.

Atualmente, diversos dispositivos utilizam *Bluetooth* com múltiplas finalidades. Como exemplo, podem-se citar celulares que enviam fotos, vídeos ou disponibilizam jogos *multiplayer*; *vídeo games* que se comunicam com *joysticks*; *players* de música transmitem áudio a fones de ouvido sem fio; celulares que se integram a automóveis possibilitando a utilização do aparelho de som do automóvel ou o painel eletrônico para buscar dados na agenda e outros.

O padrão *Bluetooth* se encontra na versão 3 (abril de 2009). O SIG tem mostrado preocupação com retro compatibilidade, isto é, um dispositivo da versão 3 será capaz de se comunicar com um dispositivo versão 2 e assim sucessivamente. A cada nova versão tem se mostrado um grande ganho de velocidade; a versão 1.2 tinha uma taxa de transferência máxima possível de 1Mbit/s, a versão 2.0+EDR possui taxa de transferência de 3Mbit/s e a versão 3 suporta taxas de até 24Mbit/s.

Dispositivos *Bluetooth* podem ser classificados em três classes de acordo com seus transmissores: Classe 1, que opera com potência máxima de 100 mW e pode obter um alcance de até 100 metros; Classe 2 opera com potência máxima de transmissão de 2,5 mW e possui alcances de 10 metros; e Classe 3, opera com potência máxima de transmissão de 1 mW permitindo alcance máximo de 1 metro (BLUETOOTH RANGE, 2011). O tempo de conexão

entre dois dispositivos Bluetooth, na primeira vez que se comunicam, é de aproximadamente 6 segundos. (ORACLE, 2011)

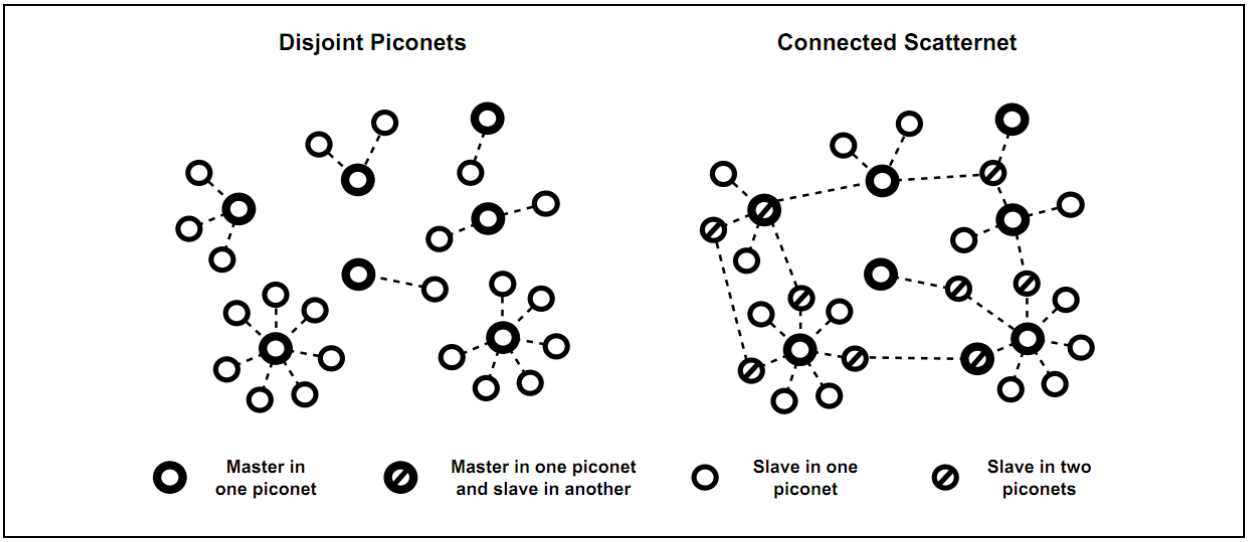

Figura 20 - Scatternet formada por duas piconets
Fonte: (WHITAKER, HODGE e CHLAMTAC, 2004)

Dispositivos *Bluetooth* podem se comunicar de forma *Ad Hoc* ou podem ser agrupados em redes conhecidas como *piconet's*. Cada *piconet* consiste exatamente em um dispositivo cuja função é de mestre, e um máximo de outros sete dispositivos ativos, cujos papéis são escravos. Os papéis de mestre e escravo são relativos a uma *piconet*, em um determinado instante no tempo (WHITAKER, HODGE e CHLAMTAC, 2004).

Embora o máximo de escravos ativos seja limitado a sete, é possível haver outros 255 escravos estacionados (estado *parked*) na rede. Dispositivos que não pertencem a nenhuma *piconet* permanecem no modo espera (estado *standby*). Toda comunicação é feita entre o mestre e um escravo; não é possível a comunicação direta entre escravos. A comunicação entre mestre e escravo ocorre utilizando uma multiplexação por divisão de tempo, onde o mestre transmite em slots de tempo pares e os escravos transmitem em slots de tempo ímpares numerados (WHITAKER, HODGE e CHLAMTAC, 2004).

A interconexão de mais de uma *piconet,* possibilitando a comunicação entre quaisquer dois membros é conhecida como *scatternet*. Um mesmo dispositivo pode participar de mais de uma piconet assumindo diferentes papéis. Este cenário pode ser visualizado na Figura 20 onde há dispositivos atuando nas formas de mestre em apenas uma piconet, escravo em

apenas uma piconet, mestre em uma piconet e escravo em outra piconet ou escravo em duas piconet's.

Os dados em uma rede *piconet* são transmitidos em quadros que contém pelo menos código de acesso, um cabeçalho e um *payload*. O código de acesso identifica o mestre e permite que escravos situados dentro do alcance de rádio de dois mestres possam conhecer o destino de cada tráfego. Dispositivos *Bluetooth* quando estão operando de modo a descobrirem uns aos outros irão transmitir informações como nome do dispositivo, classe do dispositivo, lista de serviços, informações técnicas (ex.: fabricante, versão do *Bluetooth* utilizada, relógio para sincronização, etc.) e outras.

ESTADOS DOS DIPOSITIVOS BLUETOOTH

Dispositivos *Bluetooth* podem assumir dois estados de funcionamento: espera (*standby*) e conexão (*connection*). O estado de espera é o estado padrão utilizado para economia de energia e, neste caso, somente o relógio nativo permanece executando não ocorrendo interação com outro dispositivo. No estado de conexão, o mestre e o escravo podem trocar pacotes usando o código de acesso do canal e o relógio de sincronização do mestre.

Estados de *standby*:

- ***Inquiry***: utilizado para descobrir a identidade dos dispositivos Bluetooth que estão próximos.
- ***Inquiry Scan***: utilizado quando o dispositivo está escutando para tornar-se disponível para dispositivos que estão no estado de *Inquiry* (pesquisando).
- ***Inquiry response***: um dispositivo escravo responde com suas informações que incluem DAC (*Device Access Code*), clock nativo e outras.
- ***Page***: estado utilizado pelo mestre para ativar e conectar-se a um escravo.

- ***Page Scan***: estado em que um dispositivo *slave* periodicamente entra para que possa escutar uma possível requisição para estabelecer conexões.

- ***Slave response***: estado em que um dispositivo escravo responde à mensagem Page enviada pelo mestre e espera a mensagem *Master Response* para que possa mudar para o estado de Conexão.

- **Master response**: estado em que um dispositivo mestre entra após receber a resposta do escravo (*slave response*). Envia algumas informações adicionais para o escravo e, posteriormente, entra em estado de Conexão.

Estados de Conexão:

- ***Active***: tanto mestre quanto escravos participam ativamente no canal de comunicação. Mestre e escravo são mantidos sincronizados entre-si.

- ***Sniff***: dispositivo escravo escuta apenas em slots de tempo pré-definidos de modo a economizar energia.

- ***Hold***: dispositivo deixa de suportar o tráfego de ACL (*Asynchronous Connection-Less*) por um período de tempo para que possa economizar energia e tornar o canal disponível para outras atividades *Page*, *Page Scan*, etc.

- ***Park***: dispositivo escravo não participa do canal piconet, mas continua sincronizado com o canal. Pode entrar no modo Park, que é um estado que consome baixa energia, com muito pouca atividade.

PERFIS

Um perfil Bluetooth é uma especificação de interface para comunicações entre dispositivos baseadas em Bluetooth. A forma como um dispositivo usa a tecnologia Bluetooth e sua interoperabilidade com outros dispositivos depende de sua compatibilidade com os diferentes tipos de perfil. Por exemplo, para que um automóvel possa identificar no seu painel o nome de quem está discando para o celular do motorista tanto o celular quanto o automóvel

devem prover uma compatibilidade com o perfil PBAP (*Phone Book Access Profile*), da mesma forma para que seja possível utilizar um *headset* sem fio em um celular ambos devem suportar o perfil HSP (*Headset Profile*).

Esta seção enumera os diferentes tipos de perfil existentes na especificação Bluetooth e agrupa em 5 categorias: Perfis Controle de mídia, Perfis relacionados a redes, Perfis de controle de telefonia, Dispositivos específicos e Outros perfis.

PERFIS DE CONTROLE DE MÍDIA

1. ***Generic Audio/Video Distribution Profile*** **(GAVDP):** Perfil de distribuição generalizado de áudio/vídeo. Fornece o suporte básico para o A2DP e VDP.

2. ***Advanced Audio Distribution Profile*** **(A2DP):** Define como áudio de alta qualidade (estéreo ou mono) pode ser transferido em tempo real de um dispositivo para o outro sobre uma conexão Bluetooth. Por exemplo, músicas podem ser transferidas de um telefone celular para um fone de ouvido sem fio.

3. ***Audio/Video Remote Control Profile*** **(AVRCP):** Designado para prover uma interface padrão para controlar TVs, equipamentos de alta fidelidade, entre outros, permitindo o uso de um único controle remoto (ou outro dispositivo) para controlar todos os equipamentos de áudio/vídeo aos quais o usuário tenha acesso. Pode ser usado em conjunto com o A2DP ou VDP.

4. ***Video Distribution Profile*** **(VDP):** Permite o transporte em tempo real de vídeo. Ele pode ser usado, por exemplo, para transmissão de conteúdo de um vídeo gravado em um computador pessoal ou um computador media center para um player portátil, ou de uma câmera de vídeo digital para uma TV. Obrigatoriamente o perfil suporta a especificação H.263 e há suporte opcional coberto pela especificação para MPEG-4 Visual Simple Profile e os perfis 3 e 8 do H.263.

5. ***Basic Imaging Profile*** **(BIP):** Designado para enviar imagens entre dispositivos e inclui a habilidade de redimensionar e converter imagens automaticamente para torná-las compatíveis com o dispositivo receptor.

6. ***Basic Printing Profile* (BPP):** Permite aos dispositivos enviar textos, e-mails, vCards ou outros itens para impressoras baseadas em tarefas de impressão. Ele difere do HCRP por não haver dependência de drivers específicos de impressoras. Isso o torna mais acessível e flexível para dispositivos como telefones celulares ou câmeras digitais que não podem ser facilmente atualizados com drivers dependentes de fabricantes de impressoras.

PERFIS RELACIONADOS A REDES

1. ***Common ISDN Access Profile* (CIP)**: Provê acesso irrestrito aos serviços que a tecnologia ISDN oferece.

2. ***File Transfer Profile* (FTP)**: Provê acesso ao sistema de arquivos do outro dispositivo. Isso inclui o suporte para transferir informações e listagem de arquivos, trocar para diferentes pastas, transferir, enviar e excluir arquivos. O perfil usa OBEX como um transporte e é baseado no GOEP.

3. ***LAN Access Profile* (LAP)**: O perfil de acesso LAN habilita um dispositivo Bluetooth a acessar uma rede LAN, WAN ou Internet por outro dispositivo que tem acesso físico à conexão da rede. Ele usa PPP sobre RFCOMM para estabelecer conexões. A especificação LAP também é usada para permitir que o dispositivo acesse redes *Bluetooth ad-hoc*. Esse perfil foi substituído pelo perfil PAN na especificação *Bluetooth*.

4. ***Personal Area Networking Profile* (PAN)**: Permite o uso do protocolo de encapsulamento de rede Bluetooth (*Bluetooth Network Escapsulation Protocol*) na camada 3 para transporte de informações sobre um link Bluetooth. Funciona de maneira semelhante ao LAN Access Profile, permitindo que dispositivos se unam e formem uma rede pessoal onde cada um pode usar os serviços disponíveis nela.

5. ***Wireless Application Protocol Bearer* (WAPB)**: Permite o uso do protocolo *Wireless Application Protocol* (WAP) sobre ponto a ponto sobre o Bluetooth.

6. ***Serial Port Profile*** **(SPP)**: Baseado na especificação ETSI TS 07.10 e usando o protocolo RFCOMM, emula um cabo serial para prover uma simples implementação sem fio para as conexões seriais RS-232 existentes e seus aplicativos, incluindo um controle familiar de sinais. Provê também o básico para os perfis DUN, FAX, HSP e AVRCP.

7. ***Dial-up Networking Profile*** **(DUN)**: Provê um padrão para o acesso à internet e outros serviços dial-up sobre Bluetooth. O cenário mais comum é o acesso à internet por um notebook discando para um provedor de acesso por telefone celular, sem fio. Ele é baseado no Serial Port Profile (SPP), e provê uma conversão relativamente fácil de produtos existentes, através das várias características que tem em comum com os protocolos seriais com fio para as mesmas tarefas.

PERFIS DE CONTROLE DE TELEFONIA

1. ***Fax Profile*** **(FAX)**: Desenvolvido para prover uma interface bem definida entre um telefone celular ou linha de telefone fixo e um computador com um software de fax instalado.

2. ***Cordless Telephony Profile*** **(CTP)**: Desenvolvido para telefones sem fio que utilizam Bluetooth. Um uso possível seria em telefones celulares que podem usar um gateway Bluetooth CTP conectado à uma linha de telefone fixo quando em casa. Assim, logo que o telefone saísse do alcance do rádio residencial, o telefone automaticamente alternaria para a rede celular.

3. ***SIM Access Profile*** **(SAP, SIM, rSAP)**: Permite que dispositivos, como telefones automotivos com receptores GSM embutidos, se conectem a um cartão SIM em um telefone ativado por Bluetooth.

4. ***Intercom Profile*** **(ICP)**: Geralmente referido como Perfil de walkie-talkie (walkie-talkie profile), é outra especificação baseada na especificação de protocolo de controle de telefone (Telephone Control protocol Specification, TCS), dependente também do SCO para tramissão de áudio. É proposto para permitir chamadas de voz entre dois aparelhos Bluetooth compatíveis.

5. ***Phone Book Access Profile*** **(PBAP, PBA)**: permite a troca de objetos da agenda telefônica entre dispositivos. É muito usado entre um kit automotivo e um telefone celular para permitir que o sistema do carro possa mostrar o nome de uma pessoa ligando.

CLASSES ESPECÍFICAS DE DISPOSITIVOS

1. ***Human Interface Device Profile*** **(HID):** Provê suporte para dispositivos como mouses, joysticks e teclado, assim como suporte para simples botões e/ou indicadores em outros tipos de dispositivos. O perfil foi designado para prover um link de baixa latência e potência, com baixo consumo de energia.

2. ***Hands-Free Profile*** **(HFP):** Comumente usado para permitir que kits automotivos de handsfree possam se comunicar com telefones celulares no alcance do carro

3. ***Headset Profile*** **(HSP):** É um dos perfis mais comumente utilizados da tecnologia, provendo suporte para os *headsets* e fones de ouvido Bluetooth para serem utilizados com telefones celulares. É dependente do SCO para codificação de áudio à 64kbits/s CVSD ou PCM e contém um subconjunto de comandos AT da GSM 07.07 para controle mínimo do sistema, incluindo a habilidade para discar, atender ou desligar uma chamada e ajustar o volume.

4. ***Hard Copy Cable Replacement Profile*** **(HCRP):** Provê uma simples alternativa sem-fio para a conexão via cabo entre um dispositivo e uma impressora. Porém, essa especificação não tem um padrão definido entre a comunicação atual entre a impressora e o dispositivo, por esse motivo drivers são requeridos para o modelo específico de impressora. Isso torna o perfil menos flexível para dispositivos como câmeras digitais e palmtops, visto que atualizações de software podem ser potencialmente problemáticas.

OUTROS PERFIS

1. ***Generic Access Profile*** **(GAP):** provê a base para todos os outros perfis da tecnologia.

2. ***Device ID Profile*** **(DIP):** Permite um dispositivo ser identificado além das limitações de classe de dispositivos já disponíveis no Bluetooth. Ele ativa a identificação do fabricante, identificação de produto (Device ID) e sua respectiva versão, além da versão de produto. Isso é útil, por exemplo, para um computador reconhecer a conexão de um dispositivo novo e fazer o download dos drivers necessários. O perfil habilita aplicações similares aquelas usadas pela especificação Plug-and-play nos computadores pessoais.

3. ***Generic Object Exchange Profile*** **(GOEP):** Provê a base para todos os perfis de troca de dados. Baseado no OBEX (*OBject EXchange Profile*).

4. ***Health Device Profile*** **(HDP):** Perfil projetado para facilitar a transmissão e recepção de dados de Dispositivos Médicos. A API's dessa camada interage com o nível mais baixo Multi-Channel Adaptation Protocol (MCAP camada), mas também realizar SDP comportamento para se conectar a dispositivos remotos HDP. Também faz uso do dispositivo de identificação Perfil (DIP).

5. ***Object Push Profile*** **(OPP):** Um perfil básico para envio de "objetos", tais como imagens, cartões de visita ou detalhes de compromissos. O OPP utiliza a API do perfil OBEX com os comandos *connect* (conectar), *disconnect* (desconectar), *put* (colocar), *get* (pegar, transferir) e *abort* (abortar, cancelar). Por usar essas API's a camada do OPP reside sobre o OBEX e por este motivo segue as mesmas especificações.

6. ***Service Discovery Application Profile*** **(SDAP)**: descreve como um aplicativo deve usar o SDP para descobrir serviços em um dispositivo remoto. Assim, um aplicativo pode ser habilitado para descobrir quais serviços Bluetooth estão disponíveis em um dispositivo ao qual ele se conecta.

7. ***Synchronization Profile*** **(SYNCH):** Permite a sincronização de itens de gerenciadores de informações pessoais. É também referido como IrDA visto que esse perfil originalmente fazia parte das especificações da tecnologia IrDA, sendo

então adotado pela Bluetooth SIG como um perfil principal da tecnologia Bluetooth.

SEGURANÇA

O Bluetooth tem três modos de segurança, variando desde nenhuma segurança até total criptografia de dados e controle de integridade (TANENBAUM, 2003).

Supõe-se que os dois dispositivos – mestre e escravo - compartilham uma chave secreta configurada com antecedência. Em alguns casos, ambos são fisicamente conectados pelo fabricante (por exemplo, um fone de ouvido e um telefone celular vendido como uma unidade). Em outros casos, um dispositivo (por exemplo, o fone de ouvido) tem uma chave embutida no código e o usuário tem de digitar essa chave no outro dispositivo (por exemplo, o telefone celular) como um número decimal. Essas chaves compartilhadas são chamadas chaves de passagem.

Para estabelecer um canal, o escravo e o mestre verificam se a outra máquina conhece a chave de passagem. Nesse caso, eles negociam para ver se esse canal será criptografado, terá sua integridade controlada ou ambos. Em seguida, eles selecionam uma chave de sessão aleatória de 128 bits, na qual alguns bits podem ser públicos.

A criptografia utiliza uma cifra de fluxo chamada E0; o controle de integridade emprega o SAFER+. Ambos são cifras de blocos de chave simétrica tradicional. O SAFER+ foi submetido aos rígidos testes de aprovação do AES, mas foi eliminado na primeira rodada, porque era mais lento que os outros candidatos.

Bluetooth realiza a autenticação apenas de dispositivos, não de usuários; assim, o furto de um dispositivo Bluetooth pode dar ao ladrão acesso às finanças e às outras contas do usuário. No entanto, o Bluetooth também implementa segurança nas camadas superiores. Portanto, até mesmo na eventualidade de uma violação da segurança no nível de enlace, deve restar alguma segurança, especialmente para aplicações que exigem a digitação de um código PIN em algum tipo de teclado para completar a transação.

4.3 - REDES 802.11

As redes IEEE 802.11 são também conhecidas como redes Wi-Fi (Wireless Fidelity) e são muito difundidas atualmente sendo suportadas por diversos dispositivos como computadores, consoles de vídeo game, celulares e outros. Atualmente existem diversas versões do protocolo que variam em termos de largura de banda, algoritmos criptográficos ou simplesmente nas frequências emitidas para atender a requisitos legais de determinados paises.

As redes 802.11 podem funcionar com ou sem a presença de uma estação base. O primeiro caso - ver Figura 21 (a) -, descreve as Redes Infraestruturadas (ou BSS – *Base Service Set*) onde toda a comunicação deve passar pela estação base (chamada ponto de acesso), formando o que se chama de célula. No outro caso, chamado redes Ad-Hoc (ou IBSS – *Independent Basic Service Set*), os computadores simplesmente transmitiriam diretamente uns para os outros. Um exemplo típico é de duas ou mais pessoas juntas em uma sala não equipada com uma LAN sem fio, fazendo seus computadores se comunicarem diretamente conforme ilustra a Figura 21 (b).

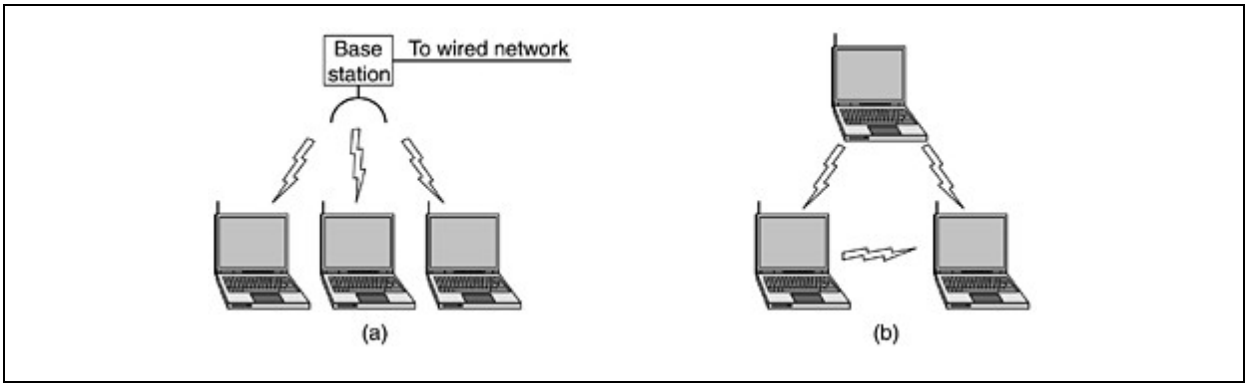

Figura 21 - Modos de operação de uma rede 802.11.
Fonte: (TANENBAUM, 2003)

O alcance das redes 802.11 frequentemente é de algumas centenas de metros (TANENBAUM, 2003). Devido ao fato de ser projetada para atender aos requisitos para aplicações de LAN sem fio, redes 802.11 possuem consumo de energia bastante elevado em comparação com alguns outros padrões. Tecnologias como Bluetooth (projetado para suportar aplicações wireless PAN) fornecer uma gama muito menor de propagação e assim, em geral,

têm um menor consumo de energia (IEEE, 2006). O alto consumo de energia do Wi-Fi torna a vida da bateria em dispositivos móveis, uma preocupação. Os principais padrões 802.11 e suas respectivas velocidades são descritas na Tabela 7.

Tabela 7 - Características dos principais padrões 802.11
Fonte: Elaborado pelo autor (2010)

Versão	Velocidade
802.11ª	54Mbps
802.11b	11Mbps
802.11g	54Mbps
802.11n	600Mbps

O Padrão IEEE 802.11 descreve três diferentes classes de quadros: dados, controle e gerenciamento, cada qual com um cabeçalho diferente. Os seus serviços divididos nas categorias Serviços de Distribuição que são fornecidos pela estação-base e lidam com a mobilidade das estações à medida que entram e saem das células e os serviços Intracélula que relacionam-se com a ações dentro da célula e são usados depois que ocorre a associação. São eles: (TANENBAUM, 2003)

Serviços de Distribuição

- **Associação**: utilizado pelas estações móveis para conectá-las às estações-base.
- **Desassociação**: utilizado pela estação móvel ou pela estação-base para interromper o relacionamento.
- **Reassociação**: utilizado no deslocamento de uma célula para outra.
- **Distribuição**: determina como rotear quadros enviados à estação-base
- **Integração**: converte o quadro do formato IEEE 802.11 para o formato exigido pela rede de destino.

Serviços Intracélula

- **Autenticação**: após a associação, a estação-base envia um quadro de "desafio" especial para permitir a autenticação da estação móvel. Esta criptografa o quadro

de desafio e devolve à estação base. Se o resultado for correto a estação é registrada na célula.

- **Desautenticação**: utilizado no momento que uma estação autenticada deseja deixar a rede

- **Privacidade**: este serviço administra a criptografia e decriptografia (ALGORITMO RC4)

- **Entrega de dados**: serviço para transmitir e receber dados, sem confiabilidade.

SEGURANÇA

Quando a segurança do protocolo 802.11 é ativada, cada estação tem uma chave secreta compartilhada com a estação base. A forma como as chaves são distribuídas não é especificada pelo padrão. Elas poderiam ser pré-carregadas pelo fabricante, trocadas com antecedência pela rede fisicamente conectada. Finalmente, a estação base ou máquina do usuário poderia escolher uma chave ao acaso e enviá-la à outra máquina pelo ar, codificada com a chave pública da outra máquina. Uma vez estabelecidas, em geral as chaves permanecem estáveis por meses ou anos. Um dentre os algoritmos criptográficos que seguem podem ser utilizados:

- **WEP - *Wired Equivalent Privacy***: o objetivo do uso do WEP é garantir a confidencialidade e a integridade das informações na rede Wireless. É o protoloco original de autenticação e criptografia definido pelo IEEE 802.11, sua chave varia de 40 e 128 bits (opcional). Possui um vetor de inicialização de 24 bits e é transmitido em texto claro, isso diminui consideravelmente a força do algoritmo. Existem ferramentas que ficam capturando os vetores de inicialização dos quadros e dessa forma conseguem decifrar a chave. Utiliza o protocolo RC4 para cifrar os dados.

- **TKIP - *Temporal Key Integrity Protocol***: foi criado com o propósito de contornar os problemas do WEP. Usa chave de 128 bits, o vetor de incialização é 48 bits e também utiliza o protocolo RC4 para cifrar os dados. Utiliza uma chave por pacote

(*per-packet key mixing*). Cada estação combina a sua chave com seu endereço MAC para criar uma chave de criptografia que é única. A chave compartilhada entre o ponto de acesso e os clientes wireless são trocadas periodicamente. Esse é o motivo pela qual falamos que o TKIP utiliza chaves dinâmicas de criptografia. Dessa forma fica mais difícil para um atacante quebrar a chave, mesmo que ele consiga, a vida útil da chave será pequena.

- **WPA - *Wi-Fi Protected Access***: Em 2001 a IEEE começou o desenvolvimento do padrão IEEE 802.11i com intuito de prover maior segurança em redes Wireless. Em 2002 a Wi-Fi Alliance optou por usar o que já estava pronto desse padrão, nesse momento ainda não estava completo o IEEE 802.11i. Assim surgiu o WPA que representa o pre-IEEE 802.11i. O WPA utiliza o TKIP para criptografia dos dados e padrão 802.1x(EAP) para autenticação. Existe também a possibilidade de utilizar o WPA-PSK que traz as vantagens do WPA sem a necessidade de um servidor RADIUS. A autenticação ocorre com uma chave compartilhada, parecido com o WEP. Depois que acontece a autenticação deriva-se outra chave para a criptografia dos quadros.

- **WPA2**: agregou vários itens do WPA, como o uso do IEEE 802.1x/EAP e adicionou novidades, como a utilização do algoritmo forte de criptografia, o AES (*Advanced Encryption Standard*). É o mecanismo mais seguro.

4.4 –NEAR FIELD COMMUNICATION (NFC)

Near Field Communication (NFC) é um padrão definido pela NFC Fórum, um consórcio global de hardware, software, empresas de cartão de crédito, bancos, provedores de-rede, e outros que estão interessados na promoção e padronização da tecnologia. As normas que definem a tecnologia NFC são as normas ISO 18092, ISO 14443 tipo A e ISO 14443 tipo B e FeliCa (Felicity Card – padrão para *smart card* sem contato desenvolvido pela Sony).

A aplicabilidade de chips NFC é imensa. Chips NFC podem conter dados simples ou instruções que lhes permitam desempenhar funções como destravar portas, pagar as mercadorias, troca de dados entre pessoas e outros. Recentemente a tecnologia NFC vem sendo muito comentada pelos sites especializados, como a tecnologia que vai revolucionar as transações eletrônicas em *smart phones*, contudo a tecnologia já é utilizada em celulares desde 2004, conforme pode ser verificado no Anexo A.

A intenção principal da tecnologia NFC é ser utilizada em conjunto com dispositivos de telefonia móvel. Existem três usos específicos de acordo com NFC Fórum:

- **Modo leitura/escrita**: o dispositivo NFC está ativo e lê uma etiqueta RFID passiva. Exemplo: cartazes inteligentes.

- **Modo P2P**: dois dispositivos NFC trocam dados. Exemplo: cartões de visita virtual, fotos digitais.

- **Modo Emulação de Cartão**: O dispositivo NFC se comporta como um cartão de contato existente e pode ser usado com as atuais infraestruturas tecnológicas.

NFC utiliza uma comunicação por radio de curto alcance que opera na frequência 13.56 MHz e que possibilita a transmissão de dados a uma velocidade de até 424 quilobits por segundo. A comunicação NFC é disparada quando dois dispositivos compatíveis com NFC são aproximado a até no máximo 4 cm. Como o alcance da transmissão é curto, as transações baseadas em NFC são inerentemente seguras. O tempo de conexão entre dois dispositivos também é rápido e normalmente inferior a um décimo de milissegundo. (ORACLE, 2011) O consumo de energia da um módulo NFC é inferior a 12 mA. (WIKIPEDIA, 2011)

A comunicação entre dispositivos NFC pode ser realizada partir de dois modos: comunicação passiva e comunicação ativa.

- **Comunicação passiva**: o dispositivo que inicia a comunicação fornece um campo de transporte e o dispositivo alvo responde através da modulação do campo existente. Nesse modo o dispositivo alvo pode tirar a sua potência de

funcionamento a partir do campo eletromagnético fornecido pelo dispositivo iniciador.

- **Comunicação ativa**: tanto o iniciador quanto o dispositivo alvo se comunicam alternadamente gerando seus próprios campos eletromagnéticos. Um dispositivo desativa seu campo de rádio frequência enquanto espera por dados. Neste modo ambos os dispositivos necessitam de fontes de alimentação própria.

4.5 - COMPARATIVO DAS TECNOLOGIAS SEM FIO

O padrão 802.11 (Wi-Fi), conforme visto anteriormente, é um padrão para redes WLAN enquanto que os padrões Bluetooth e IrDA são projetados para redes WPAN. Realizar uma comparação com tecnologias destinadas a diferentes tipos de rede inviabiliza uma comparação justa, pois tipos de redes diferentes têm diferentes propósitos.

Contudo diversos *smart phones* modernos e outros dispositivos tipicamente utilizados em redes WPAN que possuem restrições energéticas significativas, por questões de interoperabilidade, são atualmente compatíveis com o padrão 802.11. Por isso as características das três tecnologias serão comparadas e as devidas considerações serão realizadas de acordo com o cenário de utilização.

Nesta seção será realizado um comparativo de acordo com os seguintes critérios: 1-propagação do sinal (omnidirecional x direcional), 2-taxa de transferência, 3-alcance, 4-segurança, 5-consumo energético e 6-custo.

1. **Propagação do sinal (omnidirecional x direcional)**: tanto Bluetooth quanto o padrão Wi-Fi possuem uma natureza omnidirecional contrapondo-se ao IrDa que tem uma natureza direcional e, essas características opostas, possuem vantagens e desvantagens que não podem satisfazer a todas as necessidades dos usuários (SUBIR KUMAR SARKAR, 2008). Com uma tecnologia omnidirecional, por exemplo, uma pessoa pode sincronizar seu celular com um computador pessoal sem tirar o celular do bolso ou da bolsa, pois a capacidade omnidireccional

possibilita que a sincronização se inicie quando os dispositivos estão no mesmo raio de alcance, o que não é possível com IrDA. De forma contrária, IrDA leva uma vantagem em aplicações que envolvam troca de dados ponto-a-ponto. Em uma mesa de reunião, por exemplo, cartões eletrônicos podem ser trocadas entre duas pessoas, apontando seus dispositivos IrDA para o outro; isso não é possível utilizando Bluetooth ou Wi-Fi porque o dispositivo irá detectar todos os dispositivos similares no ambiente e o usuário teria que selecionar a pessoa pretendida dentre todos os identificados. A comunicação via NFC é omnidirecional, mas a sua pouca abrangência e o fato de ter sido concebido para a comunicação entre dois dispositivos tornam esse fator irrelevante e pode ser comparado ao padrão IrDA.

2. **Taxa de transferência:** conforme visto anteriormente a tecnologia Wi-Fi possui as mais altas taxas de transferência, seguida, respectivamente, por IrDA, Bluetooth e NFC;

3. **Alcance**: Wi-Fi é destinado a redes locais que por definição são redes maiores que as redes pessoais, dessa forma possibilitam um maior alcance. Dispositivos *Bluetooth* possuem alcance pequeno, porém maiores que o alcance dos dispositivos IrDA que são maiores que o alcance proporcionado por NFC.

4. **Segurança**: a tecnologia Bluetooth oferece mecanismos de segurança que não estão presentes no IrDA (MULLER, 2000) (SUBIR KUMAR SARKAR, 2008). Da mesma forma, a tecnologia Wi-Fi possui algoritmos criptográficos mais seguros que o *Bluetooth*, como o algoritmo AES que é mais seguro que o algoritmo RC4 (TANENBAUM, 2003). O NFC não possui mecanismos de segurança, contudo a extrema proximidade que o dispositivo deve estar do outro para que possam se comunicar pode ser considerado um mecanismo de segurança.

5. **Custo**: dispositivos IrDA são mais baratos que dispositivos Bluetooth e dispositivos *Bluetooth* são mais baratos que dispositivos Wi-Fi. Módulos NFC possuem custo intermediário entre IrDA e *Bluetooth*.

6. **Consumo energético**: dispositivos IrDA consomem menos energia que dispositivos Bluetooth e estes consomem menos energia que dispositivos Wi-Fi. Módulos NFC consomem mais energia que módulos Bluetooth classes 2 e 3 e menos energia que a classes 1 Bluetooth.

DIVINEY (2003) sintetiza os cenários de aplicabilidade das tecnologias IrDA, *Bluetooth* e WiFi:

> *"Com sua grande variedade e seu suporte completo ao modelo TCP/IP, a tecnologia Wi-Fi é a escolha óbvia para realizar uma conectividade sem fio em um escritório."*
>
> *"Bluetooth representa um candidato ideal para a manipulação Personal Area Network (PAN) de tráfego, incluindo telefonia local conectividade de rede, alcance limitado, e outros usos comuns."*
>
> *"Nos casos em que um ad-hoc, ponto-a-ponto de troca de dados é necessária, IrDA é o vencedor claro. "*

Em ORACLE (2011) é destacada como grande vantagem da tecnologia NFC o tempo de emparelhamento pelo fato de não é realizada qualquer tipo de criptografia, uma vez que a segurança baseia-se na proximidade. Com relação NFC x Bluetooth é sugerida uma combinação das duas tecnologias:

> *"Um bom cenário é a combinação de NFC e Bluetooth, NFC, onde é utilizado para emparelhar (autenticar) e uma sessão de Bluetooth utilizado para a transferência de dados"*

CAPÍTULO 5
- COMUNICAÇÃO SERIAL

Dispositivos podem ser ligados fisicamente a computadores e a outros dispositivos por diversos tipos de interface serial. Entre os tipos de interface serial é possível citar: USB (Universal Serial Bus), RS-232, Ethernet e outros. As portas seriais são ideais para muitas comunicações entre sistemas embarcados ou entre sistemas embarcados e computadores (AXELSON, 2007, p. 1).

Este capítulo descreve características básicas do padrão USB e do protocolo de comunicação UART. O intuito principal é descrever elementos que possibilitarão explicar o desenvolvimento do dispositivo SCREAD MOD, que é o foco desta dissertação, bem como justificar escolhas.

5.1 –*UNIVERSAL ASYNCHRONOUS RECEIVER / TRANSMITTER* (UART)

Um transmissor/receptor assíncrono universal (ou UART) é um circuito integrado (ou parte de um circuito integrado) que possibilita comunicação de dados de forma assíncrona e que converte a transmissão dos dados da forma serial para paralela e vice-versa. Dispositivos UART são comumente incluídos em microcontroladores e também são utilizados em conjunto com portas seriais RS-232 (AXELSON, 2007).

Quando utilizados em conjunto com portas seriais RS-232, os níveis lógicos, detalhes sobre os níveis de tensão, taxa de variação, e comportamento de curto-circuito são geralmente controlados por um controlador, que converte níveis lógicos do UART (níveis lógicos TTL[12])

[12]Na lógica de sinais dos circuitos TTL é possível uma variação 0,4V sem causar erros, pois os seguintes intervalos devem ser obedecidos: saída lógica baixa <= 0.4V; entrada lógica baixa <=0.8V; saída lógica alta>2,4V; entrada de lógica alta >= 2V (AXELSON, 2007).

para sinais compatíveis RS-232, e um receptor[13] que converte os sinais RS-232 para sinais compatíveis com os níveis lógicos UART. A Figura 22 resume a relação entre dispositivos UART, lógica de comunicação TL e TTL e o padrão de comunicação RS-232.

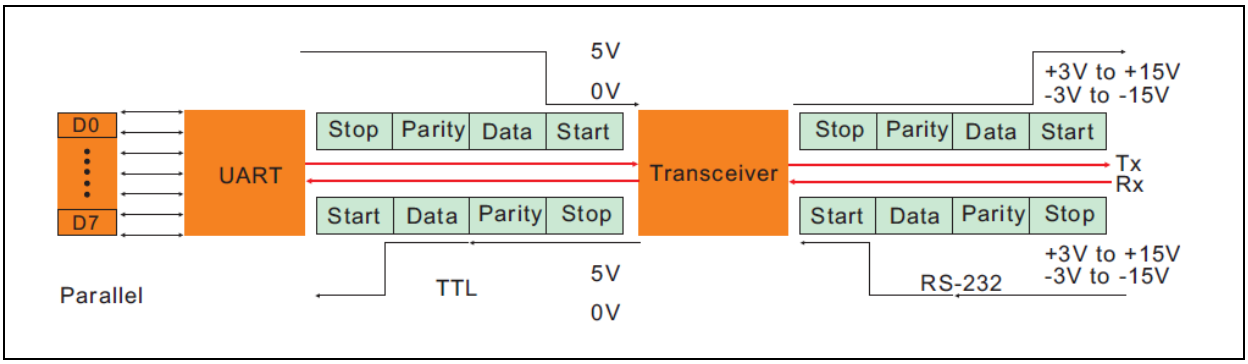

Figura 22 - Relação entre UART, sinal TTL e sinal RS-232

Fonte: http://www.rg2i.com/moxa/pdf_files/CH10/ch10_The_Basics_of_RS-232_422_485.pdf

Deve-se destacar também que a forma como dispositivos UART se comunicam, ou o protocolo de comunicação utilizado por dispositivos UART, é também conhecido como protocolo de comunicação UART. Dessa forma o termo UART pode referir-se tanto a dispositivos quanto ao protocolo de comunicação que envolve questões como paridade, bits de início de finalização, nível lógico e outros.

O protocolo de comunicação UART define uma transmissão de dados em blocos, muitas vezes chamados palavras. Cada palavra contém um bit de inicialização (*start bit*), *bits* de dados, um *bit* de paridade (*parity bit*) opcional, e um ou mais bits de parada (*stop bits*) (AXELSON, 2007).

Como é possível perceber na Figura 22, antes da transmissão de cada nova palavra um *bit* de inicialização (*start bit*) é adicionado. O *start bit* é usado para alertar o receptor que uma palavra de dados está prestes a ser enviada e para forçar a sincronização do relógio do receptor com o relógio do transmissor.

[13] O circuito integrado MAX 232 é um exemplo de dispositivo muito utilizado que permite fazer a conversão de sinais RS-232 para sinais compatíveis com TTL.

Após o envio completo de uma palavra, o transmissor pode adicionar um *bit* de paridade (*parity bit*). O *bit* de paridade pode ser usado pelo receptor para executar verificação de erros simples.

Após o receptor ler os dados, busca um *bit* de parada (*stop bit*). Se o *stop bit* não aparece quando é suposto, o UART considera a palavra inteira ilegível e reportará um erro de enquadramento (*framing error*) para o processador *host* quando a palavra de dados é lida. A causa usual de um *framing error* é que o emissor e o receptor estão trabalhando com velocidades diferenciadas, ou que o sinal foi interrompido. É comum a utilização de até dois *bits* de parada para velocidades acima de 1200bps. Se outra palavra está pronta para ser enviada, o *start bit* da nova palavra somente será enviado após o envio do *stop bit* da palavra anterior.

Muitos microcontroladores contêm um ou mais portas de comunicação UART (AXELSON, 2007). Entre os microcontroladores da Microchip é possível citar microcontroladores das famílias 18F (ex.: 2455, 2550, 4455, 4550), famílias 16F (873A, 874A,876A, 877A) e outros. Quando um UART em *hardware* não está disponível, o *software* embarcado no microcontrolador pode emular o protocolo UART, tipicamente com a ajuda de um temporizador (AXELSON, 2007). O desenvolvimento do protocolo UART em *software* ocorreu no desenvolvimento do SCREAD MOD, pois o microcontrolador utilizado disponibilizava apenas uma única porta UART em *hardware*.

5.2 – UNIVERSAL SERIAL BUS (USB)

A topologia física do barramento USB é uma topologia estrela em camadas (Figura 23). O centro cada estrela é um *hub* e cada conexão ao hub é um ponto da estrela. O *hub* raíz é chamado *host*. A estrela em camadas apenas descreve a conexão física. Na programação, o que importa é a conexão lógica. Os aplicativos *host* e o *firmware* do dispositivo não precisam saber ou se importar se a comunicação passa por um *hub* ou cinco, por exemplo. (AXELSON, 2009)

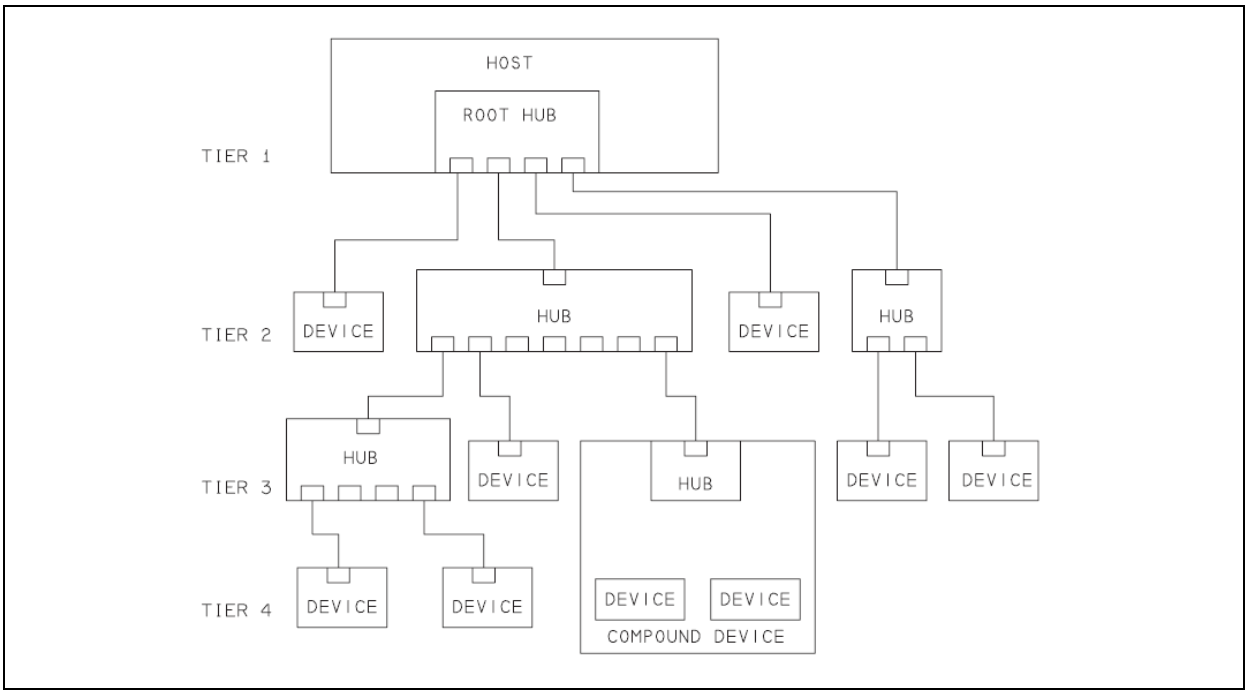

Figura 23 - Topologia física do barramento USB
Fonte: (AXELSON, 2009)

O *host* é responsável por detectar a inclusão e remoção de dispositivos, gerenciar o fluxo de controle de dados entre os dispositivos conectados; Fornecer alimentação (tensão e corrente) aos dispositivos conectados, monitorar os sinais do bus USB.

Podem existir até 127 dispositivos conectados em uma rede USB. O conector utilizado pelos dispositivos para se conectar às portas podem ser de dois tipos: conector tipo A e conector tipo B. Cada tipo de conector pode possuir um de três possíveis tamanhos: tamanho padrão, tamanho mini e tamanho micro. O conector Micro USB Tipo A é também conhecido como conector tipo AB. A Figura 24**Erro! Fonte de referência não encontrada.** resume os seis possíveis conectores USB.

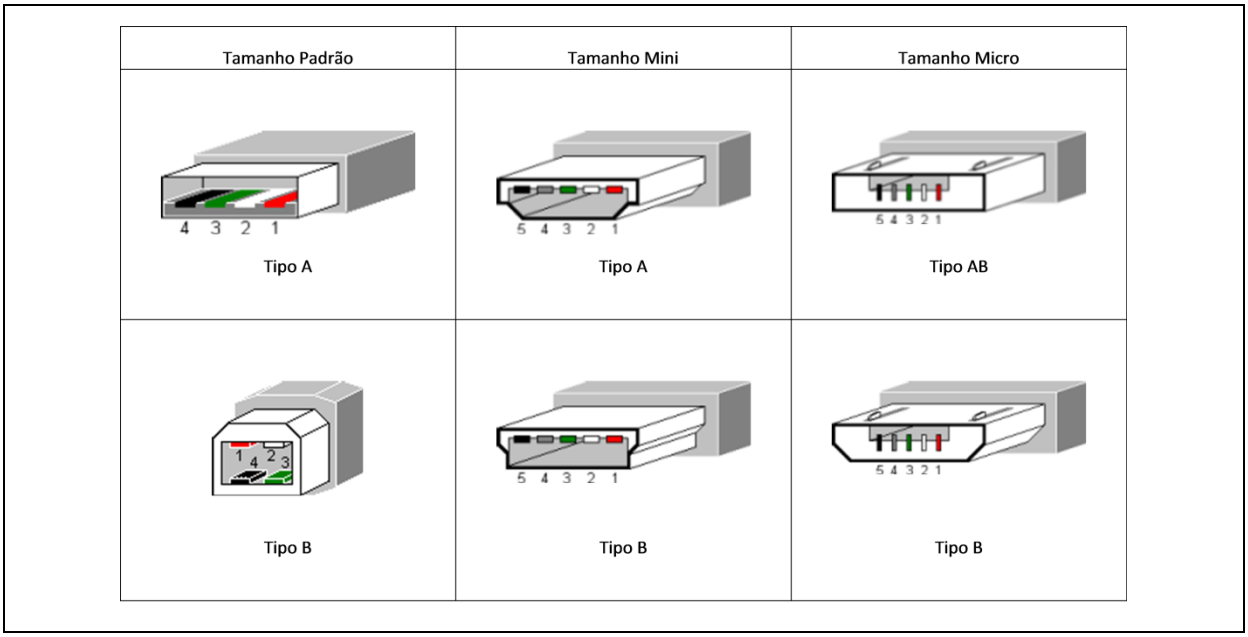

Figura 24 - Conectores USB

Fonte: http://www.lammertbies.nl/comm/cable/USB-connector.html

USB OTG – USB ON-THE-GO

A comunicação USB exige um e apenas um controlador *host* (BEYOND LOGIC, 2010). No entanto, a On-The-Go é uma especificação trazida pelo USB 2.0 que introduziu um protocolo *Host Negotiation* que permite que dois dispositivos possam negociar o papel de *Host*. O objetivo disto é atender a conexões ponto a ponto, como por exemplo, uma conexão realizada entre um telefone móvel e outro dispositivo e não para múltiplos dispositivos ou múltiplos *hubs*. O *host* USB é responsável pela realização de todas as operações e programação de largura de banda. Os dados podem ser enviados por diferentes métodos de transação utilizando um protocolo baseado em *token*. (BEYOND LOGIC, 2010)

Dispositivos móveis comuns que suportam conexões USB em sua maioria funcionam apenas como USB *Devices* (ou periféricos), incluindo telefones celulares, câmeras fotográficas e outros. Isso significa que um celular comum não pode ser conectado por um cabo USB a uma máquina fotográfica comum pela porta USB, porém ambos podem se conectar a um computador pessoal porque o computador que funciona como USB host. Impressoras mais modernas são exemplos de dispositivos que implementam o padrão USB OTG, pois quando ligadas a uma porta USB de um computador funcionam como *Devices*,

porém quando conectadas a câmeras digitais para possibilitar uma impressão direta funcionam como um host.

Dispositivos USB OTG quando funcionando como *hosts* possuem algumas restrições quando comparados a um *host* convencional. Um dispositivo OTG, por exemplo, não possui suporte a hubs externos, múltiplos dispositivos conectados ao mesmo tempo alternar entre velocidade alta e baixa (AXELSON, 2009). A Tabela 8 contempla um comparativo com os recursos suportados por ambas as tecnologias.

Tabela 8 - Comparativo USB 2.0 Host convencional x USB OTG funcionando como Host
Fonte (AXELSON, 2009)

Capacidade ou Recurso	USB 2.0 Host convencional	USB 2.0 OTG Device Funcionando como Host
Comunicação em *high speed*	Sim	Opcional
Comunicação em *full speed*	Sim	Sim
Comunicação em *low speed*	Sim	Opcional (não permitido em modo dispositivo)
Suporta hubs externos	Sim	Opcional
Fornece uma lista de periféricos de destino	Não	Sim
Funciona como um periférico	Não	Sim (quando não está funcionando como host)
Suporta *Session Request Protocol*	Opcional	Sim
Suporta *Host Negotiation Protocol*	Não	Sim
Corrente mínima por porta	500 mA (100mA se alimentado por bateria)	8 mA
Desliga VBUS quando não está sendo utilizado	Não	Sim
Conecotr	1 ou mais. Padrão A	1 Micro AB

O microcontroladores PIC24FJ128DA106 é um exemplo de microcontrolador compatível com USB-OTG.

CAPÍTULO 6
- NORMA ISO/IEC 7816

A comunicação de um computador ou de um dispositivo processado com um cartão inteligente pressupõe atender a uma série de padrões para tornar a comunicação possível. Os cartões inteligentes, foco deste trabalho, são aqueles que atendem à norma ISO 7816, pois tal norma é compatível com *smart cards* utilizados por Autoridades Certificadoras pertencentes à ICP-Brasil. A norma ISO 7816 se referencia a cartões inteligentes (ou *smart cards*) como ICC (*Integrated Circuit Card*). Os leitores de cartões inteligentes são também referenciados como CAD (*Card Acceptance Device*) e a norma ISO costuma referenciá-los como IFD (*Interface Device*). Dessa forma qualquer uma das nomenclaturas "cartão inteligente" e "*smart card"* ou "ICC" poderá ser utilizada de forma indistinta, assim como os nomes "CAD", "IFD" ou simplesmente "leitor de cartão".

Este capítulo inicia-se com a definição de tipos de *smart card* e contem os pontos relevantes da norma ISO-7816 para a construção e teste do dispositivo SCREAD MOD. No fim do capítulo, são apresentados os passos necessários para a realização de uma assinatura digital utilizando uma comunicação em baixo nível através de comandos reconhecidos pelos cartões.

6.1 - TIPOS DE SMART CARD

Smart cards são comumente referenciados como *Chip Card* ou ICC (*Integrated Circuit Card*). ICC's podem ser classificados de acordo com funcionalidade provida pelo chip e de acordo com a forma como ele se comunica com o CAD.

Com relação à funcionalidade do chip, existem dois tipos diferentes de chips que podem ser classificados em:

- **ICC de memória**

- Memória com segurança lógica (EVERETT, 2002) ou Memória com lógica rígida (U.S. GSA, 2004): fornece criptografia e autenticação de acesso à memória e seu conteúdo. Fornece um sistema de arquivo estático para apoiar múltiplas aplicações, com acesso codificado opcional para o conteúdo da memória. Seus sistemas de arquivos e conjunto de comando só podem ser alterados reprojetando a lógica do IC.

- Somente memória: Apenas armazenam dados, não sendo capazes de processar qualquer tipo de lógica ou cálculo.

- **ICC Microcontrolado**: possui um microcontrolador, um sistema operacional e memória do tipo leitura/escrita que pode ser atualizada várias vezes. Contém e executa a lógica e os cálculos e armazena os dados de acordo com seu sistema operacional. Tudo o que precisa para funcionar é alimentação de energia e um terminal de comunicação.

Com relação à forma de interagir com o CAD, podem ser classificados em *smart cards* de contato e *smart cards* sem contato. A Figura 25 contém um resumo com as classificações adotadas para cartões inteligentes.

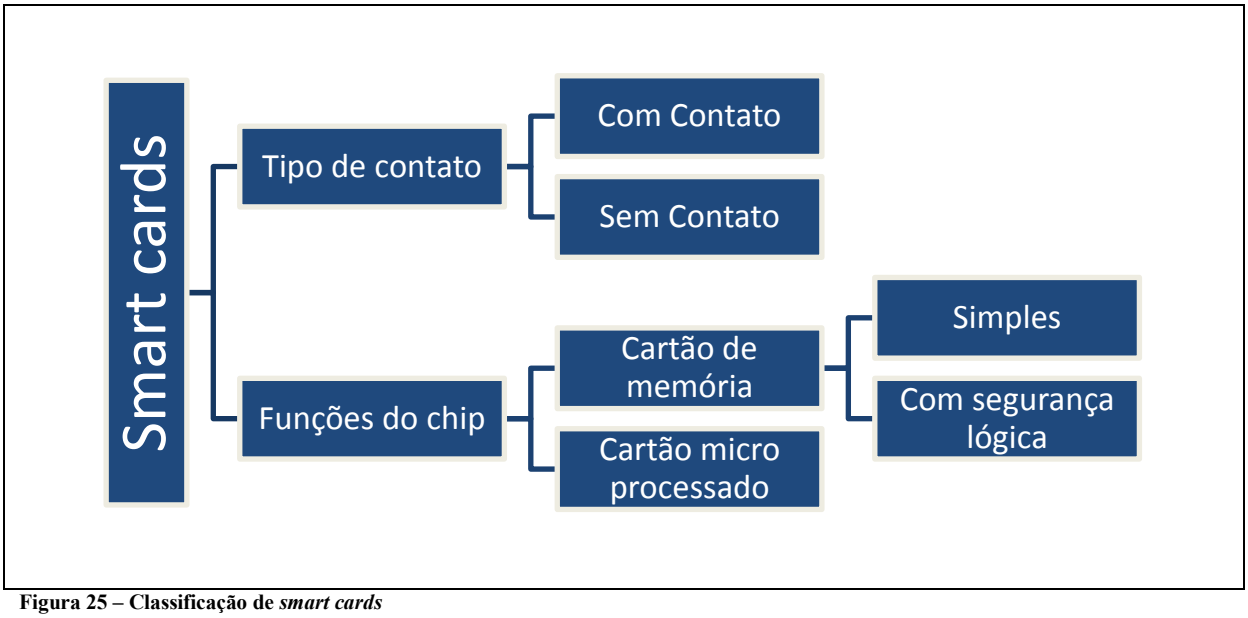

Figura 25 – Classificação de *smart cards*

Fonte: Elaborado pelo autor (2011)

6.2 - ORGANIZAÇÃO DA NORMA

A norma ISO/IEC 7816 se refere a um conjunto de quinze normas que tratam de diferentes questões relacionadas à *smart cards*. Por exemplo, a norma 7816-1 se refere a características físicas de *smart cards* microcontrolados com contato que inclui, entre outras coisas, especificação dos limites de exposição para um número de fenômenos eletromagnéticos. A Tabela 9 inclui os documentos que compõem o conjunto de normas ISO/IEC 7816.

Tabela 9 – Norma ISO/IEC 7816 e suas divisões
Fonte: (ISO/IEC 7816-1, 2011)
Nota: Adaptado pelo autor

Norma	Aplicação
7816-1	Características físicas dos cartões com contato
7816-2	Dimensões e localização dos contatos
7816-3	Sinais eletrônicos e protocolos de transmissão
7816-4	Comandos para comunicação – protocolo APDU
7816-5	Sistema de numeração e identificadores de aplicação
7816-6	Dados para intercâmbio interindústria
7816-7	Comando interindústria SCQL
7816-8	Comandos para operações seguras
7816-9	Comandos para gerenciamento de cartões
7816-10	Sinais eletrônicos e ATR para cartões síncronos
7816-11	Verificação pessoal através de métodos biométricos
7816-12	Cartões com contato USB – Interfaces elétricas e procedimentos operacionais
7816-13	Comandos para gerenciamento de aplicações em ambientes multiaplicações
7816-15	Aplicações de informações criptográficas

A parte ISO 7816-1 define as características físicas de um cartão, especificando, por exemplo, níveis de tolerância de um cartão quando ele é dobrado ou flexionado. Conexões entre os conectores de superfície e de I/O, pinos dos moldes de silicone incorporados devem ser mantidos além de suportar o stress mecânico. Isto é para se ter certeza de que os cartões de plástico com chips embutidos são manufaturados em uma maneira que garanta um funcionamento sem problemas durante o tempo de vida médio de um cartão. Tal informação é importante para os fabricantes de cartão. Eles são os que escolhem as matérias e estabelecem um processo que incorpora o circuito integrado do cartão e não será tratada neste capítulo.

6.3 - 7816-2 PARTE 2: DIMENSÕES E LOCALIZAÇÃO DOS CONTATOS

ISO 7816-2 define as dimensões e localização dos contatos. Esta parte inclui normas sobre a função, número e posição dos contatos elétricos.

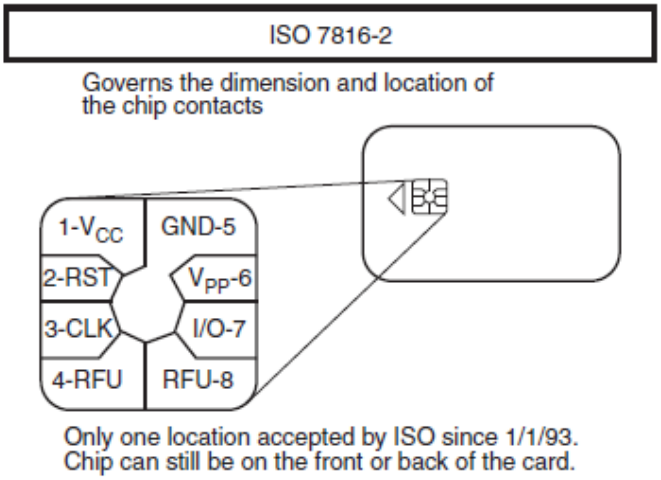

Figura 26 - Conector ISO 7816-2

Fonte: http://pro.rezel.net/~tem/elec381/wiki/lib/exe/fetch.php?media=smartcard_interface.pdf

A placa do cartão de circuito integrado (ICC) tem oito contatos elétricos, conforme pode ser visto na Figura 26. Eles são referidos como C1 a C8. No entanto, apenas seis contatos elétricos são conectados ao chip microprocessador embutido e os outros dois permanecem reservados para uso futuro. A Tabela 10 descreve a finalidade de cada um dos contatos.

Tabela 10 - Contatos de um *smart card*

Fonte: ISO 7816-3

Contato	Objetivo
C1 - Vcc	Alimentação elétrica de 5 ou 3,3V [14]
C2 - RST	Linha através da qual o IFD pode sinalizar ao chip do cartão inteligente microprocessador para iniciar a sua sequência de instruções de *reset*.

[14] A alimentação elétrica faz parte do ISO 7816-3.

C3 - CLK	Linha de sinal de *clock* através do qual o temporizador pode ser fornecido ao microprocessador do chip. Utilizado para controlar a velocidade de operação e prover um quadro comum de comunicação entre o IFD e o ICC.
C4 - RFU	Reservado para uso futuro
C5 - GND	Terra. Proporciona aterramento comum entre as IFD e ICC.
C6 - Vpp	Conexão de energia usada para programar EEPROM de ICCs primeira geração.
C7 - I/O	Linha de entrada/saída que prove um canal de comunicação hal-duplex entre o leitor e entre o smart card.
C8 - RFU	Reservado para uso futuro.

A localização de cada um dos contatos de um *smart card* está descrito na Figura 27 tomando-se como referência a distância de cada contato a partir do canto superior esquerdo do cartão.

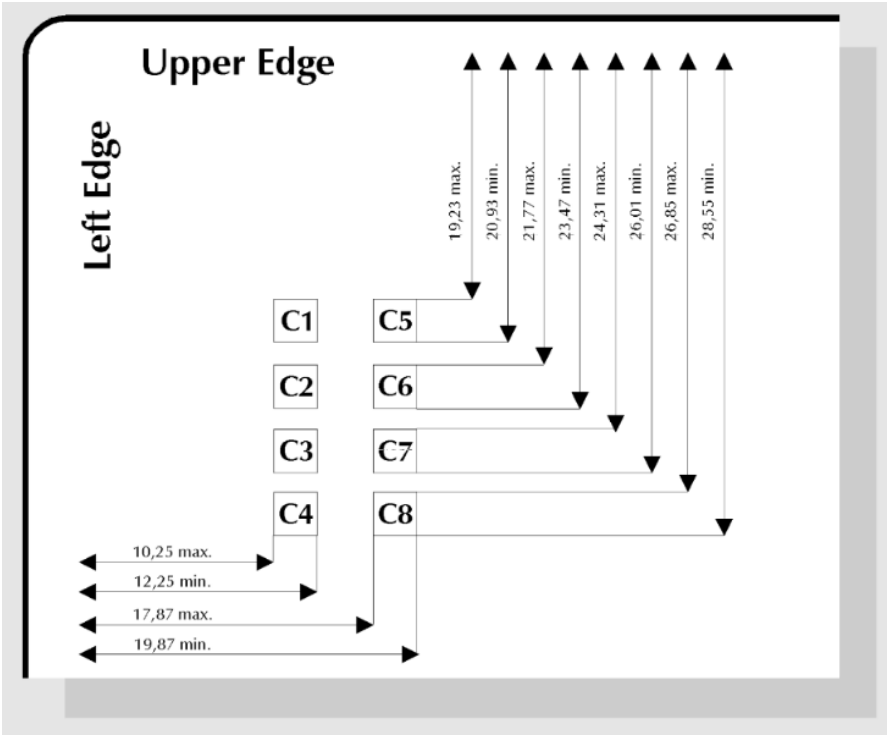

Figura 27 - Localização dos contatos de um smart card

Fonte: http://kbts.neic.nsk.su/satxpress/SmartCard/info/FIG9.GIF. Acessado em 14/04/2011

6.4 - 7816-3: SINAIS ELETRÔNICOS E PROTOCOLOS DE TRANSMISSÃO

Descreve sinais eletrônicos e protocolos de transmissão de cartões de circuito integrado. A maioria das especificações presentes em ISO 7816-3 é importante para os fabricantes de IFD's ou desenvolvedores que queiram estabelecer uma comunicação com um cartão inteligente em um nível muito baixo (o nível do sinal). Esta pode ser a comunicação a partir de um microcontrolador ou serial de um PC / paralelo / USB / porta PCMCIA.

As especificações relativas a tensões elétricas são muitas e não serão aqui descritas. Contudo, serão descritos os passos necessários para o entendimento da utilização da norma 7816-3.

MODELO DE COMUNICAÇÃO

O diálogo entre o dispositivo e a interface do cartão deverá ser realizado através de operações sucessivas:

1. **IFD → ICC**: Ligação e ativação dos contatos pelo dispositivo de interface (CAD).
2. **IFD → ICC**: Reiniciar o cartão (comando RESET).
3. **IFD ← ICC**: O cartão envia um *stream* de *bytes* ATR (*Answer To Reset*) que tem no máximo 33 *bytes*. Esta *string* contém parâmetros que especificam como a comunicação deve ocorrer, a nível físico (protocolo de transmissão, velocidade de transmissão, etc). Além disso, o ATR contém uma *string* denominada *history bytes*, que identifica o modelo do *smart card*, bem como algumas de suas capacidades. (PIGNATARO, 2006)
4. **IFD ←→ ICC**: Troca de informações posteriores entre o cartão e o dispositivo de interface.
5. **IFD → ICC**: Desativação dos contatos pelo dispositivo de interface.

COMUNICAÇÃO SERIAL ENTRADA/SAÍDA

A transmissão realizada pela maioria dos cartões microprocessadores é assíncrona. Como só há uma porta para entrada/saída, a comunicação é *half-duplex*. Há dois protocolos de

operação: protocolo T=0, que envolve transmissão de um fluxo de *bytes* e T=1 que envolve operação em blocos.

O tipo assíncrono de transmissão é semelhante à utilizada pelo conector serial RS232C presente nos computadores pessoais, embora o PC opere no modo *full duplex* (EVERETT, 2002). A transmissão de um único caractere (definido como 8 *bits*) requer uma sobrecarga de vários *bits*, conforme representado Figura 28.

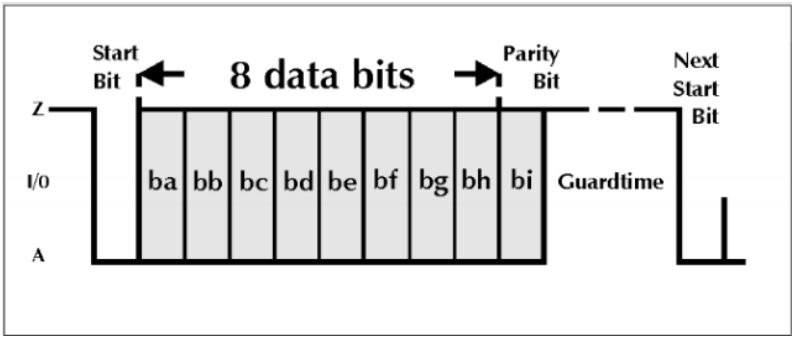

Figura 28 - Quadro de caracteres assíncronos
Fonte: (EVERETT, 2002)

O tempo de proteção é definido como sendo igual a dois períodos de *bit* (embora para o modo de blocos possa ser alterado para um período de um bit 1). Isso é semelhante a ter dois *bits* de parada em uma comunicação com um dispositivo UART, usado no PC. A interface RS232C define dois fios para transmissão e recepção, dessa forma é necessário uma modificação no hardware para permitir a comunicação direta a partir de um único fio. (EVERETT, 2002)

6.5 - 7816-4: ORGANIZAÇÃO, SEGURANÇA E COMANDOS PARA COMUNICAÇÃO

A norma 7816-4 organiza a troca de dados em quatro subtópicos: Par comando-resposta, objetos de dados, estrutura para aplicações e dados e arquitetura do sistema.

6.5.1 - PAR COMANDO-RESPOSTA

A comunicação entre uma aplicação e o cartão envolve o envio de um comando (chamado APDU – *Application Data Unit*) seguido de uma resposta do cartão para aplicação (também chamada APDU). Não pode haver comandos intermediários sem que um par comando-resposta seja concluído, isto é, um APDU de resposta deve ser recebido antes de ser iniciado um outro APDU de comando (ISO/IEC 1786-4, 2005). A Figura 29 contém a estrutura de um APDU de comando e a Figura 30 contém a estrutura de um APDU de resposta.

Field	Description	Number of bytes
Command header	Class byte denoted CLA	1
	Instruction byte denoted INS	1
	Parameter bytes denoted P1-P2	2
L_c field	Absent for encoding $N_c = 0$, present for encoding $N_c > 0$	0, 1 or 3
Command data field	Absent if $N_c = 0$, present as a string of N_c bytes if $N_c > 0$	N_c
L_e field	Absent for encoding $N_e = 0$, present for encoding $N_e > 0$	0, 1, 2 or 3

Figura 29 - APDU de commando
Fonte: (ISO/IEC 1786-4, 2005)

O APDU de comando (Figura 29) contém um cabeçalho de quatro *bytes* contendo quatro campos, seguido de outros três campos:

- ***Class byte* (CLA)**: especifica a classe do comando. Cada classe contém um conjunto de instruções específicas para determinado tipo de aplicação. A Tabela 11 especifica os possíveis valores que o campo CLA pode assumir. Bit 8 começando com 1 significa padrão proprietário, exceto quando todos os bits são 1 (FF) que é um valor inválido (ISO/IEC 1786-4, 2005);

- ***Instruction byte* (INS)**: especifica uma instrução específica dentro do conjunto de instruções;

- **Parâmetros P1 e P2**: indicam opções (ou parâmetros) para processamento dos comandos;

- **L_c**: especifica o número de bytes no campo *Command Data Field*; caso seja um valor entre '01' e 'FF' (1 byte) significa que N_c (*número de bytes no campo Coomand data field*) varia entre 1 e 255; caso seja um valor com o primeiro byte

fixo em ‘00’ e os dois próximos bytes variando entre ‘0001’ e ‘FFFF’ significa que o valor N_c varia de 1 a 65.535.

- ***Command data field***: os dados que compõem o comando;
- **L_e**: especifica o número máximo de bytes em *Response Data Field* (campo do APDU de resposta).

Response data field	Absent if $N_r = 0$, present as a string of N_r bytes if $N_r > 0$	N_r (at most N_e)
Response trailer	Status bytes denoted SW1-SW2	2

Figura 30 - ADPU de resposta
Fonte: (ISO/IEC 1786-4, 2005)

O APDU de resposta contém apenas dois campos:

- ***Response data field***: os dados da resposta com tamanho menor ou igual a Le;
- ***Response trailer***: especifica o status sobre o processamento do comando. SW1 é o primeiro byte do código de status e representa a categoria de erro. SW2 é o segundo byte do código de status e representa um status específico do comando ou o erro indicado.

Tabela 11 – Valores interindustriais do campo de cabeçalho *Class byte* (CLA) de um APDU
Fonte: (ISO/IEC 1786-4, 2005)

b8	b7	b6	b5	b4	b3	b2	b1	Significado
0	0	0	x	–	–	–	–	**Controle de encadeamento de comandos**
0	0	0	0	–	–	–	–	**-este comando é o último ou o único**
0	0	0	1	–	–	–	–	**-não é o último comando**
0	0	0	–	x	x	–	–	**Indicação de mensagens seguras (SM – Security Message)**
0	0	0	–	0	0	–	–	**-sem SM ou não indicado**
0	0	0	–	0	1	–	–	**-SM em formato proprietário**
0	0	0	–	1	0	–	–	**-SM de acordo com seção 6 da norma com cabeçalho não autenticado**
0	0	0	–	1	1	–	–	**-SM de acordo com seção 6 da norma com cabeçalho autenticado**
0	0	0	–	–	–	X	X	**Número do canal lógico variando de zero a três.**

6.5.2 - OBJETOS DE DADOS

Qualquer campo de dados quando codificado na estrutura TLV (*Tag, Length and Value*) é uma sequência de objetos de dados. Há duas categorias de objeto de dados: *simple*-TLV e BER-TLV.

SIMPLE-TLV (Simple - TAG, LENGTH AND VALUE)

Cada objeto de dados *SIMPLE-TLV* consiste de dois ou três campos consecutivos:

- ***Tag***: campo obrigatório que consiste de um único byte variando de 1 a 254. Os valores hexadecimais '00' e 'FF' são valores inválidos para o campo *Tag*. Se um registro for um objeto *SIMPLE-TLV,* então o campo *Tag* pode ser utilizado como um identificador de registro. Não codifica classes e construtores de tipos;

- ***Length***: campo obrigatório que especifica o tamanho e consiste de um ou de três bytes;

    - Se o primeiro byte não for 'FF', então o tamanho do campo consiste de um único byte codificado um número N que varia de 0 a 254.

    - Se o primeiro byte for 'FF' então o campo *Length* é codificado por um valor N codificado nos dois bytes subsequentes por um número que varia de 0 a 65.535.

- **Value:** campo que contém o valor armazenado. É opcional porque quando N é zero, não deve existir; quando N > 0 deve conter N bytes consecutivos.

BER-TLV (*BASIC ENCODING RULES-TAG, LENGTH, VALUE*)

Objetos de dados BER são codificados de acordo com a linguagem ASN.1 definida pela norma ISO/IEC 8825-1.

Cada objeto de dados BER-TLV consiste em dois ou três campos consecutivos:

- ***Tag***: campo obrigatório que consiste de um ou mais bytes consecutivos. Indica a classe, o tipo de codificação e um número. O valor '00' é inválido para o primeiro byte do campo tag;

- ***Length***: campo obrigatório de um ou mais bytes que codifica o tamanho, ou seja, um número denotado N;

- ***Value***: campo que contém o valor armazenado. É opcional porque quando N é zero, não deve existir; quando N > 0 deve conter N *bytes* consecutivos;

A Figura 31 resume os parâmetros para cada um dos três campos *Tag*, *Length* e *Value* na codificação de objetos BER-TLV.

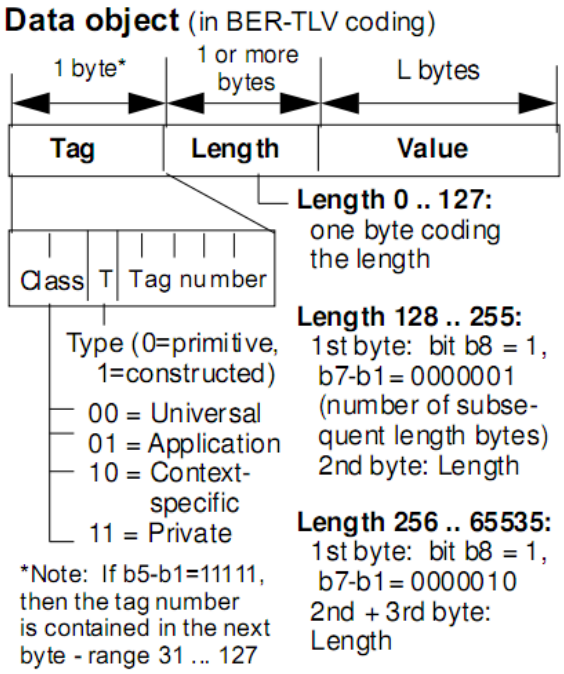

Figura 31 - Codificação do campo BER-TLV

Fonte: http://www.panstruga.de/ct-api/spec/mctp5v09.pdf. Acessado em 25/04/2011

6.5.3 - ESTRUTURA PARA APLICAÇÕES E DADOS

A memória é estruturada logicamente em um sistema de arquivos no qual existem três tipos de arquivos: *Dedicated Files* (DFs), *Elementary Files* (EFs) e *Master File* (MF). O

Master File (MF) é o arquivo raiz do sistema de arquivos; *Dedicated Files* são arquivos, análogos aos diretórios, pois podem conter DF's e EF's; EF's são arquivos que contém dados.

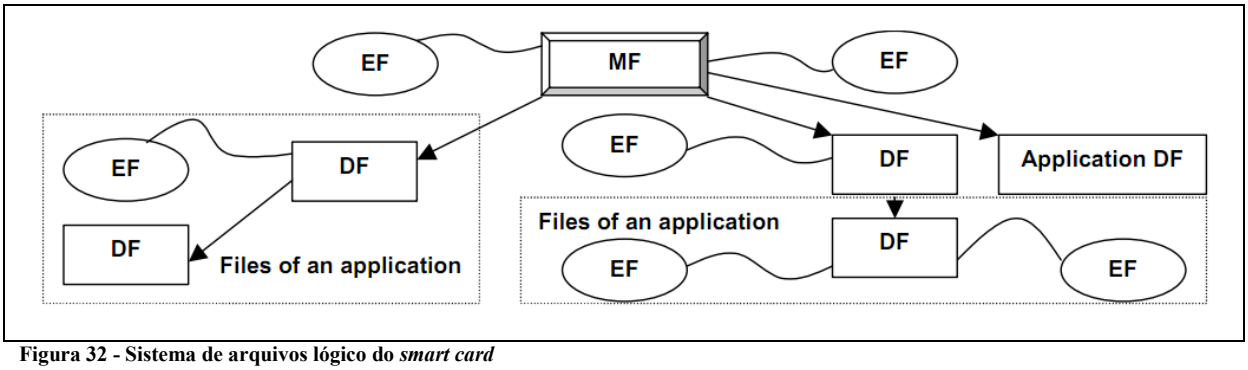

Figura 32 - Sistema de arquivos lógico do *smart card*
Fonte: (ISO/IEC 1786-4, 2005)

Existem dois tipos de EF's, os internos (*internal EF's*) e os de trabalho (*working EF's*). Os internos são utilizados pelo próprio cartão para armazenar dados internos, por exemplo, estruturas de controle. Os de trabalho são EF's disponíveis para o mundo exterior, e não são interpretados pelo cartão. (PIGNATARO, 2006)

Existe o conceito de canal lógico, que estabelece uma ligação com um DF, definindo aquele DF como o "DF atual" ou "DF selecionado". Um canal lógico pode ter uma ligação adicional a um EF do DF selecionado, definindo-o como o "EF atual" ou "EF selecionado". Após o reset do cartão, cria-se implicitamente um canal lógico no qual o MF está selecionado. Podem existir múltiplos canais lógicos, porém, para fins de simplificação, consideraremos apenas um canal lógico.

Arquivos devem poder ser referenciados por pelo menos um dos seguintes métodos:

- **Referência por ID de arquivo**: cada arquivo possui um ID, codificado em 2 bytes, e um ID referencia um arquivo sob o DF atual. Nesse esquema, todos os EF's e DF's filhos de um mesmo DF devem ter ID's diferentes para que possam ser referenciados sem ambiguidade. Os valores 3F00, 3FFF e FFFF são reservados (o primeiro é o ID do MF, o segundo é utilizado na referência por caminho e o terceiro é reservado para uso futuro);

- **Referência por caminho (*path*)**: um caminho começa com o ID do MF ou do DF atual, e acaba com o ID do arquivo referenciado, tendo os IDs dos DFs intermediários no meio. Caso o ID do DF atual não seja conhecido, o valor 3FFF pode ser usado (assim como o "." em *prompts* de commando convencionais);

- **Referência de *EF short identifier***: um short identifier consiste em um número de 1 a 30 que referencia um EF dentro do DF atual. O número 0 referencia o EF atual. A norma especifica que se esse tipo de referência for suportado pelo cartão, ele deve ser o tipo adotado;

- **Referência de DF por nome**: cada DF pode ser referenciado por um nome, uma string de 1 a 16 *bytes*. Nesse esquema, para que cada DF possa ser referenciado sem ambiguidade, cada DF do cartão deve ter um nome único.

A estrutura dos EF's podem se classificar em:

- **Estrutura transparente**: o EF é visto como uma sequência única e contínua de unidades de dados acessível por comandos de tratamento de unidade de dados;

- **Estrutura de registro**: o EF é visto como uma sequência única e contínua de registros individualmente identificáveis acessível por comandos de tratamento de registros. O tamanho dos registros pode ser fixo ou variável. A organização dos registros pode ser uma sequência linear de registros (fixos ou variáveis) ou um anel (estrutura cíclica) de registros de tamanho fixo;

- **Estrutura TLV**: o EF é visto como um conjunto de objeto de dados acessível por comando de tratamento de objeto de dados. Objetos de dados em uma estrutura TVL podem ser SIMPLE-TLV ou BER-TLV.

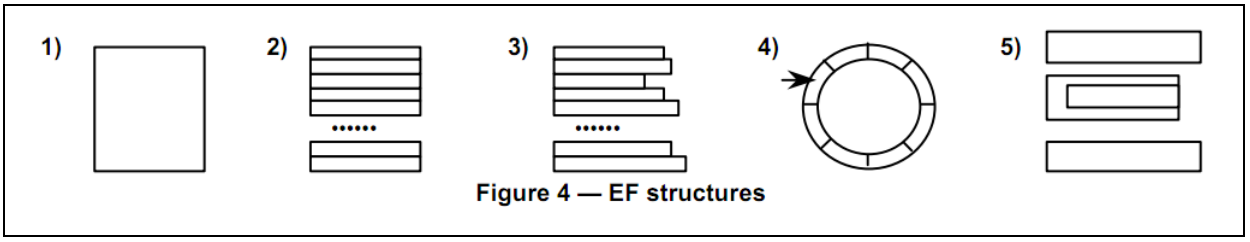

Figure 4 — EF structures

Figura 33 - Estruturas de EFs

Fonte: (ISO/IEC 1786-4, 2005)

As estruturas de EF's são modeladas na Figura 33 onde 1, 2, 3, 4 e 5 correspondem respectivamente a Estrutura transparente, Estrutura linear com registros de tamanho fixo, Estrutura linear com registros de tamanho variável, Estrutura cíclica com registros de tamanho fixo e estrutura TLV. Para referenciamento de dados em EFs, o cartão deve suportar pelo menos uma das cinco das estruturas.

6.5.4 - ARQUITETURA DE SEGURANÇA

Esta seção da norma descreve status, atributos e mecanismos de segurança.

O status de segurança representa um estado alcançado após a execução de uma ação que pode ser um comando *Answer To Reset*, a seleção de parâmetros de um possível protocolo/comando ou uma sequência de comandos, possivelmente realizados em procedimentos de autenticação.

A alteração de um status de segurança pode resultar da realização de um procedimento de segurança relacionado com a identificação das entidades envolvidas, por exemplo, após a execução dos comandos GET CHALLENGE seguido por EXTERNAL AUTENTICATE, etc.

Quatro status de segurança são considerados:

- **Status de segurança global**: Em um cartão, utilizando a hierarquia de DF's, pode ser alterado após a conclusão de um procedimento de autenticação relacionado a um MF;

- **Status de segurança específica de aplicação**: pode ser alterado após a conclusão de um procedimento de autenticação relacionado à aplicação (por exemplo, autenticação de uma entidade por uma senha ou uma chave anexada à aplicação); pode ser mantido, recuperado ou perdido pela aplicação selecionada; essa modificação pode ser relevante apenas para o aplicativo ao qual pertence o procedimento de autenticação. Se os canais lógicos são aplicáveis, então o status de segurança específico da aplicação pode depender do canal lógico;

- **Status de segurança específica de arquivos**: pode ser alterado após a conclusão de um procedimento de autenticação relacionado a um DF (exemplo: autenticação de uma entidade por uma senha ligada a um DF específico); pode ser mantido, recuperado ou perdido pela seleção de arquivos; essa mudança pode ser relevante apenas para a aplicação ao qual pertence o procedimento de autenticação.

- **Segurança específica de comandos**: existe apenas durante o processamento de um comando usando mensagens seguras e envolvendo autenticação; tal comando pode deixar o status de segurança de outros comandos inalterada.

Os atributos de segurança, quando existentes, definem as ações permitidas e sob quais condições as ações são permitidas. Atributos de segurança de arquivos dependem de sua categoria (DF ou EF) e de parâmetros opcionais existentes em suas informações de controle e/ou existentes em arquivos dos quais herdou. Atributos de segurança também podem ser associados a comandos, objetos de dados e tabelas e visões.

Em particular, os atributos de segurança podem especificar o estado de segurança do cartão em vigor antes de acessar dados, restringir o acesso aos dados de determinadas funções (por exemplo, somente leitura) se o cartão tiver um estatuto especial e definir quais as funções de segurança devem ser realizados para obter um status de segurança específico.

Os mecanismos de segurança incluem:

- **Autenticação de entidade com senha**: o cartão compara os dados recebidos do mundo exterior com dados secretos internos. Este mecanismo pode ser usado para proteger os direitos do usuário;

- **Autenticação de entidade com chave**: a entidade a ser autenticada tem que provar o conhecimento da chave secreta ou privada em um procedimento de autenticação (por exemplo, um comando GET CHALLENGE seguido por um comando externo AUTHENTICATE);

- **Autenticação de dados**: usando dados internos, quer uma chave secreta ou uma chave pública, o cartão verifica os dados redundantes recebidos do mundo exterior.

Como alternativa, utilizando uma chave secreta/privada, o cartão calcula um elemento de dados (sumário de mensagem ou assinatura digital) e insere nos dados enviados para o mundo exterior;

- **Criptografia de dados**: usando uma chave secreta/privada, armazenada internamente, o cartão decifra um criptograma recebido em um campo de dados. Como alternativa, utilizando uma chave secreta ou uma chave pública armazenada internamente, o cartão calcula um criptograma e insere em um campo de dados, possivelmente junto com outros dados.

O resultado de uma autenticação pode ser registrado em um EF interno de acordo com os requisitos das aplicações.

6.5.5 - COMANDOS DE COMUNICAÇÃO

A norma ISO/IEC 7816 prevê vários comandos para lidar com ICC's. A Tabela 12 contém todos os comandos previstos em (ISO/IEC 1786-4, 2005) ordenados pelo nome do comando. Além do nome do comando, é descrito o *byte* de instrução INS (campo do cabeçalho APDU) e o local onde o comando é detalhado, podendo ser outro documento que compõe a norma (ex.: ACTIVATE FILE é descrito em 1786-9) ou em uma seção da norma 7816-4.

Esta seção irá detalhar o comando SELECT para que seja possível entender como os comandos são descritos na norma.

Tabela 12 - Lista de comandos em ordem alfabética
Fonte: (ISO/IEC 1786-4, 2005)

Nome	INS	Seção
ACTIVATE FILE	'44'	Part 9
APPEND RECORD	'E2'	7.3.7
CHANGE REFERENCE DATA	'24'	7.5.7
CREATE FILE	'E0'	Part 9
DEACTIVATE FILE	'04'	Part 9
DELETE FILE	'E4'	Part 9
DISABLE VERIFICATION REQUIREMENT	'26'	7.5.9
ENABLE VERIFICATION REQUIREMENT	'28'	7.5.8
ENVELOPE	'C2', 'C3'	7.6.2

ERASE BINARY	'0E', '0F'	7.2.7
ERASE RECORD (S)	'0C'	7.3.8
EXTERNAL (/ MUTUAL) AUTHENTICATE	'82'	7.5.4
GENERAL AUTHENTICATE	'86', '87'	7.5.5
GENERATE ASYMMETRIC KEY PAIR	'46'	Part 8
GET CHALLENGE	'84'	7.5.3
GET DATA	'CA', 'CB'	7.4.2
GET RESPONSE	'C0'	7.6.1
INTERNAL AUTHENTICATE	'88'	7.5.2
MANAGE CHANNEL	'70'	7.1.2
MANAGE SECURITY ENVIRONMENT	'22'	7.5.11
PERFORM SCQL OPERATION	'10'	Part 7
PERFORM SECURITY OPERATION	'2A'	Part 8
PERFORM TRANSACTION OPERATION	'12'	Part 7
PERFORM USER OPERATION	'14'	Part 7
PUT DATA	'DA', 'DB'	7.4.3
READ BINARY	'B0', 'B1'	7.2.3
READ RECORD (S)	'B2', 'B3'	7.3.3
RESET RETRY COUNTER	'2C'	7.5.10
SEARCH BINARY	'A0', 'A1'	7.2.6
SEARCH RECORD	'A2'	7.3.7
SELECT	'A4'	7.1.1
TERMINATE CARD USAGE	'FE'	Part 9
TERMINATE DF	'E6'	Part 9
TERMINATE EF	'E8'	Part 9
UPDATE BINARY	'D6', 'D7'	7.2.5
UPDATE RECORD	'DC', 'DD'	7.3.5
VERIFY	'20', '21'	7.5.6
WRITE BINARY	'D0', 'D1'	7.2.4
WRITE RECORD	'D2'	7.3.4

COMANDO SELECT

Após o ANSWER TO RESET, normalmente, a MF ou um DF é implicitamente selecionado através do canal de lógico.

O comando SELECT cria um ponteiro lógico para um arquivo específico no sistema de arquivos do ICC. Esse ponteiro é necessário para qualquer operação de manipulação de arquivo. Os parâmetros 1 e 2 do comando SELECT são descritos nas tabelas 14 e 15.

O acesso ao sistema de arquivos do *smart card* não é *multithread*, porém é possível ter vários ponteiros de arquivo definidos em qualquer ponto no tempo. MANAGE CHANNEL (INS ‘70’) estabelece vários canais lógicos entre o aplicativo situado no lado representado

pelo leitor de *smart card* (IFD) e o cartão, permitindo que diferentes arquivos no cartão possam estar, simultaneamente, em vários estados de acesso no aplicativo leitor.

Tabela 13 - Parâmetro P1 aplicado ao comando SELECT
Fonte: (ISO/IEC 1786-4, 2005)

b8	b7	b6	b5	b4	b3	b2	b1	Significado	Campo Comando
0	0	0	0	0	0	x	X	**Seleção pelo identificador**	
0	0	0	0	0	0	0	0	-selecionar MF, DF ou EF	**ID do arquivo ou nulo ou ausente**
0	0	0	0	0	0	0	1	-selecionar DF filho (do DF atual)	**ID do DF**
0	0	0	0	0	0	1	0	-selecionar EF filho do DF atual	**ID do EF**
0	0	0	0	0	0	1	1	-selecionar o DF pai do DF atual	**Ausente**
0	0	0	0	0	1	x	X	**Selecionar pelo nome do DF**	
0	0	0	0	0	1	0	0	-selecionar nome DF	**Nome DF**
0	0	0	0	1	0	x	X	**Selecionar pelo caminho**	
0	0	0	0	1	0	0	0	-Selecionar a partir da raiz	**Caminho com ID do MF**
0	0	0	0	1	0	1	1	-Selecionar a partir do DF atual	**Caminho com ID do DF**

A resposta de um comando SELECT retorna uma *string* de *bytes* denominada *File Control Information* (FCI) que pode estar disponível tanto para EF's quanto para DF's ou outras estruturas. Se o primeiro *byte* estiver compreendido entre '00' e 'BF' significa que o byte está codificado segundo a codificação BER-TLV, caso contrário não está codificado de acordo com a norma (ISO/IEC 1786-4, 2005). Existem três *templates* para a informação de controle de arquivos codificada como objetos BER-TLV:

- **FCP**: contém um conjunto de parâmetros para controle dos arquivos que inclui, por exemplo, tamanho do arquivo incluindo o metadados, tamanho do arquivo desconsiderado o metadados, ID do arquivo e outros;

- **FMD**: conjunto de dados de gerenciamento de arquivos, isto é, objetos de dados interindustriais como um identificador de aplicação, data de expiração da aplicação e outros;

- **FCI**: conjunto de parâmetros para controle de arquivos e dados de gerenciamento de arquivos, em outras palavras, inclui os dois anteriores.

O *byte* correspondente ao parâmetro P2 da APDU do comando SELECT especifica qual o formato (ou *template*) de FCI (FCP, FMD ou FCI) que o cartão deve retornar além de especificar o comportamento no caso de a referência ao arquivo ser considerada ambígua. A Tabela 14 especifica os possíveis valores do parâmetro P2 e os respectivos significados.

Tabela 14 - Parâmetro P2 aplicado ao comando SELECT
Fonte: (ISO/IEC 1786-4, 2005)

b8	b7	b6	b5	b4	b3	b2	b1	Significado
0	0	0	0	–	–	x	x	**Ocorrência do arquivo**
0	0	0	0	–	–	0	0	**-primeira ou única ocorrência**
0	0	0	0	–	–	0	1	**-última ocorrência**
0	0	0	0	–	–	1	0	**-próxima ocorrência**
0	0	0	0	–	–	1	1	**-ocorrência anterior**
0	0	0	0	x	x	–	–	**Informação de controle de arquivo**
0	0	0	0	0	0	–	–	**-retorna FCI template, uso opcional da tag FCI e length**
0	0	0	0	0	1	–	–	**-retorna FCP template, uso obrigatório da tag FCP e length**
0	0	0	0	1	0	–	–	**-retorna FMP template, uso obrigatório da tag FMD e length**
0	0	0	0	1	1	–	–	**-sem resposta caso campo L_e esteja ausente ou formato proprietário no caso contrário**

6.6 – 7816-5 PARTE 5: PROCEDIMENTOS DE REGISTRO PARA PROVEDORES DE APLICAÇÕES

A parte 5 da norma 7816 estabelece um padrão – um nome universal para aplicações de *smart card* chamadas Identificador de Aplicação (AID – *Application Identifier*). Um AID possui duas partes:

- ***Registered Application Provider Identifier* (RID)**: um número de tamanho fixo de 5 bytes que é único para cada provedor de aplicação;

- ***Proprietary Application Identifier Extension* (PIX)**: identificador de tamanho variável utilizado pelo desenvolvedor de aplicação para identificar de forma única as suas aplicações.

A concatenação do RID + PIX deverá fornecer uma identificação universal e única de uma determinada aplicação.

A norma também especifica como realizar o registro internacional dos identificadores únicos de aplicação a partir da explanação do processo de registro, das autoridades responsáveis por realizar o registro e outros.

6.7 – 7816-6: DADOS PARA INTERCÂMBIO INTERINDÚSTRIA

A parte 6 da norma 7816 especifica diretamente ou por referência DE's (*Data Elements*), incluindo DE's compostos, utilizado na transferência interindustrial. São identificadas as seguintes características de cada DE: identificador, nome, descrição e referência ISO, formato e codificação.

A intenção da parte 6 da norma é que cada objeto de dados utilizado seja descrito pela mesma, evitando que diferentes fornecedores armazenem coisas iguais utilizando modos diferentes. Devido a esta característica, existem diversos DE's descritos pela norma, entre os quais é possível citar endereço do indivíduo, data de expiração do cartão, código do país, código da moeda utilizada, data de aniversários, sexo, política de uso de PIN e outros.

6.8 – 7816-8: COMANDOS PARA OPERAÇÕES SEGURAS

Especifica comandos para cartões de circuito integrado (seja com ou sem contatos de contatos) que podem ser utilizados para operações criptográficas. Estes comandos são complementares e baseados em comandos listados na ISO/IEC 7816-4.

Os comandos descritos na parte 8 da norma envolvem operações relacionadas a assinaturas digitais, certificados digitais, importação e exportação de chaves assimétricas e criptografia. Não é obrigatório que cartões compatíveis com esta parte da norma suportem todos os comandos ou todas as opções suportadas pelo comando. Entre os comandos descritos na norma 7816-8 estão:

- GENERATE PUBLIC KEY PAIR
- PERFORM SECURITY OPERATION

- COMPUTE CRYPTOGRAPHIC CHECKSUM
- COMPUTE DIGITAL SIGNATURE
- HASH
- VERIFY CRYPTOGRAPHIC CHECKSUM
- VERIFY DIGITAL SIGNATURE
- VERIFY CERTIFICATE
- ENCIPHER
- DECIPHER

6.10 – 7816-9: COMANDOS PARA O CARTÃO E GERÊNCIA DE ARQUIVOS

Esta seção da norma especifica comandos para a gestão de cartões e de arquivo. Estes comandos irão cobrir todo o ciclo de vida do cartão e, portanto, alguns comandos podem ser usados antes que o cartão tenha sido emitido para o titular do cartão ou depois que o cartão expirou.

Os comandos responsáveis por gerenciar o ciclo de vida dos cartões e do arquivo são

- **CREATE FILE**: inicia a criação de um arquivo (DF ou EF) que será colocado imediatamente abaixo do DF atual;
- **ACTIVATE FILE**: inicia a transição de um estado a partir de qualquer arquivo de criação do estado ou o estado de inicialização ou o estado operacional (desativado) para o estado operacional (ativado);

- **TERMINATE EF**: inicia uma transição irreversível que leva o EF especificado para o estado de terminação. Para que o comando seja realizado, a EF deve estar no estado de ACTIVE ou DEACTIVATED;

- **DELETE FILE**: inicia a exclusão de uma referência EF imediatamente abaixo do DF atual, ou de uma DF com a sua completa sub árvore. Após a conclusão bem sucedida deste comando, o arquivo que foi excluído não pode mais ser selecionado;

- **DEACTIVATE FILE**: inicia a desativação reversível de um arquivo. Após a conclusão bem sucedida do comando, além do comando SELECT, somente o ACTIVATE FILE, DELETE FILE, TERMINATE EF e, no caso de uma DF, o comando TERMINATE DF é permitido;

- **TERMINATE DF**: inicia uma transição irreversível que leva o DF especificado para o estado de terminação. Após a conclusão bem sucedida do comando, o DF está em um estado encerrado e as funcionalidades disponíveis a partir do DF e dos DF's hierarquicamente inferiores são reduzidas;

- **TERMINATE CARD USAGE:** inicia uma transação irreversível que leva o cartão ao estado de terminação. O uso deste comando significa uma seleção implícita do MF.

A Figura 34 contém os estados que os arquivos (EF's e DF's) podem assumir bem como as possíveis transições.

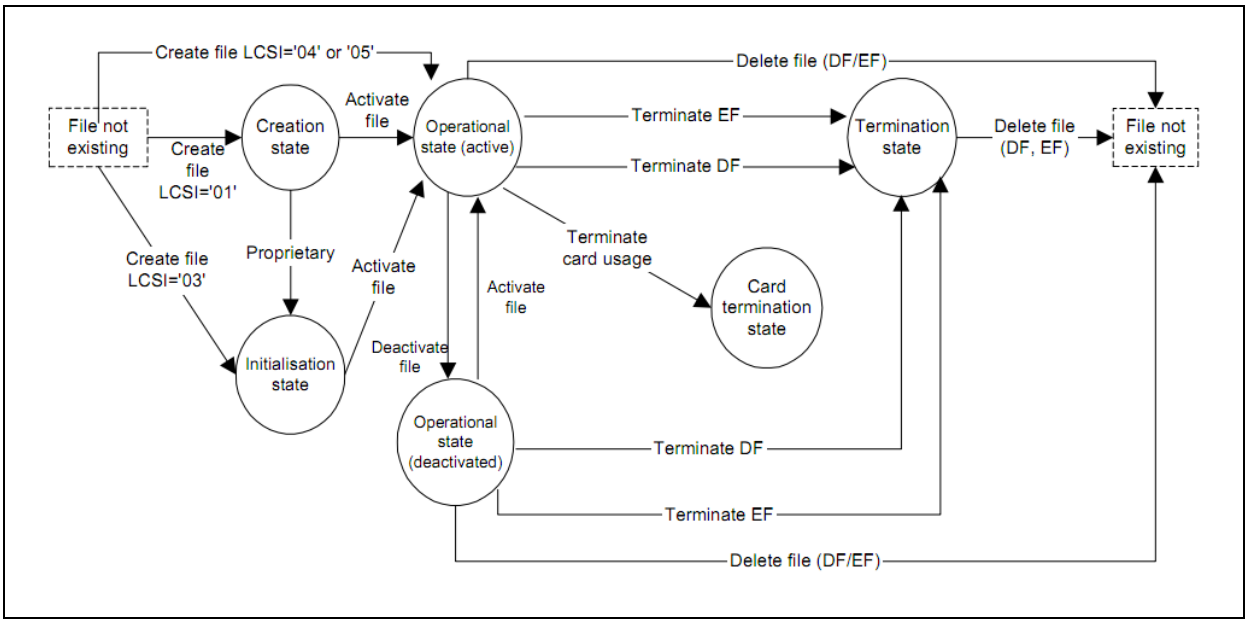

Figura 34 - Diagrama do ciclo de vida de um arquivo
Fonte: (ISO/IEC 7816-9, 2004)

6.11 – 7816-15: APLICAÇÃO DE INFORMAÇÕES CRIPTOGRÁFICAS

O objetivo da parte 15 da norma ISO / IEC 7816 é fornecer um arcabouço para identificação segura dos usuários de sistemas de informação, bem como para outros serviços de segurança essenciais como não repúdio, assinaturas digitais e distribuição de chaves de cifragem de confidencialidade.

Esta parte da NBR ISO / IEC 7816 é baseada no padrão PKCS#15 descrito em (RSA SECURITY INC, 2000). A relação entre a norma ISO/IEC 7816 e PKCS#15 é a seguinte: (ISO/IEC 7816-15, 2004)

- Um núcleo comum é idêntico em ambos os documentos;
- Os componentes do PKCS#15 que não se relacionam com cartões IC foram removidos;
- Esta parte da ISO/IEC 7816 inclui melhorias para atender às necessidades específicas do cartão do CI.

Esta seção irá tratar do núcleo comum compartilhado entre o PKCS#15 e a norma 7816-15, baseado no padrão PKCS#15.

O padrão especifica uma estrutura lógica de arquivos/objetos. Nessa estrutura são definidos quatro tipos de objetos: Chaves, Certificados, objetos de autenticação e objetos de dados. Todas as classes desses objetos possuem subclasses; por exemplo, uma classe Chave contém como subclasses as classes Chave Privada e Chave Pública e a instâncias dessas classes são os objetos armazenados no cartão. A Figura 35 descreve esse cenário.

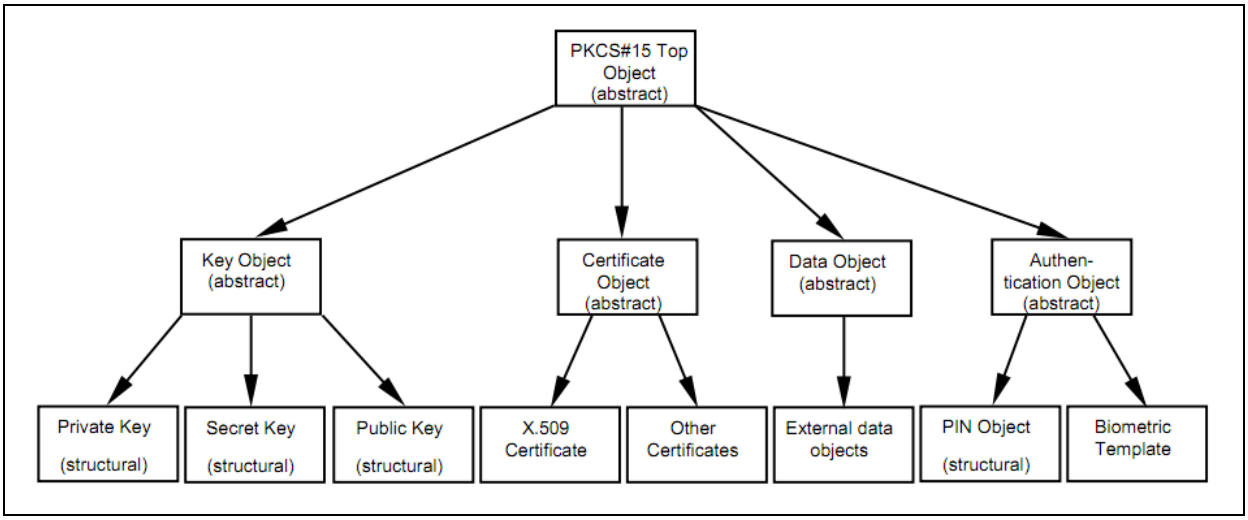

Figura 35 - Estrutura lógica de objetos do padrão PKCS#15
Fonte: (RSA SECURITY INC, 2000)

Os objetos que irão instanciar as classes descritas na Figura 35 são EF's serão armazenados em um mesmo DF – o DF(PKCS #15). O DF PKCS#15 pode ser selecionado a partir do nome A0:00:00:00:63 || "PKCS-15" (PIGNATARO, 2006).

Os objetos podem ser privados (protegidos contra o acesso não autorizado) ou públicos. As restrições relativas a um objeto privado (ler, escrever, etc) são definidas por objetos de autenticação (que inclui também procedimentos de autenticação).

A estrutura típica de arquivos armazenados na DF PKCS#15 contempla os seguintes arquivos elementares EF's, cuja estrutura é transparente:

- **ODF - *Object Directory File***: objeto obrigatório de diretório de arquivos (ODF) é um ficheiro elementar, que contém ponteiros para outras EF's (PrKDFs, PuKDFs,

SKDFs, CDF, DODFs e AODFs), cada um contendo um diretório sobre PKCS# 15 objeto de uma classe particular;

- **PrKDFs – *Private Key Directory Files***: objeto que contém informações sobre as chaves privadas, como rótulos, uso pretendido, etc. Possui um ID denominado "auth ID", que é uma referência cruzada para o ID do PIN que autoriza o uso da chave. Existe também um identificador denominado apenas "ID", que permite relacionar a chave privada à correspondente chave pública e também, possivelmente, a um certificado. O "*keyReference*" (referência de chave) é um *byte* que identifica a chave privada apenas dentro do contexto do próprio cartão. Este *byte* é informado no comando MANAGE SECURITY ENVIRONMENT para especificar qual chave utilizar (PIGNATARO, 2006);

- **CDFs - *Certificate Directory Files***: contém referências para os EF's onde estão armazenados os certificados. Estas referências são obrigatórias, mas o CDF pode opcionalmente conter os certificados. Entretanto, geralmente eles não contêm o certificado completo, pois isso constitui duplicação de informação no cartão, ou seja, desperdício de espaço, já que os certificados obrigatoriamente estão presentes completos em outros EF's. Contém também o ID que permite fazer referência cruzada com um par de chaves associado ao certificado;

- **PuKDFs - *Public Key Directory Files***: pelo menos um objeto deve existir quando o ICC contém chaves públicas ou referências a chaves públicas. Esses arquivos podem ser considerados como diretórios de chaves públicas conhecidas da aplicação PKCS#15;

- **SKDFs - *Secret Key Directory Files***: pelo menos um objeto deve existir quando o ICC contiver chaves privadas ou referências a chaves privadas. Contêm atributos gerais como chave, *labels*, intenção de uso, identificadores, etc. Quando é o caso, eles contêm referência cruzada com ponteiros para objetos de autenticação usados para proteger o acesso às chaves;

- **AOD -** ***Authentication Objects***: podem ser considerados diretórios de objetos de autenticação (ex: PIN's, dados biométricos, chaves de autenticação). Pelo menos um OADF deve estar presente no cartão quando este contem objetos de autenticação para restringir o acesso a objetos PKCS#15 restritos. Eles contêm atributos de objeto de autenticação genéricos, tais como (no caso dos PIN's) caracteres permitidos, tamanho do PIN e outros. Além disso, contêm ponteiros para os objetos de autenticação si (por exemplo, ponteiros para o DF no qual o arquivo reside o PIN).

6.12 – ETAPAS DA ASSINATURA DE UM DOCUMENTO

Uma forma comum de gerar documentos assinados é descrita na Figura 36. Conforme é possível perceber na figura, inicialmente é gerado um sumário de mensagem (*hash*) do arquivo que se deseja assinar, utilizando um algoritmo de *hash* (ex.: SHA-1, SHA-2, etc). Posteriormente, o *hash* é enviado para o *smart card* para que este possa assinar o *hash* e devolver à aplicação que está gerenciando o processo. A partir do momento que a aplicação tem o *hash,* assinado ela gera um arquivo que segue um padrão de assinatura de documento (ex.: padrão PKCS#7).

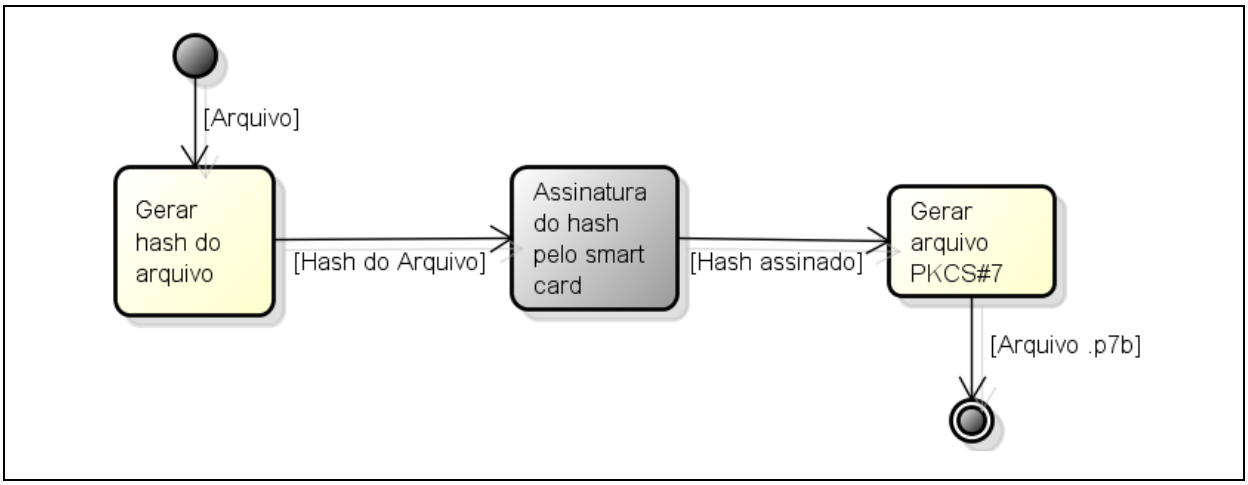

Figura 36 - Passos para a assinatura de um documento

Fonte: Elaborado pelo autor (2011)

A atividade de "Assinatura do *hash* pelo *smart card*" envolve o envio de uma série de comandos APDU's que estão descritos ao longo das diversas partes da norma ISO/IEC 7816 e

PKCS. É necessário destacar que nem todos os cartões inteligentes suportam todos os recursos da norma. Isso não significa que o ICC é incompatível com a norma, uma vez que a norma muitas vezes especifica requisitos mínimos, como é o caso relacionado às possíveis formas de se referenciar a arquivos (ISO/IEC 7816-4) em que a norma especifica que arquivos podem ser referenciados por pelos menos um dentre três métodos possíveis (*ID*, *path* e *EF short identifier*), ou seja, permitir apenas uma das três formas significa conformidade com o padrão.

Dessa forma, os comandos utilizados devem ser direcionados pela norma, mas é necessária a verificação da compatibilidade no manual do fabricante do *smart card* utilizado. O exemplo utilizado neste tópico, extraído de (PIGNATARO, 2006), enumera os passos necessários à realização de uma assinatura digital de um sumário de mensagem previamente calculado.

As respostas retornadas pelo *smart card* podem estar codificadas nos padrões SIMPLE-TLV, BER-TLV ou DER-TLV (especialização do BER-TLV). Em todos os casos é necessário o entendimento da norma que especifica cada um dos padrões para que se possa realizar a interpretação correta da resposta e obter os valores corretos e desejáveis.

A assinatura de um conteúdo qualquer utilizando o *smart card* envolve as fases passos: Buscar certificados no cartão e Assinar o documento que são decompostas em mais de um passo. Para tornar mais claro o entendimento, o comando APDU enviado para o ICC, no primeiro passo de cada uma das fases, será detalhado.

6.12.1 - BUSCAR OS CERTIFICADOS NO CARTÃO

Para assinar digitalmente um documento, é preciso inicialmente recuperar o certificado do assinante, bem como os certificados que compõem sua cadeia de validação, e validar o certificado. Essas ações decompõem-se nos passos:

1. **Acessar o DF do PKCS#15:** DF que é o “diretório raiz” do padrão PKCS#15. Deve-se selecionar (comando SELECT) o DF para que posteriormente seja possível acessar o arquivo ODF.

2. **Acessar o ODF e ler o ODF:** conforme visto anteriormente, o ODF contém um ponteiro para outros arquivos, incluindo os arquivos que serão utilizados posteriormente CDF (*Certificate Directory Files*) e PrKDF (*Private Key Directory File*). Dessa forma, é necessário aplicar um comando de seleção (SELECT) sobre o ODF, posteriormente um comando de leitura (READ BINARY) e realizar um *parser* da resposta para que seja possível obter o identificador do CDF e do PrKDF.

3. **Acessar e ler o CDF:** uma vez que o identificador do CDF for conhecido é necessário selecionar o CDF (comando SELECT), ler o seu conteúdo (READ BINARY) e realizar um *parser* na resposta para que se possa obter o identificador dos arquivos de certificados. Um dos certificados deve ser escolhido para realizar a assinatura.

4. **Acessar e ler os certificados:** normalmente é disponibilizada ao usuário a opção de selecionar um dos certificados a partir de informações contidas no certificado, como por exemplo, o nome da pessoa para qual o certificado foi emitido (*Subject Name*). A forma de se obter as informações contidas no certificado é selecionando (comando SELECT), lendo o certificado (comando READ) e realizando um *parser* para interpretar os dados armazenados.

5. **Verificar a validade do certificado**: após o usuário escolher um certificado para realizar assinatura, a aplicação deve verificar a validade do certificado, isto é, certificados expirados não podem ser utilizados. Certificados que foram revogados também não são considerados válidos. Esta etapa não envolve comunicação com o ICC.

6. **Acessar e ler os dados da chave privada que vai assinar do documento**: quando um certificado contém uma chave pública, cuja chave privada reside também no cartão, o certificado e a chave privada devem partilhar o mesmo identificador (RSA, 2010). O identificador da chave privada (igual ao identificador do certificado) foi obtido no passo 4. As informações sobre chaves privadas estão armazenadas no EF (PrKDF) cujo identificador pôde ser obtido no passo 2. Dessa

forma, é necessário acessar o EF (PrKDF) (comando SELECT), ler o conteúdo do PrKDF (comando READ) e realizar um *parser* para obter as seguintes informações sobre as chaves privadas armazenadas:

a. *KeyReference*: byte que identifica a chave privada apenas dentro do contexto do próprio cartão. Este byte é informado no comando MANAGE SECURITY ENVIRONMENT para especificar qual chave utilizar.

b. *AuthId*: contém o identificador do objeto de autenticação (PIN – *Personal Identification Number*)

Para acessar o DF do PKCS#15 deve-se enviar ao *smart* card a APDU **00:A4:04:0C:0C:A0:00:00:00:63:50:4B:43:53:2D:31:35**, cujos campos possuem os valores que seguem:

- **Class (CLA)**: 0x00 (Descrito na Tabela 11)
- **Instruction (INS)**: 0xA4 (comando SELECT, ISO/IEC 7816-4 – Tabela 13)
- **P1**: 0x04 ou 0b00000100 (selecionar DF pelo nome - Tabela 14)
- **P2**: 0x0C ou 0b00001100 (não retornar informações de controle de arquivo - Tabela 15)
- $\mathbf{L_c}$: 0x0C ou 0b00001100 (*command data field* ou Nc = 12 bytes)
- **Data:** = A0 00 00 00 63 50 4B 43 53 2D 31 35 (FULL ID especificado em PKCS#15)
- $\mathbf{L_e}$: ausente (numero máximo de bytes de resposta – Ne - é 0)

6.12.2 - ASSINAR O DOCUMENTO:

a. **Informar o PIN**: o PIN é utilizado para autenticação de um objeto. Nesta etapa é necessário autenticar, ou seja, o *smart card* deve verificar a autenticidade do usuário (comando VERIFY). Nesta etapa deve ser informado o *AuthId* e o PIN do usuário.

b. Criar um contexto para execução de segurança: uma vez autenticado através do PIN, deve-se iniciar um contexto de segurança (comando MANAGE SECURITY ENVIRONMENT) informando o identificador da chave privada (*Keyreference*) que irá executar as opções de segurança.

c. **Informar o *hash* do documento a ser assinado**: enviar ao *smart card* o *hash* a ser assinado (comando PERFORM SECURITY OPERATION, subcomando HASH/PUT HASH).
d. **Calcular a assinatura**: comando PERFORM SECURITY OPERATION, subcomando COMPUTE SIGNATURE.

2. O APDU anterior possui os seguintes valores para seus campos:

O procedimento de informar o PIN é realizado enviando-se ao *smart card* a APDU **00:20:00:(AuthID):08:(PID)**, cujos campos possuem os valores que seguem:

- **Class**: 0x00 (Descrito na Tabela 11)
- **Instruction**: 0x20 (comando VERIFY, ISO 7816-4)
- **P1**: 0x00 (valor fixado pelo padrão)
- **P2**: Auth ID, obtido do PrKDFs no passo 6.
- **Lc**: 0x08
- **Data**: PIN do usuário, em ASCII. Caso o PIN possua menos de 8 bytes deve ser completado com zeros à direito até atingir os 8 bytes. Por exemplo, se o PIN é 1234, então Data = 31:32:33:34:00:00:00:00, pois o código ASCII dos números 1, 2, 3 e 4 são, respectivamente, 31, 32, 33 e 34.

Caso o PIN informado esteja errado, o comando retorna SW1:SW2 = 63:CX, onde X é número de tentativas restantes antes que o cartão seja bloqueado.

CAPÍTULO 7
- DESENVOLVIMENTO DO PROJETO

O SCREAD MOD, objeto deste trabalho, é um leitor portátil de cartões inteligentes que se comunica através da tecnologia *Bluetooth.* Tal dispositivo deve ser capaz de intermediar uma comunicação entre um cartão inteligente homologado pela ICP-Brasil (por exemplo, o cartão e-CPF que segue o padrão ISO/IEC 7816) e um dispositivo computacional compatível com a tecnologia *Bluetooth*. A Figura 37 contém um esboço do resultado final do SCREAD MOD acoplado a um *smart phone.*

Figura 37 - Esboço do produto SCREAD MOD
Fonte: elaborado pelo autor (2011)

Além do desenvolvimento do dispositivo de *hardware* SCREAD MOD, este trabalho de mestrado também inclui o desenvolvimento de um *software* aplicativo denominado DS SCREAD (*Digital Signer for SCREAD MOD*) e de uma API (*Application Programming Interface*) denominada SCREAD MOD API. Ambos os sistemas deverão ser executados em um dispositivo móvel (um *smart phone* com Android) e devem trabalhar integrados de modo a tornar possível a assinatura de documentos utilizando a chave privada armazenada no cartão inteligente. As instruções devem ser enviadas pela API para o cartão inteligente através da comunicação sem fio com o SCREAD MOD; tais instruções enviadas estão no nível de

APDU (*Application Data Unit*) e seguem os padrões ISO/IEC 7816-4, ISO/IEC 7816-15 e o padrão PKCS#15.

Este capítulo tem como objetivo apresentar o circuito eletrônico do protótipo do *hardware* SCREAD MOD, os *softwares* desenvolvidos quer permitiram validar o protótipo e também a análise que permitiu decidir sobre as tecnologias utilizadas. Deve-se ressaltar que as decisões técnicas relacionadas ao desenvolvimento do SCREAD MOD são influenciadas pelas características tecnológicas disponíveis atualmente nos *smart phones* e mantêm como foco a alta mobilidade presente nos dispositivos móveis. Este capítulo está divido em quatro partes: Decisões de projeto, Projeto do circuito elétrico, Programas desenvolvidos e Integração entre os componentes de *hardware* e *software.*

7.1 - DECISÕES DE PROJETO

As decisões de projeto objetivam responder às perguntas: qual sistema operacional deve ser suportado pelo *smart phone*? A comunicação realizada entre o celular e o SCREAD MOD será por um meio cabeado ou sem fio? O dispositivo deverá se limitar a ser utilizado apenas pelo celular ao qual foi acoplado? Qual o software que vai permitir a interação entre o *smart phone* e o SCREAD MOD?

A resposta a cada uma das perguntas anteriores será respondida nos tópicos que seguem. Após a justificativa das escolhas tecnológicas do projeto, será apresentado um protótipo operacional do SCREAD MOD.

SISTEMA OPERACIONAL DO SMART PHONE

Atualmente existem diversos sistemas operacionais difundidos que executam em celulares, entre os quais, destacam-se *Symbian* (mantido pela Nokia[15]), *Android* (desenvolvido pelo Google[16]), *iOs* (desenvolvido pela Apple[17]), *Black Berry* (desenvolvido pela RIM[18]) e

15Mais informações sobre Symbian: http://symbian.nokia.com/
16Mais informações sobre Android: http://www.android.com/
17Mais informações sobre iOS (iPhone OS): http://www.apple.com/ios

Windows Mobile (desenvolvido pela Microsoft[19]). Critérios de escolha para o sistema operacional para o celular envolvem diversas características tecnológicas, a saber: consumo de energia, desempenho, sistema proprietário x sistema livre, custo dos *smart phones*, participação de mercado etc.

As duas plataformas líderes de mercado para *smart phones* - *iPhone*/*iOS* e *Android* - foram pré-selecionadas e os alguns critérios foram utilizados com base nas características dessas duas plataformas. Antes de analisar os critérios é necessário entender os desafios presentes no desenvolvimento de aplicações para dispositivos móveis que não estão presentes no desenvolvimento de aplicações para computadores pessoais (desktops ou notebooks).

Dispositivos móveis impõem algumas considerações relacionadas ao *hardware* que devem ser levadas em consideração no projeto. Meier (2009) destaca que comparado com desktops ou notebooks, os dispositivos móveis apresentam relativamente baixo poder de processamento, RAM limitada, limitada capacidade de armazenamento permanente, telas pequenas, conexões menos confiáveis, bateria como fator limitante da utilização e outros. Além das características de hardware o ambiente do usuário traz seus próprios desafios, pois dispositivos móveis são muitas vezes utilizados como uma distração ao invés do foco de atenção, dessa forma os aplicativos precisam ser rápidos, ágeis e fáceis de usar.

Além dos fatores listados em (MEIER, 2009) há um outro fator a considerar que é a inviabilidade de expansão ou *upgrade* de um dispositivo móvel, dessa forma, se um celular não possui a tecnologia *Bluetooth* ou possui um módulo *Bluetooth* incompatível com um determinado perfil desejado, não será possível utilizar essa tecnologia, diferentemente de um computador pessoal que basta acoplar um adaptador *Bluetooth* USB que é facilmente encontrado e custa aproximadamente apenas U$ 4,00.

As diversas configurações de hardware existentes em celulares unidas às restrições impostas pelas características inerentes a dispositivos móveis resultam em alguns questionamentos/problemas ao desenvolvedor. Por exemplo, um determinado algoritmo que

18Mais informações sobre Black Berry: http://en.wikipedia.org/wiki/BlackBerry_OS
19Mais informações sobre Windows Mobile: http://www.microsoft.com/windowsmobile/pt-br/default.mspx

roda em um modelo será capaz de executar em outro modelo que possui um processador mais lento de forma a manter uma experiência de usuário aceitável? A usabilidade deve ser projetada a atender apenas dispositivos que suportem *touch screen*, teclados ou ambos? O fato de funcionar em um dispositivo móvel que oferece capacidade de processamento superior é suficiente para garantir que funcionará em um dispositivo com capacidade móvel de processamento inferior?

A possibilidade de existência de diversas configurações de hardware para um mesmo sistema operacional de celular será referenciada como variabilidade de hardware. Quanto mais configurações de hardware forem disponíveis para um mesmo SO maior a variabilidade de hardware e, de forma contrária, quanto menos possibilidades de configurações, menor a variabilidade de *hardware*. A alta variabilidade de *hardware* pode ser considerado um fator positivo ou um fator negativo, conforme será visto no comparativo entre os SO's iOS e *Android* a seguir:

- **Baixa variabilidade de hardware (fator positivo para *iOS*)**: os celulares que utilizam o iOS são os celulares denominados iPhone desenvolvidos pela Apple. Atualmente, existem apenas quatro modelos (Original iPhone, iPhone 3G, iPhone 3GS e iPhone 4) que executam o iOS resultando em uma baixa variabilidade no hardware para esse SO. Isso é um fator positivo porque é possível saber a priori exatamente qual o tipo de hardware e sistema operacional que o dispositivo deverá se integrar ou uma aplicação deverá executar. Isso torna mais fácil o desenvolvimento e diminui as preocupações do desenvolvedor com diferentes tipos de dispositivos. De forma contrária, existem diversas versões de sistema operacional *Android* que rodam em diversos tipos de hardware; a responsabilidade sobre a atualização do SO normalmente é dividida entre a operadora de telefonia e o fabricante do celular de modo que existem modelos com hardware equivalentes rodando com diferentes versões de SO tornando o desenvolvimento e os testes mais complexos.

- **Alta variabilidade de hardware (fator positivo para Android)**: o fator alta variabilidade de hardware que anteriormente foi considerado um fator negativo,

sob uma nova óptica pode ser considerado um fator positivo, uma vez que limitações físicas que possivelmente seriam encontradas em um conjunto restrito de celulares poderiam ser contornadas com *Android* escolhendo um conjunto de aparelhos que atendessem às características físicas necessárias. Por exemplo, aparelhos *iPhone* não são compatíveis com módulos *Bluetooth* facilmente encontrados no Brasil e não suportam a maioria dos perfis que são implementadas por módulos Bluetooth vendidos no exterior.

- **Preço (fator positivo Android)**: o iPhone é um *smart phone* que se enquadra entre os mais caros ao lado de *smart phones* que utilizam Android como Samsung Galaxy S, Sony Ericson XPeria X10 e outros modelos considerados top's de linha. Contudo, utilizando-se a plataforma *Android* é possível escolher um *smart phone* que atenda aos requisitos técnicos e não seja tão caro quanto é um modelo *top*. Ainda como fator negativo para o *iOS*/*iPhone* a Apple cobra uma taxa anual para que um desenvolvedor possa criar aplicações para *iPhone* sem precisar recorrer a técnicas que podem resultar na perda de garantia do aparelho.

- **Plataforma de desenvolvimento (fator positivo *Android*)**: a SDK (*Software Development Kit*) oficial do Google possibilita que aplicações para *Android* sejam desenvolvidas em computadores com sistemas operacionais Linux, Windows ou MacOS ao passo que a plataforma de desenvolvimento oficial da Apple somente possui uma versão para MacOS.

Devido aos argumentos expostos, foi escolhida como plataforma de desenvolvimento do estudo de caso a plataforma *Android*. Vale ressaltar que a documentação de ambas as plataformas são muito completas e não foi encontrado nenhum motivo que justifique a escolha de um ou outro com relação a esse critério.

TECNOLOGIA DE COMUNICAÇÃO

O SCREAD MOD é um leitor de cartões inteligentes que tem como diferencial o fato de se comunicar via *Bluetooth*, poder funcionar acoplado a um *smart phone e* interagir com este. Uma comunicação genérica entre um *smart phone* e outro dispositivo pode ser realizada

a partir de um meio cabeado (ex.: USB) ou a partir de uma comunicação sem fio (ex.: WiFi). Este tópico descreve a análise realizada para a escolha da forma de comunicação entre um *smart phone* e o SCREAD MOD.

COMUNICAÇÃO CABEADA

A maioria dos *smart phones* possui uma interface USB, de modo que a comunicação USB entre o SCREAD MOD e o celular seria uma alternativa interessante. Contudo, essa escolha levaria a alguns problemas técnicos.

Os dispositivos USB, conforme visto anteriormente se comportam como clientes (ou escravos) ou se comportam como hosts (também chamados de servidores ou mestres). A tecnologia OTG (*On-The-Go*) possibilita que um determinado dispositivo se comporte em como mestre em alguns momentos e, em outros momentos, como escravo, mas com algumas poucas limitações. Todos os modelos de *smart phones* pesquisados têm apenas a capacidade de funcionar apenas como cliente, isto é, não implementam a característica USB OTG. Dessa forma, a comunicação entre o SCREAD MOD e um *smart phone* via USB torna necessário a implementação da característica *Host* USB no SCREAD MOD. Isso leva a alguns problemas:

- **Microcontrolador com suporte a USB-*host***: é necessário implementar todas as funcionalidades de um host-USB em um microcontrolador ou então utilizar algum microcontrolador ou outro CI (Circuito Integrado) que forneça esta funcionalidade de forma nativa. Alguns exemplos de CI que fornecem essa funcionalidade são a família 24F (16 bits) e 32F (32 bits) de microcontroladores da Microchip ou o CI VNC1L da FTDI. Esses microcontroladores são mais caros e consomem mais energia que as famílias de microcontroladores do mesmo e/ou de outros fabricantes. Ainda que os fatores custo/energia fossem desconsiderados o projeto eletrônico se tornaria mais complexo e demandaria mais tempo para desenvolvimento.

- **Plug USB variado**: muitos *smart phones* são compatíveis com a porta USB, contudo existem pelo menos quatro conectores que são utilizados, são eles os conectores A-Mini, B-Mini, A-Micro (ou AB-Micro) e B-Micro. Seria necessário

escolher apenas um conector e isso tornaria o SCREAD MOD difícil de ser acoplado a diferentes tipos de smart phones. Adicionalmente, existem muitos celulares que possuem um conector proprietário e necessitam de um cabo especial para conectar à porta USB do computador.

- **Indisponibilidade da porta USB**: atualmente *smart phones* ficam muito tempo conectados à porta USB do computador por dois motivos: os *smart phones* consomem mais energia e descarregam mais rapidamente e, a porta USB do computador normalmente é utilizada para carregar *smart phones*. Adicionalmente, os *smart phones* funcionam como discos rígidos portáteis, pois alguns possuem capacidade de armazenamento de até 64GB. Então, é comum usuários de *smart phones* utilizarem o celular ligado a uma porta USB de um computador e realizar as duas tarefas simultaneamente. Manter o SCREAD MOD ligado à porta USB do celular inteligente envolveria tornar o acoplamento do celular ao SCREAD MOD algo facilmente desacoplável e isso incorreriam em restrições mecânicas que provavelmente aumentariam o tamanho do SCREAD MOD indo de encontro à premissa de que deve ser o menor possível para acompanhar a mobilidade do celular. Outra solução seria fazer o SCREAD MOD funcionar como um *host* para o celular e um cliente para o computador e intermediar a comunicação; contudo, essa possibilidade tornaria a construção do dispositivo mais complexa, tanto em termos de hardware quanto em termos de software, pois além de *hardware* mais complexos ainda o SCREAD MOD deveria funcionar aos olhos do computador como o próprio dispositivo o que envolveria o desenvolvimento de drivers específicos para cada celular inteligente que funcionasse acoplado.

A grande vantagem desta última abordagem seria a possibilidade de recarregar tanto o *smart phone* quanto o SCREAD MOD simultaneamente. Contudo, esta vantagem identificada foi considerada pequena em relação às desvantagens apresentadas e, isso levou, ao estudo da possibilidade de se acoplar o *smart phone* ao SCREAD MOD através de uma tecnologia sem fio.

COMUNICAÇÃO SEM FIO

As tecnologias de acesso ao meio sem fio IrDA, *Bluetooth* e WiFi (802.11) foram analisadas e comparadas no Capítulo 4 – Tecnologias de Acesso ao meio sem fio. Pode-se perceber que a característica típica da comunicação existente entre o SCREAD MOD e um *smart phone* é a característica de uma comunicação ponto-a-ponto. Foi visto anteriormente que a tecnologia ideal para essa aplicação é a tecnologia infravermelho por ser mais barato e consumir menos energia. Contudo, alguns critérios foram utilizados para se decidir pela tecnologia *Bluetooth*, conforme será descrito a seguir.

- **Tecnologia IrDa está ficando em desuso em *smart phones*:** foi realizada uma pesquisa em um dos maiores sites comparativos de celulares do mundo, o GSMArena. A pesquisa envolveu identificar a quantidade de celulares ativos com cada uma das tecnologias de comunicação sem fio: *Bluetooth*, IrDA e 802.11. O site não disponibilizava a opção de buscar dispositivos com a tecnologia NFC. Foi realizado um cruzamento entre os celulares ativos e os celulares que possuem telas *touch screen*. A quantidade de celulares ativos é um forte indicador para identificar as tecnologias que os fabricantes estão investindo; como a popularização das telas *touch screen* são relativamente recentes, a quantidade de celulares com uma determinada tecnologia sem fio e que suporte também *touch screen* é um forte indicador de tendência de investimento por parte do fabricante. O fato de não haver a opção de filtro para dispositivos como NFC é um forte indicativo de que a tecnologia ainda não está sendo comercializada ou é comercializada em um nível ainda irrelevante, fato corroborado por uma pesquisa realizada - sem suporte do site GSMArena - que identificou apenas oito *smart phones* ativos compatíveis com a tecnologia NFC (Anexo B). A Figura 38 resume a pesquisa realizada no GSMArena.

- **Tecnologia Bluetooth está presente na maioria dos celulares e consome pouca energia:** conforme pode se perceber na Figura 38, a tecnologia *Bluetooth* está presente na maioria dos *smart phones* incluindo aqueles provavelmente mais antigos (sem *touch screen*) e os provavelmente mais recentes (com *touch screen*). Adicionalmente, no tópico 4.5 - Comparativo das tecnologias sem fio foi

identificado que a tecnologia *Bluetooth* (especialmente classe 3) consome pouquíssima energia comparado às outras tecnologias.

- **Tecnologia não fica restrita a uma comunicação ponto-a-ponto entre o celular e o SCREAD MOD:** utilizando uma comunicação Bluetooth, o SCREAD MOD pode ser utilizado por um computador enquanto que outras tarefas são realizadas pelo celular inteligente. Por exemplo, o SCREAD MOD pode ser utilizado por um desktop via *Bluetooth* enquanto que o celular transfere arquivos via 802.11 ou USB, por exemplo. Os únicos dois leitores de cartões inteligentes sem fio encontrados (BlackBerry Smart Card Reader e OMNIKEY 2061 Bluetooth Reader) utilizam a tecnologia *Bluetooth*.

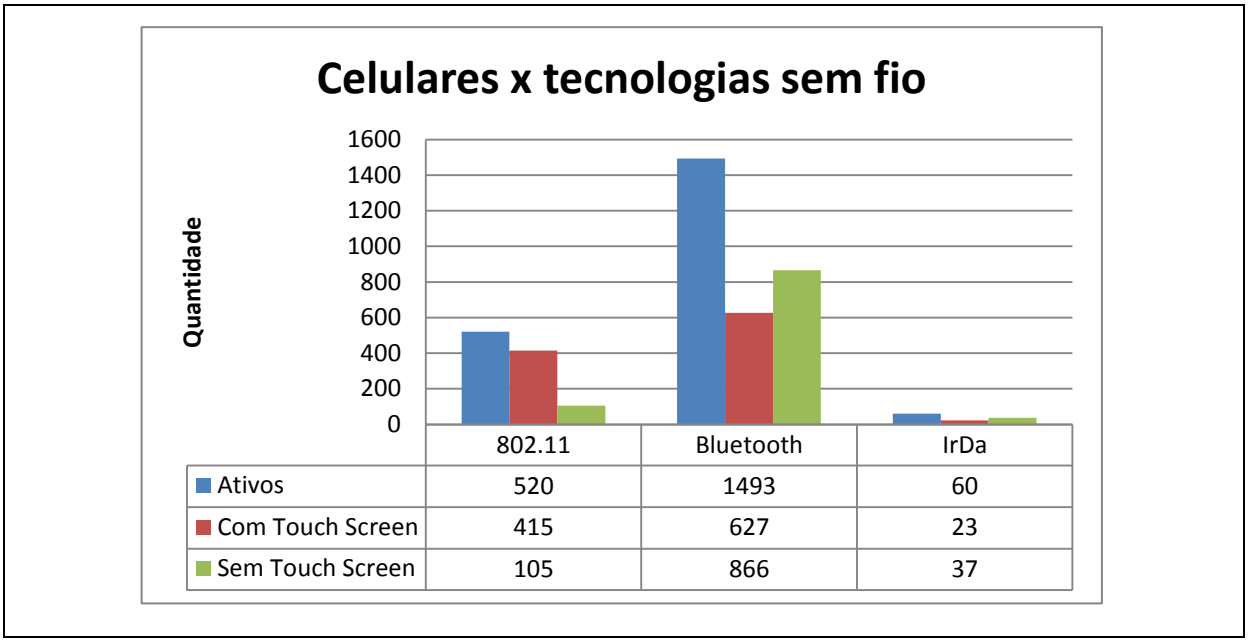

Figura 38 – Comparativo entre quantidade de celulares e tecnologias de comunicação sem fio
Fonte: GSMArena. www.gsmarena.com, acessado em 09/02/2011.

COMPATIBILIDADE *BLUETOOTH SMART PHONE* X MÓDULO *BLUETOOTH*

A tecnologia *Bluetooth* foi escolhida como meio de comunicação, contudo ainda restam algumas decisões a respeito desta tecnologia. Foi visto no tópico 4.2 - Bluetooth (IEEE 802.15) que a tecnologia *Bluetooth* permite a utilização de vários tipos de perfil. Deve-se utilizar um módulo *Bluetooth* que seja compatível com o perfil utilizado pelo *smart phone*.

Por exemplo, não é possível se comunicar com um módulo Bluetooth que utilize o perfil PAN (*Personal Area Network*) se o *smart phone* for compatível com esse tipo de perfil.

A escolha do tipo de perfil a ser utilizado foi uma escolha difícil, porque a maioria dos fabricantes de *smart phones*, não disponibiliza todos os tipos de perfil que o *smart phone* é compatível. Normalmente o fabricante apenas especifica a versão do *Bluetooh* (ex.: Bluetooth 2.1); contudo isso não quer dizer que o dispositivo suporta todos os tipos de perfil que são encontrados na especificação da versão *Bluetooth* discriminada.

A especificação *Bluetooth* especifica que um determinado perfil A é pré-requisito para a existência de um perfil B, mas mesmo assim é possível que haja dispositivos compatíveis com o perfil B sem que o fabricante libere a possibilidade de utilizar o perfil A (esse é o caso do iPhone). A Figura 39 contém os tipos de perfil Bluetooth que as diversas gerações do iPhone é compatível. O Anexo A contém um estudo de compatibilidade *Bluetooth* de *smart phones* extraído de fontes diversas como sites comparativos, fabricantes, fóruns e outros.

Device	Hands-Free Profile (HFP 1.5)	Phone Book Access Profile (PBAP)	Advanced Audio Distribution Profile (A2DP)	Audio/Video Remote Control Profile (AVRCP)[1]	Personal Area Network Profile (PAN)	Human Interface Device Profile (HID)
iPhone 4	√	√	√	√	√	√
iPhone 3GS	√	√	√	√	√	√
iPhone 3G	√	√	√	√	√	-
Original iPhone	√	√	-	-	-	-
iPad	-	-	√	√	√[2]	√
iPod touch (second, third, and 4th generations)	-	-	√	√	√[2]	√

1. iPhone 4, iPhone 3GS, iPhone 3G, iPad, iPod touch (2nd generation), and iPod touch (4th generation) support pause, play, stop, next track and previous track for AVRCP.
2. Supports Bluetooth peer-to-peer connectivity for applications. See this article for additional information. Internet tethering is not supported.

Figura 39 - Profiles Bluetooth suportados pelos IPhone

Fonte: http://support.apple.com/kb/ht3647. Acessado em 18/12/2010.

Adicionalmente, é necessário encontrar um módulo que possa ser integrado a um microcontrolador e que seja compatível com o perfil Bluetooth do *smart phone*. Alguns módulos mais sofisticados e caros, como por exemplo, os módulos Bluetooth módulos KC-21 e KC-22 (fabricante KC Wirefree) fornecem diversos tipos de perfil, mas não foram

encontrados à venda no Brasil. Isso não foi um problema real, porque os módulos encontrados no Brasil, como por exemplo, o módulo AUBTM-22 (fabricante Austar Technology), são compatíveis com o perfil SPP e o perfil SPP é compatível com muitos *smart phones* que utilizam *Android*. Por esse motivo o perfil utilizado na comunicação é o perfil SPP.

DISPOSITIVOS UTILIZADOS

MICROCONTROLADOR PIC18F4550

O PIC18F8550 é um microcontrolador que pertence à família de microcontroladores da família 18F desenvolvidos pela empresa Microchip Technology.

A arquitetura utilizada pelo microcontrolador é a arquitetura *Harvard*. A diferença mais significativa entre a arquitetura *Von Neumann* e a arquitetura *Harvard* é que esta possui uma memória para dados e outra para programas enquanto que aquela possui uma única memória utilizada para ambos os fins.

A utilização de memórias diferentes para dados e programas possibilita um maior desempenho, uma vez que a unidade de busca de instruções e dados pode ser executada de forma paralela. De forma contrária, quando os dados e programas são armazenados na mesma unidade de memória, ambos são acessados através do mesmo barramento e isso implica que o acesso aos dados e aos programas pode entrar em conflito levando a atrasos indesejados conhecidos como o gargalo da arquitetura Von Neumann (GRIDLING e WEISS, 2007).

O PIC18F4550 é um dispositivo de 8 bits que possui instruções RISC (*Reduced Instruction Set Computer*) e conta com a tecnologia de PIPELINE. As principais características do microcontrolador podem ser agrupadas em:

- **Memória**: possui três tipos de memória: memória de programa com 32Kb de capacidade de armazenamento, memória RAM com 2Kb de capacidade de armazenamento e uma memória de dados especial do tipo EEPROM (*Electrically-Erasable Programmable Read-Only Memory*) com 256 bytes com capacidade de armazenamento;

- **Alimentação elétrica**: pode ser alimentado com tensões entre 4V e 5,5V;

- **Frequência de operação**: pode operar com frequência de 48Mhz (12 MIPS – milhões de instruções por segundo) quando utilizado um oscilador externo. Também pode funcionar sem oscilador externo, pois possui um oscilador interno que permite operar a uma frequência de até 8Mhz;

- **Diversos modos de Gerenciamento de energia**: Possibilita três categorias de modos de gerenciamentos de energia que definem quais as partes do dispositivo são sincronizadas e, às vezes, a que velocidade: 1-Modo Execução (*Run mode*) , 2-Modo Ocioso (*Idle mode*) e 3-Modo Inativo (*Sleep mode*). No modo de 1-Execução tanto o microcontrolador quanto os dispositivos ficam ligados. O modo 2-Ocioso permite que a CPU seja desligada seletivamente enquanto os periféricos continuam a operar, consumindo normalmente até 5,8μA. No modo 3-Inativo tanto o microcontrolador quanto os dispositivos são desligados consumindo normalmente até 0,1 μA. Eventos externos podem fazer o microcontrolador sair do modo Inativo para outros modos;

- **Recursos para programação**: O dispositivo possibilita a técnica ICSP (*In Circuit Serial Programming*) que possibilita que o microcontrolador seja programado/ depurado no circuito, ou seja, sem a necessidade de ser removido;

- **Formas de comunicação nativas**: contempla nativamente pinos e instruções para as formas de comunicação USART, USB (*Full Speed* e *Low Speed*), SPI, I^2C e outros, facilitando a programação;

- **Outros**: conversor analógico/digital, três interrupções externas e 10 registradores que controlam interrupções internas, 4 temporizadores (*timers*), etc.

A Figura 40 contém o leiaute do PIFC18F4550. O microcontrolador é composto por 40 pinos dentre os quais 35 pinos – todos excluídos os pinos VDD (2x), VSS (2x) e OSC1/CLKI - podem ser configurados como pinos I/O digitais.

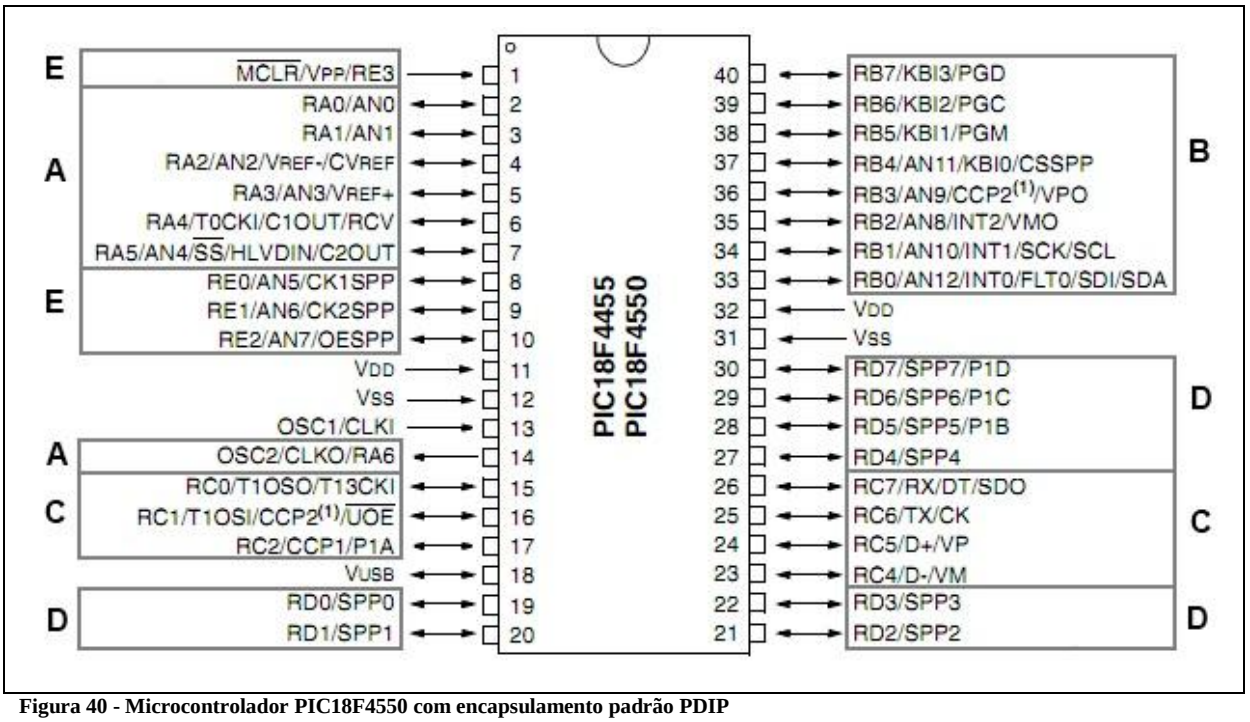

Figura 40 - Microcontrolador PIC18F4550 com encapsulamento padrão PDIP
Fonte: (MICROCHIP TECHNOLOGY, 2009)
Nota: Adaptador pelo autor (2011)

A Figura 40 contém diversos pinos agrupados por letras que são denominados Portas. O PIC18F4550 possui cinco portas disponíveis (A, B, C, D e E). Os pinos associados a elas são multiplexados com diferentes funções de periféricos. Geralmente, quando um determinado periférico é habilitado, o pino relacionado a ele deixa de ser de propósito geral e passa a desempenhar a função que lhe é concedida pelo periférico. Cada porta possui três registradores associados:

- **TRIS**: configura o sentido do fluxo de dados de uma determinada porta;
- **PORT**: escreve/lê os pinos associados;
- **Registrador LAT**: armazena o último comando de escrita.

Além dos papéis de I/O digitais genéricos, alguns pinos podem ser configurados para assumir funcionalidades específicas que são implementadas nativamente pelo microcontrolador. Por exemplo, os pinos 25 (TX) e 26 (RX) podem ser utilizados, respectivamente, para transmissão e recepção assíncrona EUSART.

MÓDULO BLUETOOTH KCWIREFREE KC-21

O módulo *Bluetooth* KC-21 é um circuito desenvolvido pela empresa KCWirefree que se enquadra como um dispositivo Bluetooth classe 2 e, dessa forma, possui limite de alcance teórico de 10 metros.

O KC-21 oferece uma comunicação serial de velocidade máxima de 921 Kbaud (baud - representa o número de mudanças na linha de transmissão, seja em freqüência, ou em amplitude, ou em fase ou em eventos, por segundo). (SILVA, 2009)

Para o funcionamento do KC-21 é necessário uma tensão de 3,3 volts. Ele possui 14 pinos de entrada e saída de propósito geral. Vale destacar ainda que o módulo possui uma memória flash de 8 Mbit (SILVA, 2007). A Figura 41 contém o posicionamento dos pinos e as dimensões físicas do módulo KC-21.

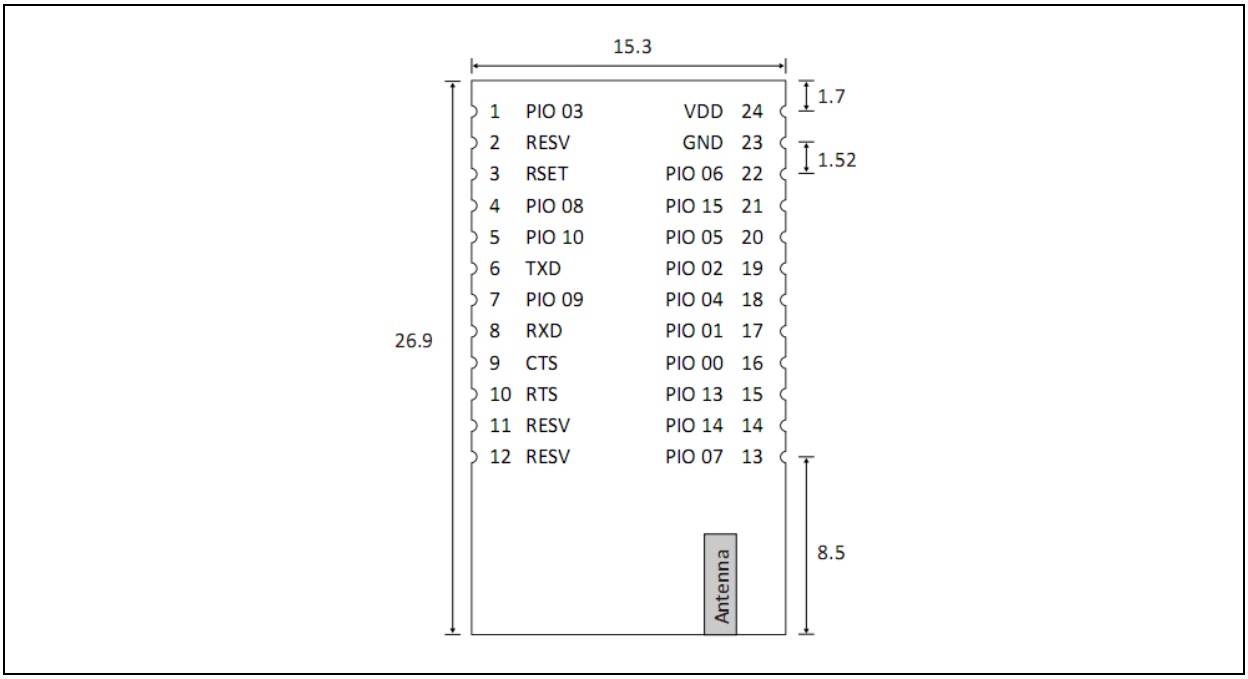

Figura 41 - Módulo Bluetooth KC-21
Fonte: (SILVA, 2009)

CELULAR LG P500 - OPTIMUS ONE WITH GOOGLE

O celular *LG P500 - Optimus One with Google*, ou simplesmente *LG P500*, foi escolhido por ser um *smart phone* compatível com a tecnologia *Bluetooth* perfil SPP (Anexo

A) que utiliza o sistema operacional *Android* e que possui um custo relativamente baixo quando comparado a outros *smart phones* que atendem aos requisitos mínimos para realização deste projeto.

Entre as características relevantes para a utilização do LG P500 destaca-se:

- **Sistema operacional**: *Android Froyo* ou versão 2.2. Atualmente essa é uma das versões de *Android* mais atualizadas disponíveis em celulares comercializados no Brasil.

- **Processador**: Arquitetura ARM com 600Mhz. Ter um processador relativamente rápido é interessante porque o celular terá duas funções principais no processo de assinatura que é calcular o *hash* do arquivo a ser assinado e montar o certificado a partir da assinatura realizada pela *smart card* seguindo o padrão PKCS#7.

- **Touch screen 3.2"**: embora a tela sensível ao toque não seja uma premissa para que seja possível realizar assinaturas digitais, esta característica unida a uma tela relativamente grande (3,2 polegadas) torna a experiência de usabilidade do usuário mais interessante.

- **Outras funcionalidades**: roteador Wi-Fi, GPS, Modem 3G, Câmera 3.2 MP, etc.

AMBIENTE DE DESENVOLVIMENTO

A realização deste trabalho envolveu o desenvolvimento de um programa para ser executado no microcontrolador e um programa para ser executado no celular. Este tópico destaca o ambiente de desenvolvimento utilizado para a programação desses dispositivos e destaca também os programas utilizados para a confecção da placa de circuito impresso e modelagem tridimensional do protótipo conforme segue:

- **Programação para o dispositivo celular**: Foi utilizado o kit de desenvolvimento de software Android SDK (*Software Development Kit*) fabricado pelo Google em conjunto com a IDE Eclipse e o ADT Plugin para eclipse. Para a geração do

envelope assinado foi utilizado uma versão adaptada do *framework* de código aberto conhecido como Bouncy Castle.

- **Programação para o microcontrolador**: foi utilizada a IDE de desenvolvimento MPLab v 8.6 fabricada pela Microchip e o compilador C denominado C18 também desenvolvido pela Microchip e específico para a família de microcontroladores PIC da família 18F. A gravação do foi utilizada um gravador USB genérico.

- **Desenvolvimento do leiaute do circuito**: para a produção do leiaute do circuito impresso foi utilizado o programa Eagle versão 5.6 desenvolvido pela empresa CadSoft.

- **Modelagem tridimensional do protótipo**: para realizar a modelagem tridimensional do protótipo foi utilizado o programa Solid Works 2010 comercializado pela empresa Dassault Systèmes.

- **Outros programas:** para gerenciar os certificados e as chaves armazenadas no cartão inteligente foi utilizado o programa SafeSign Standard desenvolvido pela SafeSign. Para realizar testes enviando APDU's para o cartão inteligente foi utilizado o Smart Card ToolSet Pro versão 3.4.

7.2 – PROJETO DO CIRCUITO ELÉTRICO

Conforme visto anteriormente, a norma ISO/IEC 7816-2 define os oito pinos que um cartão inteligente deve conter. Contudo, apenas a utilização de apenas seis pinos é atualmente especificada pela norma e dois são reservados para uso futuro. Além de identificar os pinos que devem ser utilizados é necessário seguir as dimensões, espaçamentos e identificar a posição que cada um dos pinos se encontra de acordo com as especificações da norma ISO/IEC 7816-2 para que seja possível realizar uma comunicação com um cartão inteligente compatível. Atualmente, são facilmente encontrados conectores de *SIM card* e *sockets* para inserção de cartões inteligentes que são compatíveis com as posições e localizações

estabelecidas na norma ISO/IEC 7816-2. A Figura 42 contém o exemplo de um *socket* de *smart card*.

O *socket* do *smart card* deve ser interligado ao microcontrolador para que este possa gerenciá-lo. Um dos aspectos que torna a ligação não trivial é o fato de que, conforme visto anteriormente, diferentemente de se utilizar uma porta serial RS-232 comum que tem dois fios para dois fios para envio e recebimento de dados funcionando em *full duplex*, os *smart cards* funcionam em *half-duplex*.

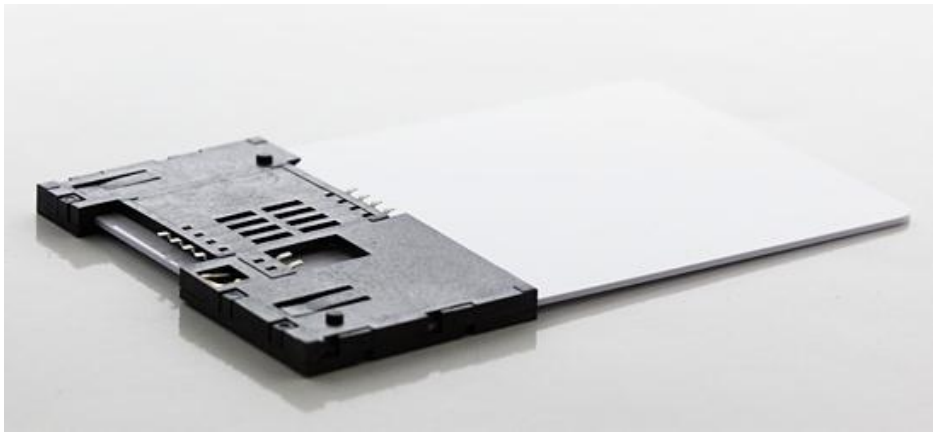

Figura 42 - *Socket* para smart card
Fonte: http://www.sparkfun.com/products/9368. Acessado em 10/01/2011

Diversos circuitos que possibilitam a interligação com *smart cards* podem ser encontrados na *WEB* em sítios especializados em circuitos. No endereço http://www.circuitsarchive.org/, por exemplo, é possível se encontrar exemplos de circuitos de IFD preparados para serem conectados utilizando uma comunicação RS-232 com um computador pessoal e até mesmo circuitos emuladores de *smart cards*.

O circuito do protótipo desenvolvido, representado pela Figura 43, foi derivado do modelo disponibilizado pela Microchip[20] para a utilização de cartões inteligentes compatíveis com a norma ISO/IEC 7816 de forma interligada ao microcontrolador PIC18F14K50 comercializado pela empresa. As mudanças realizadas no protótipo do SCREAD MOD derivaram da necessidade de se adaptar as conexões ao microcontrolador PIC18F4550 e da

[20] Modelo disponibilizado em ww1.microchip.com/downloads/en/AppNotes/01370A.pdf

necessidade de incluir recursos adicionais específicos do SCREAD MOD conforme será visto a seguir.

Objetivando facilitar a programação e depuração do programa que executa no microcontrolador PIC18F4550, o protótipo do SCREAD MOD PIC18F4550 implementou o recurso *Inline Serial Circuit Programming (*ISCP) conforme pode ser visualizado na Figura 43 (utilizando seis pinos). Obviamente este recurso não deve estar disponível em uma versão comercializável do produto possibilitando economizar conexões e espaço do circuito.

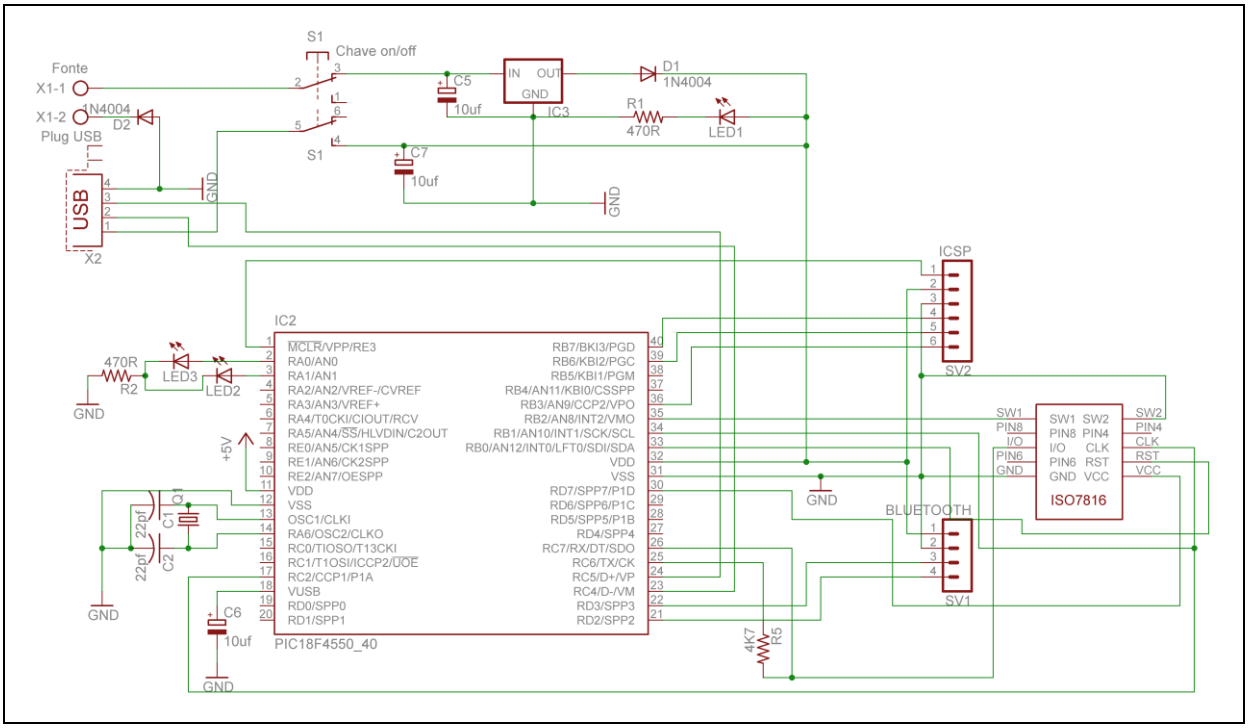

Figura 43 – Circuito do protótipo do SCREAD MOD
Fonte: Elaborado pelo autor (2011)

Outro recurso adicionado no protótipo do SCREAD MOD inclui a utilização de módulo *Bluetooth* para comunicação. A comunicação do módulo *Bluetooth* com o microcontrolador necessitou de quatro contatos elétricos: dois para alimentação elétrica (VCC e GND) e dois para possibilitar uma comunicação UART em *full duplex* (RX e TX). Como os pinos que originalmente compõem a porta UART (RC7/RX e RC6/TX) do microcontrolador estão conectados ao pino I/O do cartão inteligente e o microcontrolador dispõe de apenas uma única implementação de UART por hardware, foi necessário acrescentar ao *software* que

executa no microcontrolador rotinas que executam uma comunicação UART por *software* através dos pinos RD3 e RD2 do microcontrolador.

Através da Figura 43 também é possível perceber que o circuito pode ser alimentado por duas fontes de energia distintas: uma fonte de energia conectada aos pinos X1-1 e X1-2 e uma fonte fornecida por uma conexão USB. Uma chave comutadora *push-button* permite selecionar de forma mutuamente exclusiva qual é a fonte de alimentação utilizada. Caso o circuito seja alimentado através dos pinos X1-1 e X1-2, a fonte deve ser capaz de fornecer pelo menos 7V de tensão, pois há um regulador de tensão modelo LM7805[21] que possui 2V de queda de tensão (*dropout voltage*).

A Figura 44 contém uma foto do protótipo do SCREAD MOD com destaque para os componentes principais: microcontrolador e módulo *Bluetooth*. Ambos os componentes serão detalhados no tópico Dispositivos utilizados.

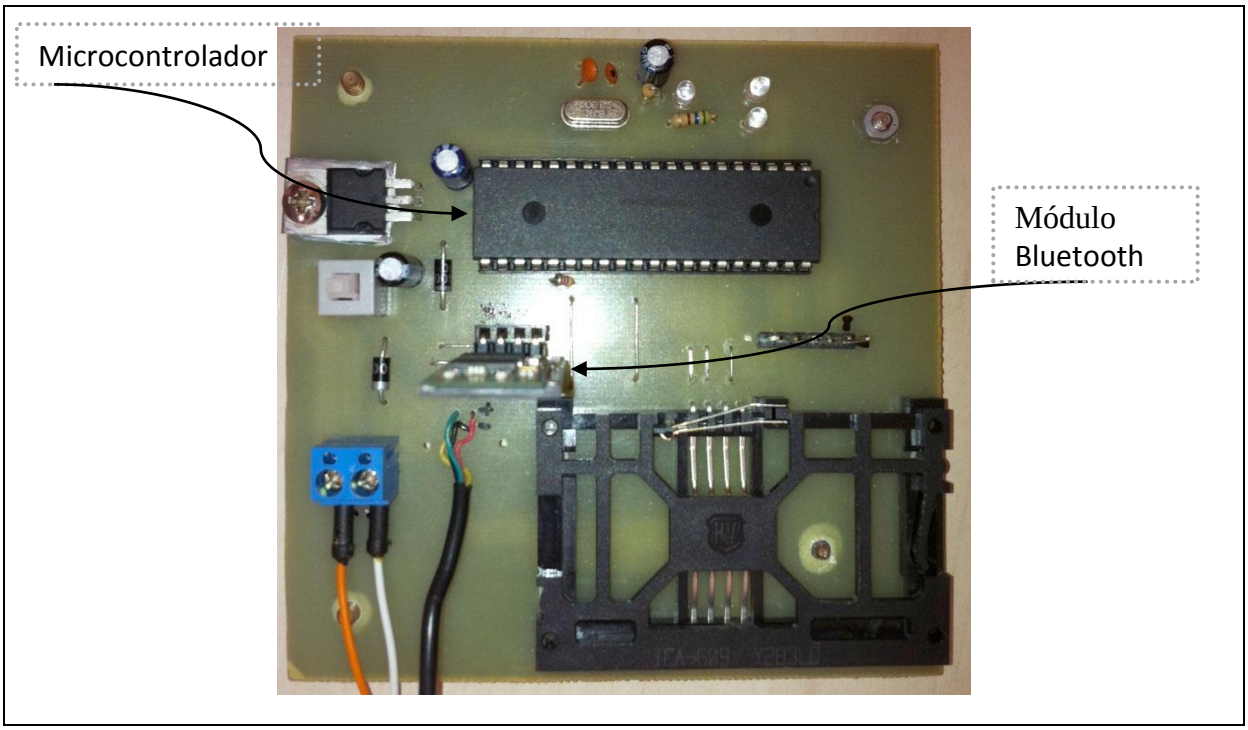

Figura 44 - Protótipo do SCREAD MOD

Fonte: Elaborado pelo autor (2011)

21 Mais informações em http://www.sparkfun.com/datasheets/Components/LM7805.pdf (20/01/2011)

7.3 – PROGRAMAS DESENVOLVIDOS

O projeto desta dissertação envolveu o desenvolvimento de cinco sistemas de *software: software* que executa no microcontrolador, *software* SCREAD MOD API, *software* DS SCREAD e dois *softwares* utilitários que forneceram meios para executar testes de forma mais eficientes ou serviram de suporte ao desenvolvimento dos demais. Estes sistemas desenvolvidos serão descritos a seguir.

SOFTWARE QUE EXECUTA NO MICROCONTROLADOR

Software embarcado que executa no microcontrolador. Foi desenvolvido em linguagem C e compilado pelo compilador C18 da Microchip. O sistema é capaz de inicializar o cartão inteligente; além de algumas operações simples, a inicialização do cartão envolve o envio da instrução de RESET, obtendo a ATR (ISO/IEC 7816-3) e repassando ao dispositivo computacional através do módulo *Bluetooth* quando solicitado. O sistema é capaz de atender a interrupções da inserção/remoção do cartão e informar ao dispositivo computacional conectado pelo módulo *Bluetooth*.

O sistema também recebe através do módulo *Bluetooth* instruções no nível de APDU, repassa tais instruções ao cartão inteligente segundo a norma ISO/IEC 7816-4, obtém a resposta e repassa ao dispositivo computacional através do módulo *Bluetooth*. Conforme mencionado anteriormente, a comunicação com o módulo *Bluetooth* foi realizada através de uma comunicação UART implementada por *software*. Uma característica desejável neste sistema, mas que não foi completamente desenvolvida por não prever mecanismos para troca de chaves envolve o envio/recebimento dos dados de forma criptografada para aumentar a segurança. Um resumo do software que roda no microcontrolador está disponível no Anexo C.

*S*CREAD MOD API

A API (*Application Programming Interface*) é uma biblioteca responsável por se comunicar com o SCREAD MOD, ou mais especificamente, com o módulo Bluetooth KC-21 que está presente no SCREAD MOD. A SCREAD MOD API deve prover primitivas que

possibilitem CONECTAR, DESCONECTAR, ASSINAR e outras funcionalidades específicas que são realizadas pelo cartão inteligente ou são providas pelo SCREAD MOD. A API encapsula instruções ADPU que seguem os padrões/normas ISO/IEC 7816-4, ISO/IEC 7816-15 e PKCS#15 provendo um nível mais alto de abstração para os programas que a utilizem. A comunicação entre uma aplicação qualquer e o SCREAD MOD deve ser realizada através desta API.

ASSINADOR DE DOCUMENTOS (DS SCREAD)

Software que tem como intuito mostrar a operacionalidade do SCREAD MOD. Foi desenvolvido e testado apenas no celular inteligente LG P500. Fornece uma interface visual gráfica que permite ao usuário habilitar/desabilitar o sistema Bluetooth, localizar e conectar/desconectar a dispositivos *Bluetooth* que suportam o protocolo SPP; através das rotinas disponibilizadas na SCREAD MOD API é capaz de identificar se o cartão inteligente está ou não conectado; também calcula e submete o *hash* de um documento armazenado na memória do celular para que o *hash* seja assinado utilizando a chave privada armazenada no cartão. Após obter a resposta do documento assinado este software gera um arquivo assinado (sufixo "p7b") que segue o padrão PKCS#7.

A criação do envelope padrão PKCS#7 foi realizada através da biblioteca de código aberto Bouncy Castle[22]. Para facilitar a verificação dos arquivos gerados pela aplicação DS SCREAD também foi adicionada a possibilidade de envio de arquivos por *e-mail*. As telas que compõem a aplicação são exibidas através da Figura 45. A Figura 45 (a) contém funcionalidades relacionadas a conexões *Bluetooth* (procurar dispositivos, conectar e descontar), Figura 45 (b) contém funcionalidades relacionadas a operações com arquivo (listar, assinar e enviar por *e-mail*) e a Figura 45 (a) contém uma tela com informações relacionadas aos envolvidos (autor e orientadora) no projeto.

[22] Mais informações em http://www.bouncycastle.org .

Figura 45 – Software assinador de documentos DS SCREAD (*Digital Signer* SCREAD)

Fonte: Elaborado pelo autor (2011)

O tópico que segue detalha como ocorre a comunicação entre os componentes de *software* e de *hardware* no processo de assinatura digital de documentos.

UTILITÁRIOS

Conforme citado anteriormente, foram necessário desenvolver dois sistemas para auxiliar no desenvolvimento dos demais:

- **Adaptador Bouncy Castle**: o programa desenvolvido para gerar pacotes PKCS#7 com a biblioteca Bouncy Castle funcionava perfeitamente bem quando compilado e executado no ambiente Windows. Porém, as tentativas de compilação para executar no ambiente Android utilizava consumia muitos recursos processador que o ambiente de desenvolvimento travava o computador em todas as ocasiões. Como o celular e o ambiente operacional destino contemplam um recurso computacional limitado quando comparado a um computador e a biblioteca é muito extensa, foi necessário gerar um programa capaz de eliminar arquivos desnecessários da biblioteca mantendo apenas os arquivos estritamente necessários. Após rodar o sistema conversor foi possível reduzir o tamanho da biblioteca a vinte por cento do tamanho

original e a mesma foi capaz de ser compilada e executada no ambiente destino.

- **Gerador de certificados digitais**: sistema capaz de gerar certificados digitais seguindo o padrão X.509. O desenvolvimento deste aplicativo se justificou pelo fato de que o cartão inteligente utilizado era originalmente um cartão virgem e não havia certificados digitais armazenados. Dessa forma foi criada uma hierarquia de certificados digitais assinados (análogo a uma infraestrutura de chaves públicas) que permitiram criar diversos tipos de certificados e chaves do tipo público/privada de modo a armazená-las no cartão e possibilitar ter exemplos que permitissem a realização de testes reais com o cartão inteligente.

7.4 – INTEGRAÇÃO ENTRE OS COMPONENTES DE *SOFTWARE* E *HARDWARE*

O desenvolvimento e a validação do funcionamento do SCREW MOD envolveu a integração de componentes de *hardware* e *software*. Este tópico descreve como os componentes (de *hardware* e *software)* interagem entre-se sob a óptica do entendimento de funcionamento global do processo de assinatura digital. A Figura 46 contém um diagrama de componentes da *Unified Modeling Language* (UML) que descreve dois componentes físicos: o celular (LG P500) e o leitor de cartões inteligentes (SCREAD MOD).

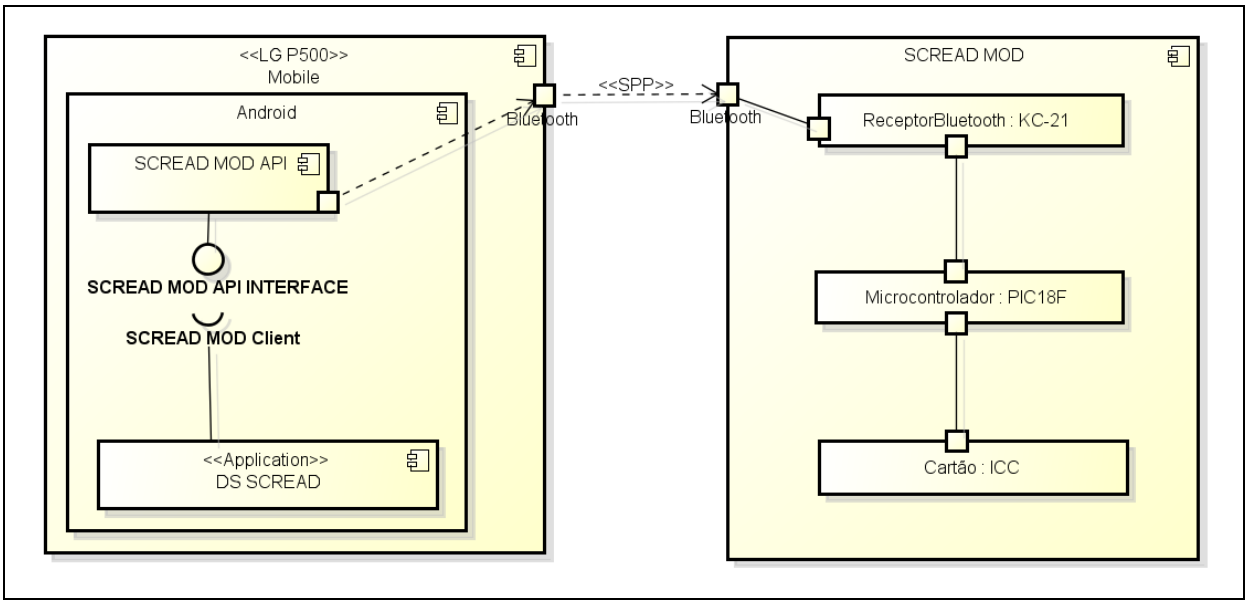

Figura 46 - Integração dos componentes para desenvolvimento da solução
Fonte: elaborado pelo autor (2011)

Conforme a Figura 46, é possível perceber que o celular contém como subcomponente lógico o sistema operacional *Android* que contém como subcomponentes a SCREAD MOD API e uma aplicação que utiliza a API SCREAD MOD para desempenhar a função de assinar documentos digitalmente. No exemplo, a aplicação chama-se *Digital Signer* SCREAD (ou DS SCREAD) que é a aplicação que permite a assinatura de digital de arquivos armazenados no celular descrita no tópico anterior. A figura deixa claro que a API fornece uma interface que é utilizada por um software cliente, que no caso é o DS SCREAD.

O dispositivo SCREAD MOD contém componentes eletrônicos como capacitor, resistor e outros. Contudo apenas o receptor *Bluetooth* (KC-21), o microcontrolador (PIC 18F) e o ICC (*Interface Circuit Card*) estão representados na Figura 46: O SCREAD MOD contém apenas contatos elétricos que se ligam aos contatos elétricos do cartão inteligente (ICC) quando este está conectado; contudo o intuito é descrever a interação dos componentes quando todos estão operacionais e o cartão só está operacional quando inserido no *socket* do SCREAD MOD, dessa forma, o cartão pode ser considerado um componente do leitor de cartões para fins de entendimento.

O papel do módulo KC-21 é aceitar conexões *Bluetooth* SPP de um *smart phone* realizando a troca de chaves e os protocolos de segurança inerentes ao protocolo *Bluetooth*.

As informações recebidas através do módulo são encaminhadas diretamente ao microcontrolador através de uma comunicação UART e as informações recebidas do microcontrolador são repassadas ao dispositivo conectado. Toda comunicação *Bluetooth* está sendo realizada através do perfil SPP.

O microcontrolador tem como papel servir de intermediário entre o módulo KC-21 e o ICC. Como uma comunicação UART tem como velocidade normalmente utilizada 9600 bps e o *Bluetooth* suporta taxas de até 3Mbps um papel importante do microcontrolador é realizar as operações de controle de fluxo e servir como *buffer*. Outras funcionalidades possíveis de serem realizadas com o microcontrolador incluem gerenciar a energia desligando os demais dispositivos quando não estão sendo utilizados e, possibilitar uma comunicação mais segura do que o que o protocolo *Bluetooth* propicia utilizando algoritmos criptográficos para criptografar e decriptografar as mensagens trocadas entre o aparelho celular e o leitor de cartões.

A comunicação UART realizada entre o KC-21 e o microcontrolador foi implementada por software, isso significa que as rotinas de comunicação UART normalmente disponibilizadas pela fabricante não puderam ser utilizadas, uma vez que os pinos reservados para essa tarefa já estavam sendo utilizados em uma comunicação UART entre o microcontrolador e o cartão inteligente.

A Figura 47 contempla os passos mais importantes necessários à realização de uma assinatura digital sob a óptica das atividades que são executadas no celular. Inicialmente o usuário informa um arquivo que deseja assinar (mensagem 1); caso o dispositivo SCREAD MOD não esteja conectado a aplicação vai solicitar à SCREAD MOD API a conexão com o dispositivo SCREAD MOD(mensagens 1.1 e 1.1.1).

Posteriormente a aplicação DS SCREAD solicita à API que busque os certificados armazenados no cartão que pertençam a um determinado sujeito (mensagem 1.2) e a API envia os comandos APDU correspondentes para realizar a ação (mensagem 1.2.1).

Após obter o certificado, a aplicação DS SCREAD calcula o resumo (*hash*) do arquivo informado pelo usuário e envia à API (mensagem 1.4) o conjunto de *bytes* correspondente ao

resumo calculado do arquivo e outros parâmetros necessários para a assinatura digital que incluem o formato de sumário de mensagem (SHA-1), o algoritmo de assinatura com chave (RSA) e a senha que permite acessar a chave privada armazenada no cartão (PIN ou *Personal Identification Number*). A API envia os comandos necessários ao dispositivo SCREAD MOD (mensagem 1.4.1) e retorna o *hash* assinado ao software cliente da API (DS SCREAD).

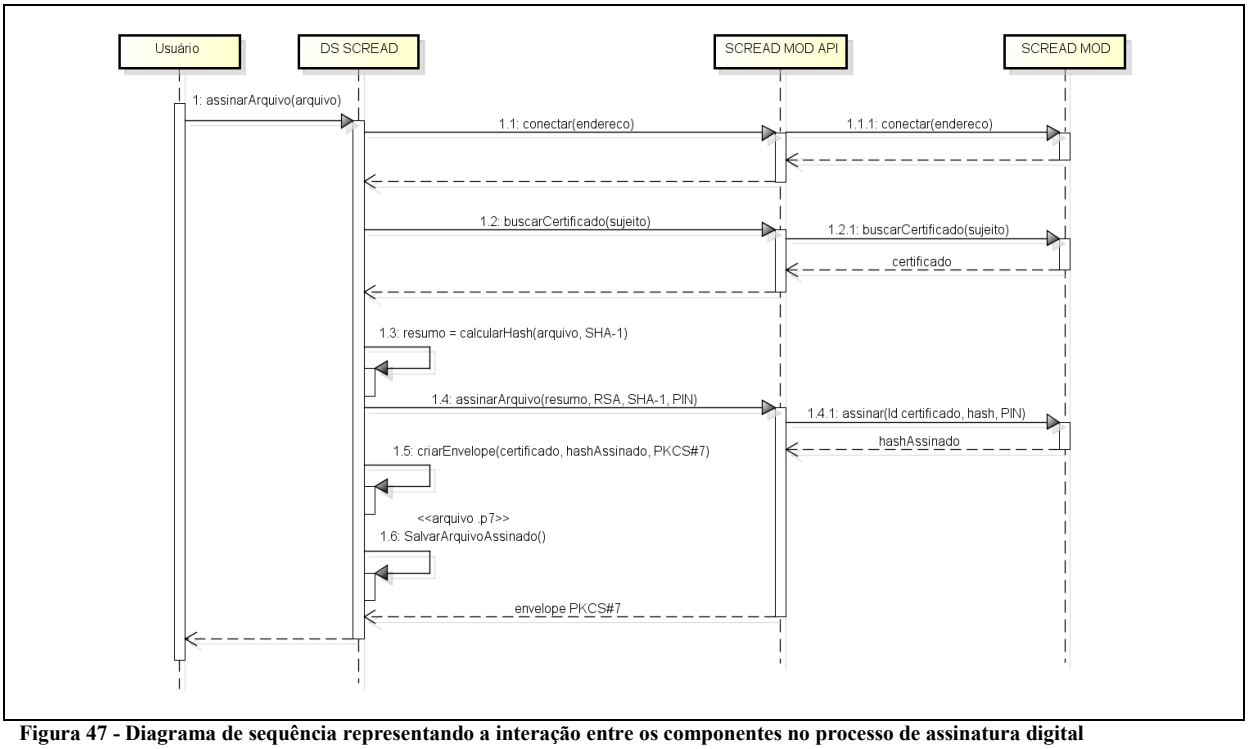

Figura 47 - Diagrama de sequência representando a interação entre os componentes no processo de assinatura digital
Fonte: Elaborado pelo autor (2011)

Após obter o *hash* assinado a aplicação DS SCREAD se encarrega de gerar um envelope de arquivo assinado padrão PKCS#7 (mensagem 1.5) e salvar o arquivo no cartão SD presente no *smart phone* (mensagem 1.6) com o sufixo ".p7b".

É possível perceber que a API retorna apenas o resumo assinado, possibilitando à aplicação criar o tipo de empacotamento desejado, podendo gerar outros formatos de arquivos e utilizar a assinatura em outros contextos, como por exemplo, para autenticação de um navegador web utilizando protocolo SSL ou TLS.

CAPÍTULO 8
- CONCLUSÃO

Este trabalho de mestrado apresentou o projeto e o protótipo do SCREAD MOD, um dispositivo que possibilita a utilização de cartões inteligentes utilizando a tecnologia de comunicação sem fio *Bluetooth*. O desenvolvimento de tal dispositivo foi guiado pela premissa da portabilidade, pois tem como intuito a utilização em conjunto com dispositivos móveis que incluem telefones celulares inteligentes, *tablet's*, *notebooks* e outros.

O desenvolvimento de um dispositivo de *hardware* agrega pouco valor caso não tenha sistemas de *software* que o tornem útil. Por este motivo, além do dispositivo SCREAD MOD foram desenvolvidos sistemas de *software* que permitiram validar o protótipo do SCREAD MOD através da demonstração de que é possível utilizá-lo para realizar assinatura digital de documentos através de um cartão inteligente conectado ao leitor de cartões desenvolvido.

A interdisciplinaridade inerente à construção de um dispositivo mecatrônico derivou a necessidade de uma pesquisa do estado da arte que envolveu questões relacionadas à parte de *software*, questões relacionadas à parte de *hardware* e questões que intercedem ambos os escopos.

Características físicas da construção de um equipamento que se relaciona a um sistema computacional incluem possíveis diferentes meios de comunicação (com fio x sem fio), diferentes protocolos/padrões de acesso ao meio, escolha de componentes que tornam a escolha possível de ser implementada e outros detalhes. Dado que diversas possibilidades de acoplamento entre o SCREAD MOD e dispositivos inteligentes móveis estavam disponíveis antes de se iniciar o projeto foi realizado um estudo que refletiu no desenvolvido dos capítulos 4 e 5.

O Capítulo 4 apresentou tecnologias de acesso ao meio sem fio que incluem as tecnologias IrDA, Bluetooth, WiFi e NFC e após um estudo comparativo entre as tecnologias sem fio conclui-se que a tecnologia *Bluetooth* é melhor alternativa para o propósito do SCREAD MOD. Por outro lado, o capítulo 5 apresentou padrões de comunicação que

pressupões o acoplamento físico através de fios amplamente utilizadas entre as quais se destaca o padrão USB e o padrão de comunicação UART, sendo este ultimo utilizado no desenvolvimento do projeto.

Cartões inteligentes são amplamente utilizados para armazenar certificados digitais e certificados digitais associados a chaves privadas e públicas são muito utilizados para se realizar assinatura digital. A realização de assinatura digital envolve a utilização de informações críticas, como por exemplo, uma senha ou PIN associado ao cartão; o tráfego de informações sigilosas pelo ar necessita de mecanismos que possibilitem que a informação que trafegue não possa ser interceptada, alterada, etc., ou seja, necessita que sejam utilizados serviços de segurança.

O Capítulo 2 apresentou os conceitos relacionados a comunicações seguras que envolvem criptografia simétrica e assimétrica, funções de *hash*, certificados digitais, assinaturas digitais e outros conceitos que permitiram compreender como é possível realizar comunicações seguras entre dispositivos e que cartões inteligentes podem ser considerados como mecanismos eficazes na realização de assinaturas digitais. Além disso, foi abordado o conceito de infraestrutura de chaves públicas e apresentada a Infraestrutura de Chaves Públicas do Brasil (ICP-Brasil) bem como suas diversas classificações para certificados digitais.

Como o foco do SCREAD MOD é a utilização acoplada a dispositivos móveis, fez-se necessário definir um dispositivo computacional móvel a ser utilizado em conjunto com o SCREAD MOD. O Capítulo 7 apresentou as decisões de projeto a respeito do desenvolvimento do SCREAD MOD que incluíram a justificativa por escolha de tecnologias adotadas e a apresentação das características dos componentes eletrônicos adotados mais relevantes.

Decisões técnicas relacionadas a tecnologias a serem utilizadas não são suficientes para estabelecer interoperabilidade com dispositivos presentes no mercado ou mesmo ter aceitações por parte de entidades regulatórias. Diversas normas internacionais estão atualmente presentes para definir características físicas de equipamentos, características

comportamentais de *softwares* e até procedimentos que impõem o modo como os componentes físicos e lógicos devem se relacionar.

O capítulo 3 apresentou uma diversidade de normas e padrões. Algumas destas normas tiveram que ser exploradas para que o desenvolvimento deste trabalho se tornasse possível. Em especial as seguintes normas (ou partes delas) tiveram importância fundamental no desenvolvimento e/ou entendimento das atividades relacionadas aos conteúdos que seguem:

- Dimensões e localizações dos contatos do chip integrado ao cartão:
    - ISO/IEC 7816-2: *Cards with contacts: Dimensions and location of the contacts*;
- Comunicação elétrica com o cartão inteligente:
    - ISO/IEC 7816-3: *Electronic signals and transmission protocols*;
- Comandos suportados pelo cartão inteligente utilizados:
    - ISO/IEC 7816-4: *Organization, security and commands for interchange*;
    - ISO/IEC 7816-15: *Cryptographic information application*;
    - PKCS#15: *Cryptographic Token Information Syntax Standard*;
- Interpretação das respostas dos cartões:
    - ITU-T X.509: *The Directory: Public-key and attribute certificate frameworks*;
    - ITU-T X.690: *ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*;
    - PKCS#15: *Cryptographic Token Information Syntax Standard*;

- Montagem de pacotes assinados:
    - PKCS#7: *Cryptographic Message Syntax Standard*;

A norma ISO/IEC 7816, por ser considerada a mais relevante na comunicação com cartões inteligentes, foi detalhada no Capítulo 6.

Após descrever como os diversos conteúdos da dissertação se inter-relacionam deve-se destacar novamente que, conforme visto na introdução, a utilização de cartões inteligentes tende a aumentar consideravelmente. Esta tendência tem como fatores relevantes iniciativas de fomento do governo em sentido amplo quem incluem edição de leis e normas que regulamentam a infraestrutura de chaves públicas do Brasil (ICP-Brasil) e dão validade jurídica às assinaturas digitais, processo de adoção do RIC (Registro Único de Identidade Civil) e do e-CPF/e-CNPJ assim como características técnicas atreladas ao uso dos cartões inteligentes.

Foi possível inferir que a solução proposta neste trabalho possui um grande potencial de mercado. Fato este possível de identificar porque atrelado à forte tendência de aumento na utilização de cartões inteligentes existe um aumento notório na venda de telefones inteligentes. Adicionalmente uma pesquisa de mercado identificou apenas um único equipamento equivalente sendo comercializado no mundo enquanto que outro deve ser lançado ainda esse ano. Paralelamente uma busca na base de patentes nacionais não identificou nenhum produto equivalente nos registros do INPI.

Dado o exposto anteriormente, conclui-se que o desenvolvimento do SCREAD MOD (*Smart Card Reader for Mobile Devices*), além de ter uma boa aplicabilidade, pode ser considerado um produto inovador, visto que não foram identificados concorrentes e nem patentes no Brasil.

Destacam-se como contribuições acadêmicas deste trabalho um comparativo entre tecnologias de comunicação sem fio, a identificação de soluções que podem ser utilizadas para minimizar o consumo de energia de dispositivos mecatrônicos, a identificação de características relacionadas a dispositivos móveis, assim como soluções relacionadas a criptografia, certificados digitais, assinaturas digitais e outros, uma vez que, ao longo do

trabalho, objetivou-se traçar um paralelo, sempre que possível, entre o conhecimento teórico e as soluções disponíveis que implementam aquele conhecimento unindo conhecimentos interdisciplinares.

8.1 - TRABALHOS FUTUROS

O dispositivo SCREAD MOD apresentado, embora funcional e inovador, ainda é um protótipo e muitas sugestões de melhorias podem ser identificadas. Como sugestões de trabalhos futuros, destacam-se:

- **API para outras plataformas**: desenvolver um driver para que possa ser utilizado em outros sistemas operacionais para celulares, como iOS, Windows Mobile, Symbian e outros sistemas;

- **Desenvolver driver para PC's**: desenvolvimento de um driver que possa ser utilizado em computadores pessoais como uma camada inferior à API PCSC que é a API padrão para acesso aos recursos de *smart cards* no ambiente Windows e está disponível também para Linux;

- **SCREAD MOD como um extensor de bateria**: os *smart phones* estão consumindo cada vez mais energia devido ao aumento de recursos e capacidade de processamento. O SCREAD MOD é um dispositivo que tem como objetivo ser acoplado a um celular, uma sugestão de melhoria é que ele possa funcionar como um extensor de bateria ao *smart phone*. A maioria dos *smart phones* podem ser alimentados pela porta USB e o fato de SCREAD MOD ser alimentado por 5V (equivalente ao USB) torna essa convergência mais fácil;

- **Interfaces para aceitar *tokens* USB**: a grande maioria dos celulares funciona como dispositivos USB *slave* e não podem interagir diretamente com outros dispositivos USB. Atualmente, além de *smart cards* existem muitos *tokens* criptográficos USB sendo utilizados. O SCREAD MOD poderia funcionar como

um Host USB para interligar um *smart phone* a um celular e tornar sua aplicabilidade ainda mais abrangente.

- **Utilizar um módulo *Bluetooth* classe 3**: o módulo *Bluetooth* utilizado – KC21 – é um módulo classe 2. A utilização de um módulo *Bluetooth* classe três irá propiciar mais economia de energia e mais segurança, uma vez que o SCREAD MOD somente será visível a dispositivos que estejam até um metro de distância.

REFERÊNCIAS

AXELSON, J. **Serial Port Complete:** COM ports, USB virtual COM ports and ports for embedded systems. Second edition. ed. Madison: Lakeview Research, 2007. ISBN ISBN 978-1931448-07-9.

AXELSON, J. **USB Complete:** The Developer's Guide. 4 edition. ed. Madison: Lakeview Research, 2009. 506 p. ISBN ISBN-13: 978-1931448086.

BEYOND LOGIC. **USB in a NutShell:** Making sense of the USB standard. [S.l.]: [s.n.], 2010. Disponivel em: <http://www.beyondlogic.org/usbnutshell/usb1.shtml>. Acesso em: 19 Novembro 2010.

BLUETOOTH RANGE. Bluetooth Range. **Bluetooth Range - Read about Bluetooth range**, 2011. Disponivel em: <http://bluetoothrange.com/bluetooth-range.html>. Acesso em: 17 Maio 2011.

BRASIL, I. **Requisitos mínimos para as políticas de certificado na ICP-Brasil**. [S.l.]. 2010. (Versão 3.2).

BURNNET, S.; PAINE, S. **RSA Secutiry's Official Guide to Criptography**. 1st Edition. ed. Chicago: McGraw-Hill, 2001. 449 p. ISBN ISBN-13:978-0072131390.

CERTISIGN. **Identidade Digital - Como os certificados digitais estão facilitando a vida das pessoas**. 2ª edição. ed. São Paulo: Câmara Brasileira do Livro, 2007. ISBN ISBN 978-85-60189-00-7.

COUTINHO, S. C. **Números Inteiros e Criptografia RSA**. Segunda edição. ed. Rio de Janeiro: IMPA, 2007. 213 p. ISBN ISBN 978-85-244-0124-4.

DEITEL, H. M.; DEITEL, P. J.; STEINBUHLER, K. **E-Business E E-Commerce Para Administradores**. São Paulo: MAKRON, 2004.

DIVINEY, G. **An Introduction to Short-Range Wireless Data Communications**. San Francisco, p. 13. 2003.

EVERETT, D. D. Smart Card Tutorial. **Smart Card News**, 2002. Disponivel em: <www.smartcard.co.uk/tutorials/sct-itsc.pdf>. Acesso em: 15 Fevereiro 2011.

FEDERAL INFORMATION PROCESSING STANDARD. Computer Security Division. **NIST.gov - National Institute of Standard and Technology**, 25 October 1999. Disponivel em: <http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf>. Acesso em: 10 January 2011.

GARCIA, F. C. O. Da validade jurídica dos contratos eletrônicos. **Jus Navandi**, Teresina, 28 Março 2004. Disponivel em: <http://jus.uol.com.br/revista/texto/4992/da-validade-juridica-dos-contratos-eletronicos>. Acesso em: 12 Março 2011.

GRIDLING, G.; WEISS, B. **Introduction to Microcontrollers**. Vienna University of Technology. [S.l.]. 2007.

HID GLOBAL. **OMNIKEY® 2061 Bluetooth® Reader Datasheet**. [S.l.]. 2011.

HYDE, J. **USB Design by Example:** A Practical Guide to Building I/O Devices. 2nd Edition. ed. [S.l.]: Intel Press, 2001. 510 p. ISBN ISBN-13: 978-0970284655.

IEEE. **IEEE 802.15: WIRELESS PERSONAL AREA NETWORKS (PANs)**. [S.l.]. 2006.

ISO/IEC 1786-4. **Identification cards — Integrated circuit - Part 4: Organization, security and commands for**. ISO/IEC. Geneva. 2005. (ISO/IEC 7816-4:2005).

ISO/IEC. **Information technology — Security techniques — Code of practice for information security management**. Geneva, p. 136. 2005. (ISO/IEC 27002:2005(E)).

ISO/IEC. Part 1: Introduction and general model. In: _____ **Common Criteria for Information Technology Security Evaluation**. Switzerland: [s.n.], 2009.

ISO/IEC 7816-1. **Identification cards — Integrated circuit cards — Part 1: Cards with contacts — Physical characteristics**. ISO/IEC. Geneva. 2011. (ISO/IEC 7816-1:2011(E)).

ISO/IEC 7816-15. **ISO/IEC 7816-15- Part 15: Cryptographic information application**. ISO/IEC. Geneva. 2004. (ISO/IEC 7816-15:2004(E)).

ISO/IEC 7816-9. **ISO/IEC 7816 - Part 9: Commands for card management**. ISO/IEC. [S.l.]. 2004.

ITI. **DOC-ICP-15 - Visão geral sobre assinaturas digitais na ICP-Brasil**. Instituto Nacional de Tecnologia da Informação (ITI). Brasília. 2010.

JR., B. S. K. **An Overview of the PKCS Standards**. [S.l.]. 1993.

KUROSE, J. F.; ROSS, K. W. **Redes de computadores e a internet. Uma abordagem top-down**. 3ª. ed. São Paulo: Pearson, 2006.

MEIER, R. **Professional Android Application Development**. First Edition. ed. Indianapolis: Wiley Publishing, 2009. ISBN ISBN: 978-0-470-34471-2.

MICROCHIP TECHNOLOGY. **PIC18F2455/2550/4455/4550 Datasheet**. MicrochipTechnology. [S.l.]. 2009.

MORENO, E. D.; PEREIRA, F. D.; CHIARAMONTE, R. B. **Criptografia em Software e Hardware**. 1ª edição. ed. [S.l.]: Novatec Editora, 2005. ISBN ISBN: 85-7522-069-1.

MORKEL, T.; ELOFF, J. **Encryption Techniques:** A timeline approach. South Africa: Information and Computer Security Architecture (ICSA) Research Group. 2004.

MULLER, N. J. **Bluetooth Demystified**. [S.l.]. 2000. (ISBN 13: 978-0071363235).

NORMAN, B. **Secret Warfare:** The Battle of Codes and Ciphers. [S.l.]: Borgo Press, 1990. ISBN SBN-13: 978-0809575794.

ORACLE. An Introduction to Near-Field Communication and the Contactless Communication API. **Oracle Developer Network**, 2011. Disponivel em: <http://java.sun.com/developer/technicalArticles/javame/nfc/>. Acesso em: 20 maio 2011.

PIGNATARO, L. **Smartcards – Operando em Baixo Nível**. Universidade de Brasilia. Brasília. 2006.

POLÍCIA FEDERAL. Passaporte Eletrônico. Disponivel em: <http://www.dpf.gov.br/servicos/passaporte/passaporte-eletronico/>. Acesso em: 01 mar. 2011.

PORTAL BRASIL. Conheça o novo Registro de Identidade Civil (RIC), 2010. Disponivel em: <http://www.brasil.gov.br/sobre/cidadania/documentacao/conheca-o-novo-registro-de-identidade-civil-ric>. Acesso em: 30 jan. 2011.

RESEARCH IN MOTION. **BlackBerry Smart Card Reader**. [S.l.]. 2007.

RSA. PKCS#15: Criptographic Token Information Format. **RSA Laboratories**, 2010. Disponivel em: <http://www.rsa.com/rsalabs/node.asp?id=2141>. Acesso em: 17 Novembro 2011.

RSA LABORATORIES. Public Key Cryptography Standards (PKCS). **RSA, The Security Division of EMC**. Disponivel em: <http://www.rsa.com/rsalabs/node.asp?id=2124>. Acesso em: 14 Dezembro 2010.

RSA LABORATORIES. What is Diffie-Hellman. **RSA Laboratories**. Disponivel em: <http://www.rsa.com/rsalabs/node.asp?id=2248>. Acesso em: 29 Novembro 2010.

RSA SECURITY INC. Public-Key Cryptography Standards (PKCS). In: INC, R. S. **Public-Key Cryptography Standards**. [S.l.]: [s.n.], 2000. Cap. PKCS#15.

SERPRO. Segurança do oiapoque ao chuí. **TEMA - A revista do SERPRO**, 2004.

SILVA, D. F. D. **Sistema de Comunicação Bluetooth utilizando microcontrolador**. Recife. 2009.

SILVA, Y. A. D. **Estudo e proposta de um novo documento de identificação eletrônica (e-ID) para o Brasil**. Universidade de Brasília. Brasília. 2007.

SNELL, S. **Mobile Design for iPhone and iPad**. First Edition. ed. Freiburg: Smashing magazine, 2010.

STALLINGS, W. **Cryptography and Network Security Principles and Practices**. Fourth Edition. ed. Upper Saddle River: Prentice Hall, 2005.

SUBIR KUMAR SARKAR, T. G. B. T. G. B. **Ad Hoc Mobile Wireless Networks**. New York: Auerbach Publications, 2008. ISBN ISBN 978.1.4200.6221.2.

TANENBAUM, A. S. **Redes de Computadores**. 4ª Edição. ed. São Paulo: Campus, 2003.

U.S. GSA. **Government smart card handbook**. U.S. General Services Administration. [S.l.]. 2004.

VYAS, D. D.; PANDYA2, D. H. N. **A Survey of Short-range Wireless Communication Technologies for Embedded Systems**. Atmiya Institute of Technology and Science. Rajkot. 2010.

WHITAKER, R. M.; HODGE, L.; CHLAMTAC, I. Bluetooth scatternet formation: A survey. **Elsevier**, Dallas, 03 March 2004.

WIKIPEDIA. Near field communication. **Wikipedia, the free encyclopedia**, 2011. Disponivel em: <http://en.wikipedia.org/wiki/Near_field_communication>. Acesso em: 12 Maio 2011.

WIKIPEDIA. Federal Information Processing Standard. **Wikipedia, The Free Enciclopedia**. Disponivel em: <http://en.wikipedia.org/wiki/Federal_Information_Processing_Standard>. Acesso em: 20 mar. 2011.

ANEXO A

– PESQUISA SOBRE SMART PHONES

COMPATIBILIDADE COM PERFIS DO *BLUETOOTH*

			Compatibilidade Bluetooth																		
	Modelo	Conexão USB	SDAP	HFP	HSP	OPP	PBAP	FTP	A2DP	AVRCP	DUN	SPP	GAP	GOEP	PAN	L2CAP	AVDTP	SDP	BPP	GAVDP	SAP	AVCTP
LG	LG P500	microUSB	x	x	x	x	x	x	x	x	x	x										
	LG P970 Optimus	microUSB							x													
	LG LU3000 Optimus	microUSB		x	x	x	x	x	x	x	x	x	x					x		x		
Samsung	Galaxy S	miniUSB		x	x	x	x	x	x	x	x	x	x			x	x	x	x	x	x	x
	Samsung I5510 Galaxy 551	microUSB		x	x	x	x	x	x	x	x	x	x				x	x		x		x
	Samsung I9100 Galaxy S II	microUSB			x																	
Sony	Xperia Mini	microUSB	x	x	x	x	x		x	x		x	x				x			x		x
	Sony Ericsson Xperia X10	microUSB		x	x	x	x	x	x	x												
	Sony Ericsson Xperia Arc / X12	microUSB		x	x	x			x	x												
HTC	HTC Google Nexus One	microUSB				x	x		x													
	HTC Magic	Proprietário							x													
	HTC Desire	microUSB	x	x	x	x	x	x	x	x		x	x	x								

COMPATIBILIDADE COM NFC

Fabricante	ID	Modelo	Sistema operacional	Ano de lançamento	Status
Nokia	1	Nokia C7-00	Symbian 3	Set / 2010	
	2	Nokia 6131	Symbian 3	Jun / 2006	Descontinuado
	3	Nokia 6680	Symbian	2005	Descontinuado
	4	Nokia 3320		2004	Descontinuado
Samsung	5	Samsung S5230	Não especificado	Março/2009	
	6	Samsung SGH-X700	Desconhecido	Novembro/2005	Descontinuado
	7	Samsung D500E	Não especificado	2004	Descontinuado
	8	Samsung Wave 578	BADA OS	Fevereiro/2011	
	9	Samsung Galaxy S II (algumas versões)	Android 2.3	Abril/2011	
	10	Samsung Google Nexus S	Android 2.3	Dezembro/2010	
	12	LG 600V contactless	Desconhecido	Outubro/2006	Descontinuado
	13	Motorola L7 (SLVR)	Desconhecido	2005	Descontinuado
	14	Benq T80	Desconhecido	Ago / 2007	
	15	BlackBerry Bold 9900/9930		Mai / 2011	
SAGEM	15	Sagem Cosyphone	Desconhecido	Nov/2010	
	16	SAGEM my700X Contactless	Desconhecido	Fev/2006	Descontinuado

FONTE:

1 - http://www.gsmarena.com/nokia_c7-3394.php
2 - http://www.gsmarena.com/nokia_6131-1434.php
3 - http://www.gsmarena.com/nokia_6680-1045.php
4 - http://www.gsmarena.com/nokia_3220-801.php
5 - http://en.wikipedia.org/wiki/Samsung_S5230
6 - http://www.gsmarena.com/samsung_x700-1298.php
7 - http://www.gsmarena.com/samsung_d500-900.php
10 - http://www.samsungnexuss.com/nexus-s-specs/
12 - http://www.gsmarena.com/lg_l600v-1745.php
13 - http://www.gsmarena.com/motorola_slvr_l7-1053.php
14 - http://www.conita.com/Mobile-Phones/BenQ-T80-Mobile-Phone-Review.html
15 - http://www.gsmarena.com/blackberry_bold_touch_9900_and_9930_announced_run_on_blackberry_os_7-news-2591.php
17 - http://www.gsmarena.com/sagem_my700x-1432.php

ANEXO B
- PINAGEM DO MICROCONTROLADOR E DO MÓDULO BLUETOOTH

Tabela 15 - Pinos do microcontrolador PIC18F4550 agrupados por funcionalidades

Fonte: (MICROCHIP TECHNOLOGY, 2009)

Nota: adaptado pelo autor (2011)

	Nome	Pinos	Descrição
ENERGIA	VDD	11, 32	Positive supply for logic and I/O pins
	VSS	12, 31	Ground reference for logic and I/O pins.
	MCLR	1	*Master Clear Reset*
PORTAS DIGITAIS	RA[0-7]	2-10	Entrada/Saida digital da porta A.
	RB[0-7]	33-40	Entrada/Saida digital da porta B.
	RC[0-7]	15-17, 23-26	Entrada/Saida digital da porta C. Obs.: O pino RC3 não está implementado.
	RD[0-7]	19-22; 27-30	Entrada/Saida digital da porta D.
	RE[0-3]	8-10;1	Entrada/Saida digital da porta E.
PORTAS ANALÓGIACS	AN[0-7]	2-10	Entrada analógica
	VRef-	4	A/D reference voltage (low) input
	VRef+	5	A/D reference voltage (high) input
	CVRef	4	Analog comparator reference output
	C[1,2]OUT	6,7	Comparator [1,2] output
	HLVDIN	7	High/Low voltage detect input
USART	TX	25	EUSART asynchronous transmit
	CK	25	EUSART synchronous clock (see RX/TX)
	RX	26	EUSART asynchronous receive
	DT	26	EUSART synchronous data
USB	D-	23	USB differential minus line (input/output)
	D+/VP	24	External USB transceiver VP input
	VUSB	18	Internal USB 3.3 Voltage
	RCV	6	External USB transceiver RCV input
	VPO	36	External USB transceiver VPO output
	VMO	35	External USB transceiver VPO output
	UOE	16	External USB transceiver OE output
INTERRUPÇÃO	T1OSO	15	Timer 1 Oscilator Output
	T13CKI	15	Timer 1/3 External clock Input
	T1OSI	16	Timer 1 oscilator input
	INT[0,1,2]	33, 34, 35	External Interrupt [0, 1, 2]
	KBI[0,1,2,3]	37-40	Interrupt on change pin

ICSP	PGM	38	Low Voltage ICSP Programming enable pin
	PGC	39	In-Circuit Debbuger and ICSP programming clock pin
	PGD	40	In-Circuit Debugger and ICSP programming data pin
SPI	SDO	26	SPI data output
	SCK	34	Synchronous serial clock input/output for SPI mode
	SS	7	SPI slave select input
	SDI	33	SPI data in
I2C	SCL	34	Synchronous serial clock input/output for I2C mode
	SDA	33	I2C data I/O
PWM	FLT0	33	PWM Fault input (CCP1 module)
	CCP[1,2]	17, 36	Capture [2] input/compare [2] output/PWM 2 output

A Tabela 16 contém a posição, a sigla que representa e a descrição dos pinos presentes no módulo KC-21.

Tabela 16 - Pinos do módulo Bluetooth KC-21
Fonte: (SILVA, 2009)

Pinos	Sigla	Descrição
1, 4, 5, 7, 13, 14, 15, 16, 17, 18, 19, 20, 21 e 22	PIO	Pinos programáveis como de entrada e de saída de dados
2, 11 e 12	RESV	Reservados
3	RSET	Entrada de reset
6	TXD	Transmissão de dados
8	RXD	Recepção de dados
9	CTS	Utilizado para controle de fluxo – Clear to Send
10	RTS	Usado para controle de fluxo – Request to Send
23	GND	Terra
24	VDD	Alimentação

ANEXO C
– PROGRAMA DO MICROCONTROLADOR

Este anexo contém as partes mais relevantes do programa fonte main.c. Será mostrado as partes principais do programa principal dividido nos seguintes tópicos: arquivos de cabeçalho, variáveis principais, cabeçalho de funções, funções de interrupção e o programa principal.

ARQUIVOS DE CABEÇALHO UTILIZADOS

```
#include <p18f4550.h>
#include <delays.h>
#include <string.h>

#include "./lib/SW_UART/sw_uart.h"
#include "./lib/SW_UART/sw_uart_config.h"

#include "./lib/Smart Card/sc_config.h"
#include "./lib/Smart Card/SClib.h"
```

VARIÁVEIS

As variáveis que seguem armazenam os comandos que contemplam o APDU de envio ao cartão e as respectivas respostas.

```
// APDU Command to the Card
SC_APDU_COMMAND cardCommand;

// APDU Response from the Card
SC_APDU_RESPONSE cardResponse;

// T=1, Prologue Field
SC_T1_PROLOGUE_FIELD prologueField;


// Store the APDU Command/Response Data in a 256 bytes register bank(RAM)
#pragma udata apdu_data
BYTE apduData[SC_APDU_BUFF_SIZE];
#pragma udata
```

FUNÇÕES:

Cabeçalho das funções declaradas no programa fonte *main.c*:

```
void configurarPIC(void);
void high_isr(void);
void LowVector (void);
void escreverBluetoothStr(char *txt);
void escreverBluetooth(SC_APDU_RESPONSE *resp);
void lerBluetooth(SC_APDU_COMMAND *cmd);
void reiniciar();
```

INTERRUPÇÕES

São tratadas duas interrupções com, respectivamente, baixa e alta prioridade: low_ISR e high_ISR.

```
#pragma code lowhVector=0x18
void interrupt_at_low_vector (void) {
      _asm goto low_ISR _endasm
}
#pragma code

#pragma interruptlow low_ISR
void low_ISR(void) {
      //check for TMR0 overflow
      if (INTCONbits.TMR0IF)  {
            //clear interrupt flag
            INTCONbits.TMR0IF = 0;
            SCdrv_DisableDelayTimer();
            //indicate timeout
            delayLapsedFlag = 1;
            reiniciar();
      }
}
```

Interrupção de alta prioridade

```
#pragma code high_vector=0x08
void interrupt_at_high_vector(void) {
      _asm GOTO high_ISR _endasm
}
#pragma code

#pragma interrupt high_ISR
void high_ISR(void) {
      if (INTCON3bits.INT2IF) {
            if (!PORTBbits.RB2) {
                  PORTAbits.RA0 = 1;
                  INTCON2bits.INTEDG2 = 1;
```

```
                  INTCON3bits.INT2IF = 0;
                  escreverBluetoothStr("COM CARTAO\0");

            }
            else {
                  PORTAbits.RA0 = 0;
                  INTCON2bits.INTEDG2 = 0;
                  INTCON3bits.INT2IF = 0;
                  escreverBluetoothStr("SEM CARTAO\0");
            }
            reiniciar();
      }
}
```

FUNÇÃO PRINCIPAL

Segue o corpo da função principal:

```
void main(void) {
      //configura as interrupções, os LEDs e os bits do microcontrolador.
      configurarPIC();

      OpenUART();

      SC_Initialize();

      while( !SC_CardPresent() );

      while( !SC_PowerOnATR() )
            breakPoint++;

      while (!SC_DoPPS() )
            breakPoint++;

      prologueField.NAD = 0;
      prologueField.PCB = 0;
      prologueField.LENGTH = 5;
      while(SC_T1ProtocolType()) {

            lerBluetooth(&apduData);

            if (!SC_TransactT1(&prologueField, apduData, &cardResponse)){
                  while(1)
                        breakPoint++;
            }

            escreverBluetooth(&cardResponse);
      }

      CloseUART();
```

```
        reiniciar();
}
```

8.1 – DOCUMENTOS EM SMART CARDS

Qualquer situação em que se exija um documento e que este documento esteja *disponível* a partir de *smart cards* como é o caso dos documentos e-CPF, e-CNPJ, Passaporte eletrônico e RIC o SCREAD MOD pode ser utilizado, desde que os cartões sejam compatíveis com a norma ISO/IEC-7816. Isso inclui situações que envolvem utilização de serviços bancários, serviços governamentais ou transações entre particulares.

Cartões de crédito e débito seguem o padrão de cartões EMV que requer conformidade total com o padrão ISO/IEC 7816. Isso significa que com as respectivas atualizações de software e/ou firmware os cartões de crédito e débito VISA/Mastercard poderão ser utilizados para compras no SCREAD MOD