UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA

CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

TOMÁS DE CARVALHO COÊLHO

RUBRIK: UMA BIBLIOTECA OPEN SOURCE PARA IMPLEMENTAÇÃO DE ASSINATURA EM ARQUIVOS DO TIPO PORTABLE DOCUMENT FORMAT

FORTALEZA

2024

TOMÁS DE CARVALHO COÊLHO

RUBRIK: UMA BIBLIOTECA OPEN SOURCE PARA IMPLEMENTAÇÃO DE ASSINATURA EM ARQUIVOS DO TIPO PORTABLE DOCUMENT FORMAT

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Computação.

Orientador: Prof. Dr. José Marques Soares.

FORTALEZA

2024

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

C621r Coêlho, Tomás de Carvalho.
Rubrik: uma biblioteca open source para implementação de assinatura em arquivos do tipo Portable Document Format / Tomás de Carvalho Coêlho. – 2024.
46 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Centro de Tecnologia, Curso de Engenharia de Computação, Fortaleza, 2024.
Orientação: Prof. Dr. José Marques Soares.

1. Assinatura digital. 2. Certificado digital. 3. Documento digital. 4. PAdES. I. Título.
CDD 621.39

TOMÁS DE CARVALHO COÊLHO

RUBRIK: UMA BIBLIOTECA OPEN SOURCE PARA IMPLEMENTAÇÃO DE ASSINATURA EM ARQUIVOS DO TIPO PORTABLE DOCUMENT FORMAT

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Computação.

Aprovada em: 20/09/2024.

BANCA EXAMINADORA

Prof. Dr. José Marques Soares (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Ricardo Jardel Nunes da Silveira
Universidade Federal do Ceará (UFC)

Eng. Artur Rodrigues Rocha Neto
PulsarES

Ao meu pai, que sempre zelou e se preocupou com a minha educação. Mãe, seu cuidado e compreensão foi que deram, em alguns momentos, a esperança para seguir. Mateus, meu irmão, você sempre foi um exemplo para mim. Muito obrigado por todo o seu apoio e companhia.

AGRADECIMENTOS

À minha família, meu pai, mãe e irmão, que sempre foram a minha motivação e exemplos de pessoas em quem sempre me inspirei. Eu tenho orgulho de tê-los na minha vida.

Aos meus amigos de faculdade, do grupo "Segue a Call", que desde o início do curso são companheiros diários e inestimáveis em minha vida.

Ao Prof. Dr. José Marques Soares, pela compreensão, humanidade e sempre excelente orientação.

Aos professores participantes da banca examinadora Prof. Ricardo Jardel Nunes da Silveira e Eng. Artur Rodrigues Rocha Neto pelo tempo, pelas valiosas colaborações e sugestões.

"Os filósofos até hoje interpretaram o mundo,
cabe agora transformá-lo" (Karl Marx, 1845)

RESUMO

Com a crescente digitalização de vários aspectos das atividades humanas, as assinaturas eletrônicas constituem uma alternativa à assinatura manual, como forma de deixar o processo de autenticação de documentos ainda mais confiável, cômodo e barato. No Brasil, ainda há barreiras para a adoção dessa tecnologia por parte das organizações. Neste trabalho, objetiva-se implementar e disponibilizar uma biblioteca escrita na linguagem de programação Ruby que facilite o desenvolvimento de serviços tecnológicos de assinatura digital de documentos. Para possibilitar isso, Rubrik, um *software* capaz de ler certificados digitais e manipular documentos digitais, foi criado. Após isso, a biblioteca foi sujeitada a três metodologias de verificação, obtendo resultados satisfatórios em todos os testes. Adicionalmente foi constatada a validade jurídica da assinatura feita por Rubrik perante a lei brasileira. Disponibilizada de forma aberta e gratuita, novos recursos associados à assinatura digital podem ser implementados e disponibilizados pela comunidade.

Palavras-chave: assinatura digital; PAdES; certificado digital; documento digital

ABSTRACT

With the increasing digitization of various aspects of human activities, electronic signatures have emerged as an alternative to manual signatures, as a way to make the process of document authentication more reliable, convenient, and cheap. In Brazil, there are still barriers to the adoption of this technology by organizations. This work aims to create a library for the Ruby programming language that facilitates the development of digital signature softwares. To enable this, Rubrik, a library capable of reading digital certificates and manipulating digital documents, was created. After that, the library was subjected to three verification methodologies, obtaining satisfactory results in all tests. Therefore, despite some specific objectives not being achieved, the legal validity of the signature made by Rubrik under Brazilian law was confirmed.

Keywords: digital signature; PAdES; digital document; digital certificate.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AC-Raiz	Autoridade Certificadora Raiz
BPMN	*Business Process Model and Notation*
CMS	*Cryptographic Message Syntax*
DSA	*Digital Signature Algorithm*
ECDSA	*Elliptic Curve Digital Signature Algorithm*
ICP-Brasil	Infraestrutura de Chaves Públicas Brasileira
IETF	*Internet Engineering Task Force*
ITI	Instituto Nacional de Tecnologia da Informação
PAdES	*PDF Advanced Eletronic Signature*
PDF	*Portable Document Format*
PKCS #7	Cryptographic Message Syntax
RI	Representação Intermediária
RSA	*Rivest–Shamir–Adleman*
TLS	*Transport Security Layer*

SUMÁRIO

1 INTRODUÇÃO

Devido à crescente automatização das atividades humanas e representação de processos de diversas naturezas em meios digitais, assinaturas eletrônicas em substituição a assinaturas manuais tornam-se imprescindíveis. Para as organizações, essa tecnologia dá mais agilidade ao processo, reduz custos com materiais de escritório e beneficia o meio ambiente com a redução do uso de papel e da necessidade de transportar documentos (DocuSign, 2020).

Entretanto, em 2023, mais da metade dos pequenos negócios no Brasil ainda utilizavam contratos físicos, segundo pesquisa do SEBRAE SP (2023), apesar de terem planos para adotar assinaturas eletrônicas. Sobre as dificuldades das empresas que utilizam assinaturas eletrônicas, destacam-se algumas apontadas:

- 17% aponta riscos de não conformidade e problemas com auditorias;
- 15% tiveram experiências ruins;
- 15% tiveram problemas com custos operacionais.

Essas dificuldades dão indícios de que ainda há necessidade da produção de *softwares* e serviços mais acessíveis, seguros e fáceis de serem utilizados.

Adicionalmente, percebe-se a limitação de recursos disponíveis para facilitar a implementação de soluções para assinaturas digitais, em especial na linguagem Ruby e sua principal aplicação, o *framework* Ruby on Rails (2004), que é bastante utilizado para o desenvolvimento de aplicações *web* em várias organizações como Shopify, GitHub e Airbnb, que estão presentes na bolsa de valores, bem como do governo do Reino Unido, segundo GOV.UK (2024).

Nesse contexto, propõe-se a criação de Rubrik, uma biblioteca que oferece suporte para criação de serviços e softwares que implementam a técnica de assinatura digital, um tipo seguro de assinatura eletrônica, que também é regulamentado pela Lei Nº 14.063, de 23 de setembro de 2020.

A criação dessa biblioteca e o estudo da técnica de assinatura digital são temas deste trabalho.

1.1 Objetivos

O objetivo deste trabalho é criar uma biblioteca que implementa a técnica de assinatura digital em documentos digitais, arquivos com a extensão PDF. Essa biblioteca tem como finalidade facilitar o desenvolvimento de *softwares* ou serviços *online* de assinatura digital no

contexto brasileiro.

Desse modo, são objetivos específicos:

1. Implementar funções para a geração de assinaturas digitais;
2. Fornecer recursos que ofereçam validade jurídica para as assinaturas criadas por Rubrik;
3. Disponibilizar a biblioteca de forma gratuita para uso comercial por meio de uma licença de uso livre, isto é, *open source*;
4. Construir uma biblioteca tolerante a falhas.

1.2 Organização do documento

A organização desse trabalho se dá da seguinte maneira: no Capítulo 2 é apresentada a técnica de assinatura digital e seus componentes, como certificados digitais, a ICP-Brasil, sobre como alterar PDFs e sobre a linguagem de programação na qual Rubrik é implementada, o Ruby. No Capítulo 3, é apresentado o planejamento e a arquitetura da biblioteca Rubrik. No Capítulo 4, a biblioteca é verificada utilizando três métodos diferentes e os resultados desses testes são discutidos. No Capítulo 5 é discutido como o Rubrik atendeu aos objetivos iniciais e são abordadas funcionalidades futuras para o *software*.

2 FUNDAMENTOS

Neste capítulo, é realizada uma análise das propriedades inerentes às assinaturas digitais, destacando seu papel e as tecnologias e padrões envolvidos na implementação de assinaturas digitais em documentos PDF por meio do padrão *PDF Advanced Eletronic Signature* (PAdES), definido em ETSI TS 319 142-1 (2021). Além disso, é tratada a importância do envolvimento dos certificados digitais, da Infraestrutura de Chaves Públicas, da *Cryptographic Message Syntax* (CMS), da estrutura de um documento digital e da linguagem de programação Ruby. Por fim, há uma comparação entre a solução proposta e as demais alternativas disponíveis para a linguagem.

2.1 Assinaturas digitais

Assinaturas digitais são um conjunto de técnicas que objetivam ser o análogo das assinaturas manuais para conteúdo eletrônico. Suas características principais incluem a irrefutabilidade e a capacidade de verificação pública segundo Katz (2010).

Ainda segundo ele, irrefutabilidade significa que ao assinar uma mensagem, o signatário não pode negar a autoria de sua firma.

Já a capacidade de verificação pública significa que qualquer entidade pode validar a autenticidade da assinatura, desde que ela possua o certificado digital do signatário, o que será explicado adiante.

Considerando as características apresentadas das assinaturas digitais, percebe-se a sua adequação para várias finalidades, incluindo a assinatura de documentos, bem como a emissão de atestados e prescrições médicas. Estas e outras interações são regulamentadas pela Lei nº 14.063 (BRASIL, 2020).

Logo, esse tipo de assinatura contribui para a economia de papel, além de proporcionar maior agilidade ao processo de obtenção de firma, especialmente quando empregadas plataformas de assinatura via internet. Adicionalmente, a manutenção de registros é facilitada por essa modalidade de assinatura, visto que os registros são eletrônicos e podem ser indexados por software.

2.2 Funcionamento das assinaturas digitais

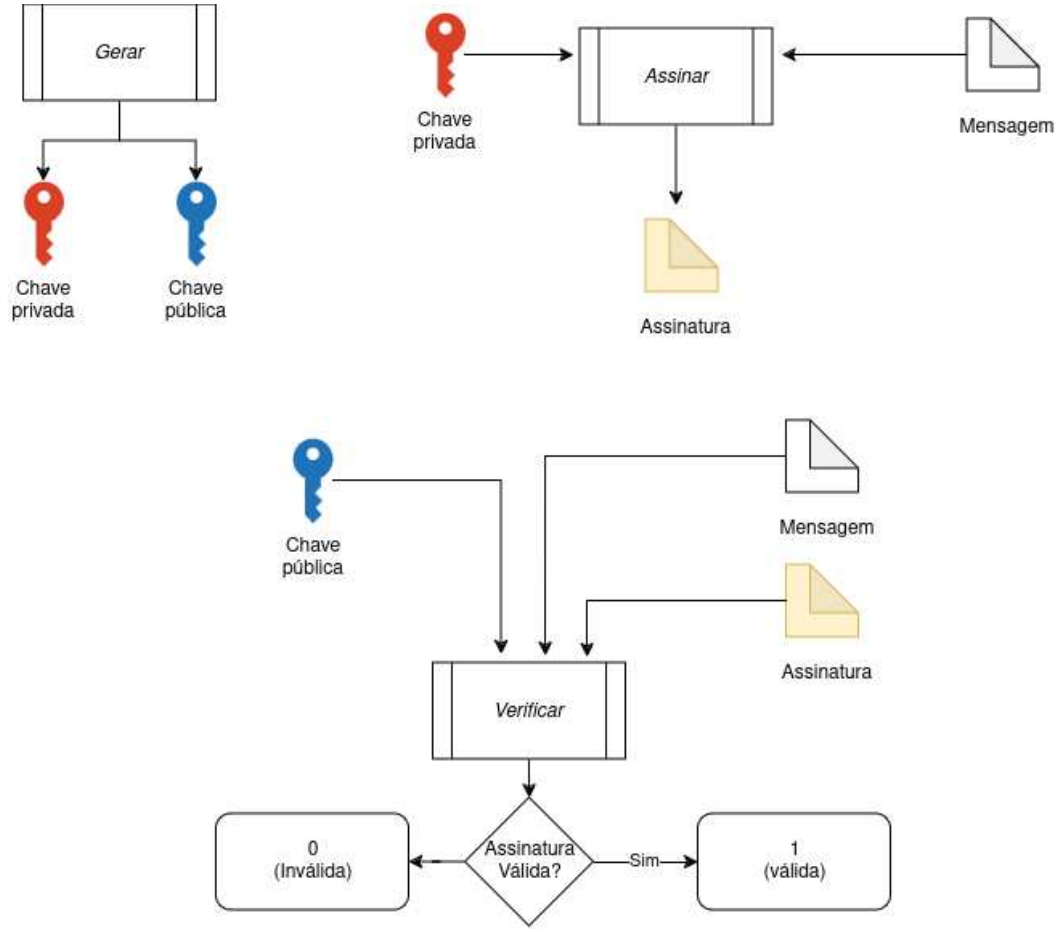
Segundo Katz (2010), é possível descrever essa tecnologia de maneira resumida como sendo uma estruturada em torno da implementação de três algoritmos: *Gerar*, *Assinar* e *Verificar*.

O algoritmo *Gerar* processa um parâmetro aleatório e resulta na produção de duas chaves inter-relacionadas, uma pública e uma privada, que são essenciais para os algoritmos restantes.

Por sua vez, o algoritmo *Assinar* funciona por meio da recepção de uma chave privada e de uma mensagem a ser assinada, retornando a assinatura da mensagem.

Finalmente, o algoritmo *Verificar* solicita como entrada uma chave pública, uma mensagem e a respectiva assinatura desta. Como resultado, esse algoritmo produz um único bit, que retorna um valor de 0 quando a mensagem não é aceita, ou 1, quando é verificada com sucesso. Esse processo está exemplificado de maneira gráfica na Figura 1.

Figura 1 – Representação gráfica das funções envolvidas na assinatura digital.



Fonte: Elaborado pelo autor.

Na prática, exemplos de algoritmos capazes de atender esses requisitos são *Rivest–Shamir–Adleman* (RSA), *Digital Signature Algorithm* (DSA) e *Elliptic Curve Digital Signature Algorithm* (ECDSA) para implementar as três funções descritas anteriormente. (BUCHMANN *et al.*, 2009).

2.3 Cadeias de certificados e infraestrutura de chaves públicas

Como visto na Seção 2.1, para que haja a verificação da assinatura digital, é preciso que o validador tenha uma cópia da chave pública do signatário, que é utilizada na função *Verificar* de acordo com o descrito na Seção 2.2.

Diante dessa necessidade, em diversas aplicações de assinaturas digitais, as chaves públicas são difundidas por meio de certificados digitais. Estes, por sua vez, são caracterizados pela *Internet Engineering Task Force* (IETF) no RFC 5280. Um exemplo de aplicação que utiliza certificados digitais é o protocolo *Transport Security Layer* (TLS), que atesta a identidade de sites na Internet.

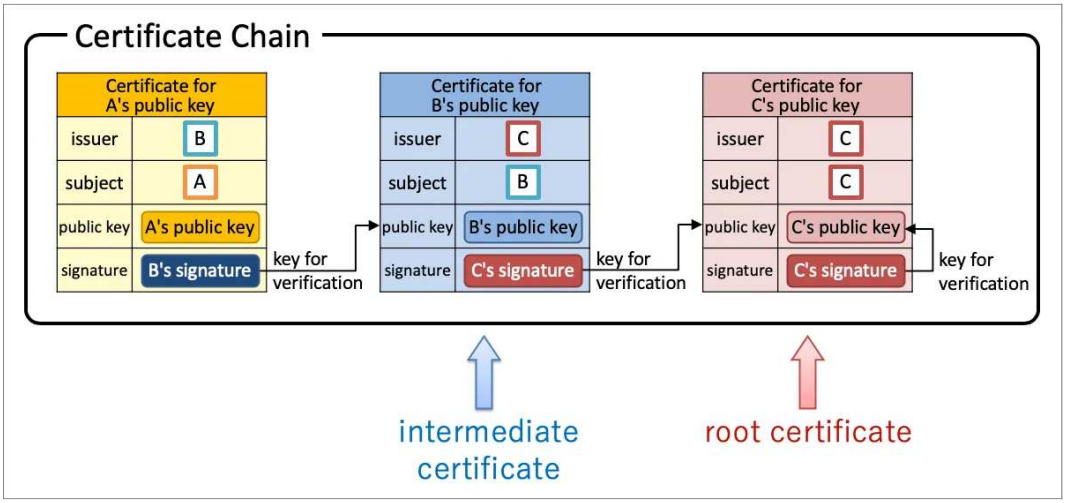
Segundo o RFC 5280, os certificados digitais compreendem uma estrutura de dados binária que atribui uma chave pública a uma entidade. Essa estrutura contém informações sobre o emissor do certificado, informações sobre a organização que solicita o certificado, uma chave pública e a assinatura de todo o conteúdo efetuada pelo emissor, por meio de sua chave privada, que está vinculada a um outro certificado digital, que pertence ao emissor.

Desse modo, é possível observar que todo certificado é assinado por outro certificado. Essa relação entre as assinaturas visa estabelecer uma cadeia de confiança, que parte do princípio de que se alguém confia que um certificado digital realmente pertence a uma entidade, então ela irá confiar que os certificados digitais assinados pela chave pública da entidade também pertencem a quem dizem pertencer. Na Figura 2, é possível identificar um exemplo da relação descrita anteriormente.

É possível evidenciar que o Certificado C, atuando como raiz da cadeia, é notavelmente auto-assinado, observação feita ao olhar os campos *issuer* e *signature* presentes. O Certificado B, em contraste, foi emitido e autenticado pelo Certificado C. Em sequência, o Certificado A foi emitido e autenticado pelo Certificado B.

Para a validação da autenticidade do Certificado A neste contexto, a obtenção do Certificado B torna-se necessária. Ademais, requer-se que o validador confie no Certificado C por meios extrínsecos à própria cadeia de certificados, visto que é auto assinado. Um meio

Figura 2 – Representação esquemática de uma cadeia de certificados.



Fonte: Kawasaki, Takahiko (2020).

externo poderia ser a troca presencial de certificados digitais por meio de *pendrives* ou outros dispositivos de armazenamento.

Assegurando que todas as condições ditas anteriormente são cumpridas, autentica-se a assinatura do Certificado B, contida em si mesma e feita pelo Certificado C. Após esta etapa, procede-se para a autenticação da assinatura de B para o Certificado A. Caso todas as assinaturas estejam válidas, é possível confiar em toda a cadeia.

Essa estrutura também está descrita no RFC 5280, que é a base da Infraestrutura de Chaves Públicas.

Uma Infraestrutura de Chaves Públicas inclui algumas entidades, sendo a principal a Autoridade Certificadora, que tem como função emitir e revogar certificados digitais para usuários da infraestrutura, mediante a confirmação de sua identidade.

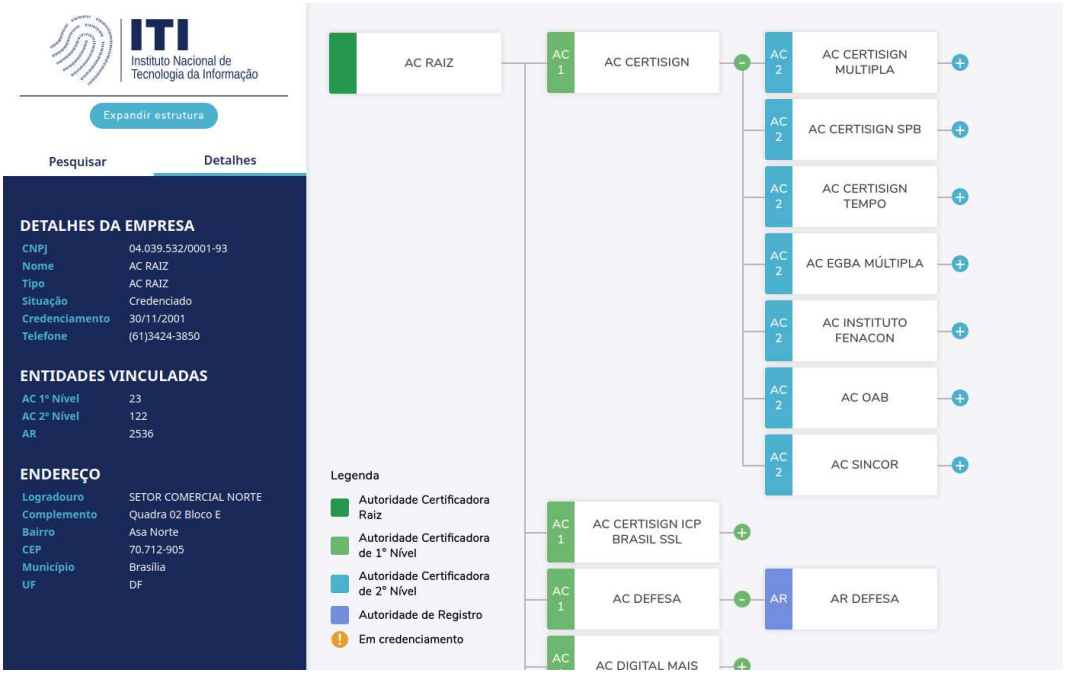
Dentre elas, temos a Autoridade Certificadora Raiz (AC-Raiz). Essa entidade é a única que possui um certificado digital não assinado por qualquer outra entidade. Essa característica única confere à AC-Raiz a capacidade de servir como a fundação para toda a cadeia de confiança.

Logo, a infraestrutura completa funciona da seguinte forma: a AC-Raiz emite um certificado. Esse certificado é difundido por fora da rede e é implicitamente confiado por todos. Ele é utilizado para assinar certificados para várias Autoridades Certificadoras, que, por sua vez, irão emitir certificados para pessoas físicas e jurídicas. Tendo em posse todos os certificados da cadeia até a raiz, é possível confiar na identidade de todos os participantes da cadeia.

No Brasil, há uma Infraestrutura de Chaves Públicas, a ICP-Brasil. Ela é administrada pelo Instituto Nacional de Tecnologia da Informação (ITI), que também exerce o papel de AC-

Raiz. A importância desse conjunto é destacada na legislação, como dito na Seção 2.1, a qual estipula que somente os certificados pertencentes à ICP-Brasil são autorizados para algumas finalidades, tais como, a emissão de atestados e receituários médicos e a transferência de imóveis em cartório, entre outras atividades.

Figura 3 – Estrutura da ICP-Brasil.



Fonte: ITI (2024).

Na Figura 3, podemos ver um pequeno recorte dessa Infraestrutura, que contém uma única AC-Raiz e várias Autoridades Certificadoras, além de uma Autoridade de Registro, que pode exercer algumas funções de Autoridades Certificadoras.

2.4 Documentos digitais e assinaturas

Muitas aplicações envolvem documentos digitais, como transferências de imóveis e prescrições médicas. Logo, além das técnicas vistas anteriormente, é necessário um padrão para a distribuição das assinaturas, visando a interoperabilidade de sistemas de informação e estabelecer definições seguras, conforme justifica DOC-ICP-15 (2009), que é o documento norteador sobre as assinaturas digitais na cadeia ICP-Brasil.

Um desses padrões é o PAdES, que define procedimentos para embutir uma assinatura digital, junto com uma cadeia de certificados digitais em um arquivo PDF, formato de arquivo mais popular da Internet segundo Common Crawl Foundation (2023).

Para compreender esse padrão, é fundamental entender o formato de arquivo PDF,

discutido na próxima subseção.

2.4.1 Portable Document Format

Esse formato é definido pela ISO 32000-2 (2020) e consiste em três componentes: objetos, estrutura do documento e estrutura do arquivo.

Os objetos são valores que podem ser de nove tipos primitivos: booleanos, inteiros, reais, *strings*, dicionários, *streams* e o objeto nulo. Eles também podem ter identificadores únicos. Em várias situações, um objeto pode ser substituído por seu identificador. Há ainda tipos complexos, que são compostos de dicionários com chaves e valores fixos.

A estrutura do documento consiste na composição de vários objetos, em um formato de árvore, de maneira a formar um documento válido. A raiz dessa árvore é um objeto do tipo *Document Catalog*, que é um dicionário que apresenta as chaves *Type, Version, Extensions, Pages, PageLabels*, entre outras. A estrutura de um documento PDF está representada na Figura 4.

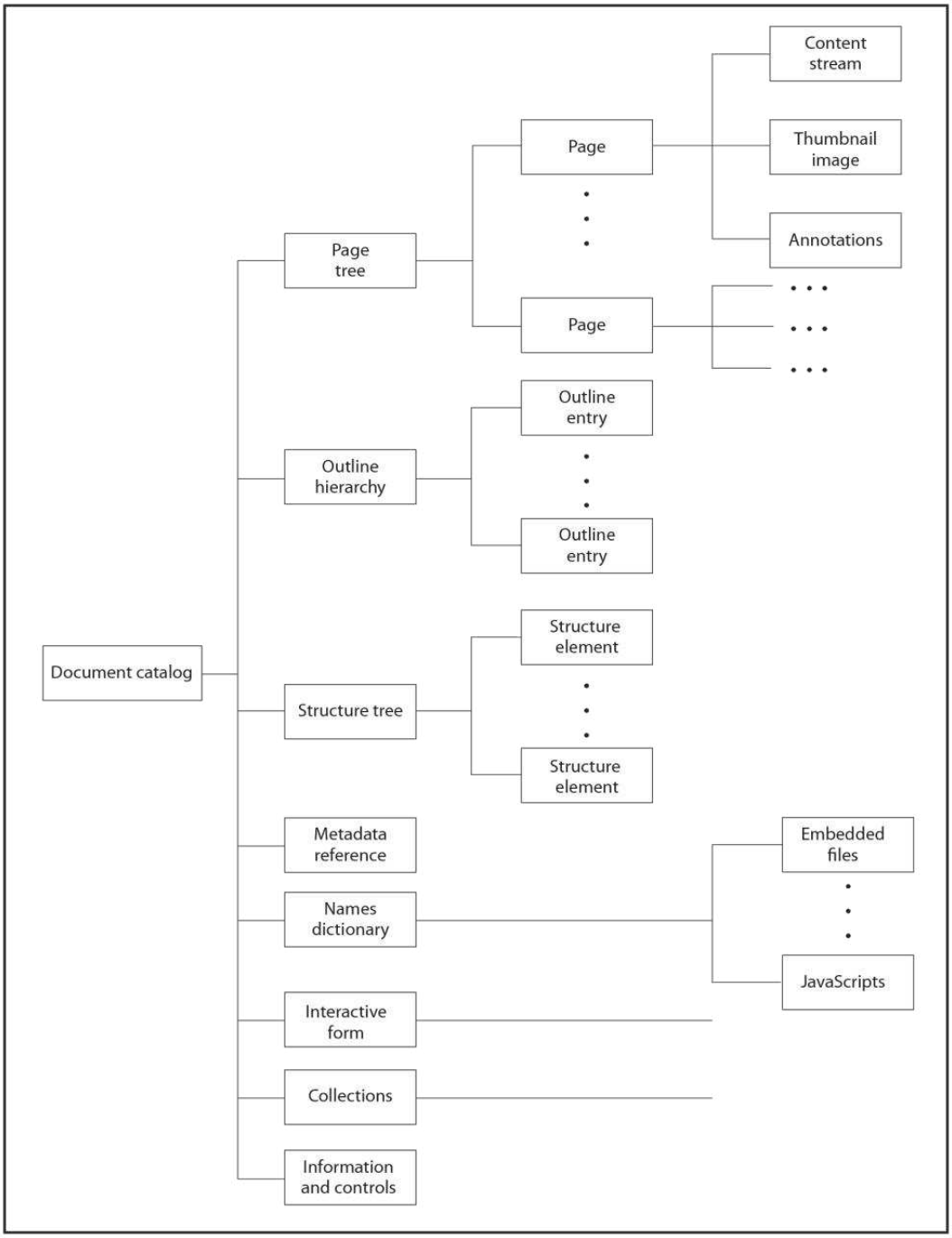
Nessa imagem, podemos observar, por exemplo, o *Document Catalog* na raiz e vários outros objetos relacionados, entre eles a raiz da *Page Tree*, que é uma estrutura de dados que representa as páginas do PDF e *Interactive Form*, que guarda informações relacionadas a um formulário contido no documento.

Já a estrutura do arquivo diz respeito a como esses objetos são serializados em disco. De acordo com ISO 32000-2 (2020), são especificadas 4 seções, como representado na Figura 5.

Na Seção *Header*, está contida a informação da versão da especificação utilizada. Em *Body* estão presentes os objetos serializados segundo suas sintaxes específicas. Após isso, a *Cross-reference table* guarda informações sobre como acessar qualquer objeto do documento de maneira randômica, registrando sua posição (deslocamento em bytes em relação ao começo do arquivo). Por fim, a seção *Trailer* contém um dicionário com metadados sobre o arquivo, incluindo a localização da **Cross-reference table**. No Apêndice A, há a reprodução de uma estrutura mínima de um documento PDF.

Adicionalmente, para a compreensão do mecanismo de assinatura, faz-se necessário entender a funcionalidade das atualizações incrementais. Elas permitem a modificação de um documento, sem a perda de qualquer informação prévia, característica análoga a de um sistema de versionamento.

Figura 4 – Estrutura do documento PDF.



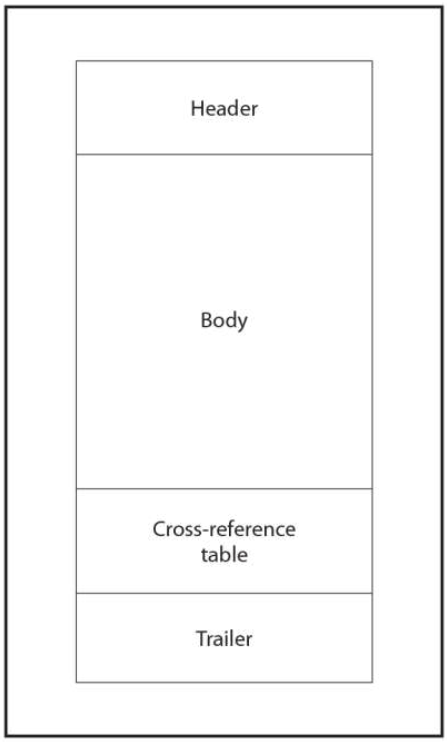
Fonte: ISO 32000-2 (2020, p. 98).

2.4.2 Atualizações incrementais

Para um bom funcionamento das assinaturas digitais, é importante que a assinatura possa ser feita sem modificar o conteúdo original do documento, visto que no padrão PAdES a assinatura é embutida dentro do arquivo. Isso é alcançado com as atualizações incrementais.

Essa técnica consiste em acrescentar novas seções **Body**, **Cross-reference** e **Trailer** no final do arquivo. Isso significa a adição de novos objetos. Como consequência dessa adição,

Figura 5 – Estrutura do arquivo PDF.



Fonte: ISO 32000-2 (2020, p. 54).

podem ser criadas novas páginas no documento por exemplo. Pode-se também, redefinir objetos já presentes no documento original. Nesse caso, os leitores de PDF sempre escolhem a versão mais recente do objeto, isto é, da atualização incremental mais recente.

O resultado desse modo de operação é que a leitura parcial do arquivo proporciona a recuperação da versão original do conteúdo. Ler a primeira atualização incremental sucede na exibição da primeira revisão do arquivo, e assim por diante, como mostra a Figura 6.

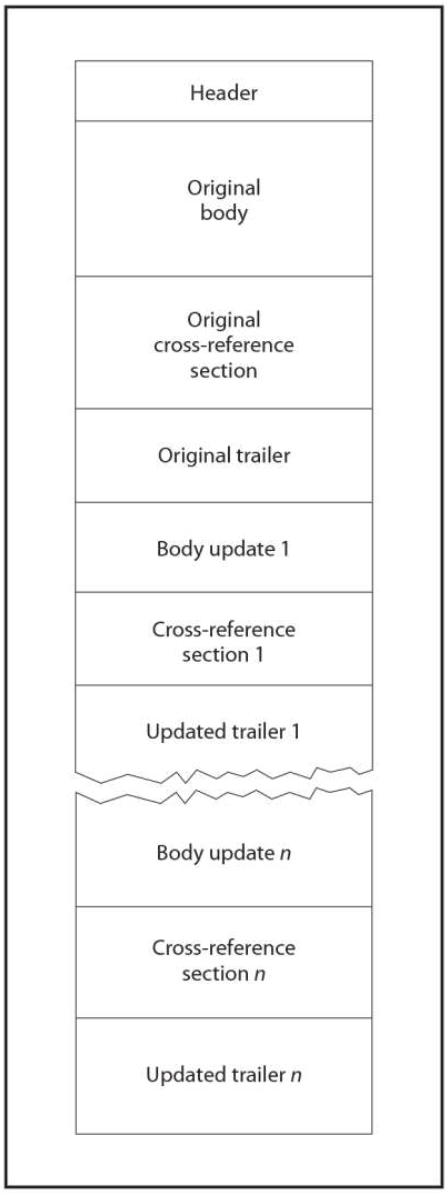
2.4.3 Objetos relacionados à assinatura digital

A seção 12.8 de ISO 32000-2 (2020) mostra como é possível embutir assinaturas digitais dentro de um documento. Para isso, deve-se acrescentar ou modificar alguns objetos ao documento, que serão descritos a seguir.

Dentro da estrutura do documento, as assinaturas digitais estão contidas em **Interactive Form**. Esse objeto é um dicionário que contém configurações do formulário presente no documento. Entre as chaves dele, temos **Fields**, que guarda uma lista de campos desse formulário.

Abaixo do formulário na hierarquia, há os objetos do tipo **Field**, que podem ter

Figura 6 – Atualizações incrementais.



Fonte: ISO 32000-2 (2020, p. 61).

vários subtipos, entre eles o **Signature Field**, que representa um campo de assinatura.

Já o objeto do tipo campo de assinatura tem como destaque uma chave chamada **V**, que aponta para outro objeto, do tipo **Signature**.

Em **Signature**, finalmente temos informações sobre a assinatura, como o algoritmo utilizado e o valor da assinatura em si, que será discutido na próxima seção.

Diante disso, conclui-se que é necessária a criação ou modificação desses três objetos: *Interactive Form*, *Signature Field* e *Signature Value* para embutir uma assinatura eletrônica em um PDF.

2.4.4 CMS (Cryptographic Message Syntax)

Definido pelo RFC 5652, essa estrutura de dados binária contém as informações necessárias para validar a assinatura digital, como a cadeia de certificados discutida anteriormente, Lista de Certificados Revogados e os atributos da assinatura.

A Lista de Certificados Revogados consiste em um rol que denota certificados que se tornaram inválidos por razões diversas, tais como fraude, óbito do detentor, ou encerramento de uma empresa. Essa lista é autenticada pela autoridade certificadora responsável. A falta de um certificado digital numa versão atualizada da lista tem implicações na validade do referido certificado naquele momento.

Em contrapartida, os atributos da assinatura são uma subestrutura que guarda metadados sobre a firma. Esses atributos podem incluir o motivo da assinatura em si, o momento em que ela ocorre, até mesmo o registro no Conselho Regional de Medicina de um médico, por exemplo, no caso de se tratar de um documento profissional, dentre demais variáveis. Tais atributos podem ser assinados junto com o documento ou não.

2.5 PDF Advanced Electronic Signature (PAdES)

O padrão PAdES consiste em restrições, extensões e orientações sobre as assinaturas digitais de documentos PDF. Ele versa sobre como preencher as estruturas vistas na Seção 2.4.3, sobre quais algoritmos e atributos são obrigatórios na estrutura CMS, como visto na Seção 2.4.4 e sobre quais às restrições relativas aos certificados digitais para a segurança da assinatura, como o algoritmo de assinatura.

Essa metodologia está descrita em ETSI TS 319 142-1 (2021) e serve de base para o padrão brasileiro, como descreve DOC-ICP-15 (2009).

2.5.1 Políticas de Assinatura

Nessa especificação, são estabelecidos níveis de conformidade para a assinatura. Esses níveis são estruturados em uma hierarquia, na qual os níveis superiores consistentemente mantêm conformidade com os níveis precedentes. Adicionalmente, para cada nível atingido, uma nova garantia é adquirida em termos da confiabilidade da assinatura.

Em DOC-ICP-15.03 (2021) por exemplo, são estabelecidos quatro principais Políticas de Assinatura:

Referência Básica - AD-RB. Padrão mínimo de assinatura, inclui o documento, alguns campos assinados, como a própria política de assinatura utilizada e o valor da assinatura.

Referência do Tempo - AD-RT. Além dos atributos básicos, inclui um carimbo do tempo assinado por uma Autoridade Certificadora, trazendo garantias sobre o momento da geração do documento.

Referências Completas - AD-RC. Nessa política é incluída uma cópia de todos os certificados da cadeia junto com seu estado de revogação, como descrito na Seção 2.4.4.

2.6 Trabalhos Relacionados

Existem 6 bibliotecas capazes de processar PDFs na coleção pública de pacotes da linguagem Ruby, como mostra a plataforma The Ruby Toolbox (2009), que agrega os pacotes da linguagem de maneira atualizada. A maior parte dessas soluções não implementa a assinatura digital de documentos. Outras bibliotecas estão sem receber atualizações há anos, algumas são pagas para fins comerciais e só uma implementa os perfis brasileiros de assinatura digital definidos em DOC-ICP-15.03 (2021).

Em função dessa escassez de bibliotecas de qualidade em Ruby, foram elencados os seguintes requisitos desejáveis no contexto de Aplicações *Web* para comparar os *softwares* existentes:

R1 - Implementação e dependências integralmente em Ruby. É preferível que o código seja nativo e não utilize interfaces com outras linguagens de programação para que haja maior portabilidade e facilidade de adoção.

R2 - Código fonte disponível para consulta. Liberar o código fonte para consulta permite a auditoria e melhoria do software por terceiros, que podem enviar propostas de melhorias ou correções.

R3 - Licença permissiva para fins comerciais. Torna a adoção do *software* mais fácil por empresas e indivíduos ao não incorrer em cobranças para sua utilização e também ao permitir a personalização de cópias.

R4 - Especialização em assinaturas digitais. O foco da maioria das ferramentas de manipulação de PDFs é em implementar o máximo possível do padrão ISO 32000-2 (2020), sendo a assinatura digital somente mais uma dentre diferentes funcionalidades. Com isso, esses *softwares* acabam se tornando complexos, além de não necessariamente focarem no recurso de assinatura digital, deixando de lado os tipos mais complexos de firma.

R5 - Implementação de perfis brasileiros de assinatura. Apesar de não ser obrigatório, o manual da plataforma Validar, do ITI, exige que os desenvolvedores priorizem o padrão PAdES com perfis de assinatura da ICP-Brasil (ITI, 2023).

R6 - Desenvolvimento ativo. Irá se considerar desenvolvimento ativo os *softwares* que tiveram lançamentos de novas versões no último ano e que suportem uma versão da linguagem Ruby que ainda tenha suporte. Esse critério é adotado porque ainda há muitos softwares utilizando bibliotecas desatualizadas, como Origami (2011), que segundo The Ruby Toolbox (2009), já acumula mais de um milhão e meio de downloads desde 07/10/2017, quando foi lançada a última versão do pacote.

Na Tabela 1, são elencadas as principais bibliotecas de assinatura digital de documentos e sua avaliação de acordo com os requisitos expostos.

Tabela 1 – Comparativo das principais das bibliotecas de assinatura digital em Ruby.

Biblioteca	R1	R2	R3	R4	R5	R6
HexaPDF	Sim	Sim	Não	Não	Não	Sim
Origami	Sim	Sim	Sim	Não	Não	Não
Chillkat	Não	Não	Não	Não	Sim	Sim
Apryse SDK	Não	Não	Não	Não	Não	Não
Rubrik	Sim	Sim	Sim	Sim	Sim	Sim

Fonte: elaborada pelo autor.

Somente a biblioteca Origami (2011) permite sua adoção em projetos comerciais sem haver a necessidade de adquirir uma licença ou de disponibilizar o código fonte do *software* comercial.

Outrossim, nenhuma ferramenta foca exclusivamente em assinaturas digitais. Entre implementação e testes, as bibliotecas Origami (2011) e HexaPDF (2016) tem respectivamente, 14470 e 46799 linhas de código de acordo com uma consulta feita pelo autor, além de 100 e 401 arquivos, respectivamente. Com isso, é possível notar que não são projetos triviais, requerendo bastante esforço de entendimento e manutenção.

Também é possível destacar que metade das bibliotecas não são mais atualizadas. Como dito anteriormente, o último lançamento de Origami (2011) foi no final de 2017, já a ferramenta Apryse SDK (1998) não menciona suporte à versão 3.0 da linguagem Ruby, a mais antiga ainda suportada.

Por último, somente o pacote Chilkat (2000) suporta os perfis de assinatura da ICP-Brasil. Porém ele só cumpre o requisito de estar em desenvolvimento ativo, disponibiliza

código fonte, não é gratuito para fins comerciais, não é especializado em assinaturas digitais e é totalmente feito em outras linguagens, sendo oferecidos somente uma interface para que seja possível acessar o *software* dentro da linguagem Ruby.

Desse modo, destaca-se a importância de Rubrik, uma biblioteca proposta nesse trabalho, que é permissiva para fins comerciais e especializada em assinaturas digitais, incluindo os perfis da ICP-Brasil.

3 RUBRIK: UMA BIBLIOTECA DE ASSINATURAS DIGITAIS

Neste capítulo, é abordado o planejamento do trabalho, incluindo detalhes sobre o cronograma de desenvolvimento. Além disso, é apresentado o funcionamento da biblioteca por meio da explicação das tarefas executadas pelo código, assim como sua organização em classes e módulos.

3.1 Planejamento

Visando disponibilizar uma solução de qualidade e que cumpra os requisitos elencados na Seção 2.6, foi idealizada a biblioteca Rubrik para a linguagem Ruby, com foco em assinaturas digitais avançadas.

A fim de disponibilizar o código fonte da biblioteca Rubrik e realizar o controle de versões, foi utilizada a plataforma GitHub (2008). Foi escolhida também a licença MIT para o projeto. Logo, o código fonte e os lançamentos do *software* podem ser encontrados em: https://github.com/tomascco/rubrik, cumprindo os requisitos (R2) e (R3).

Para cumprir com o requisito de focar somente em assinaturas digitais (R4), foi adotada uma estratégia que torna mínima a implementação necessária para o funcionamento do *software*. Algumas tarefas são delegadas à outras bibliotecas, desde que apresentem um nível mínimo de qualidade e sejam mantidas ativamente.

3.2 Descrição do projeto

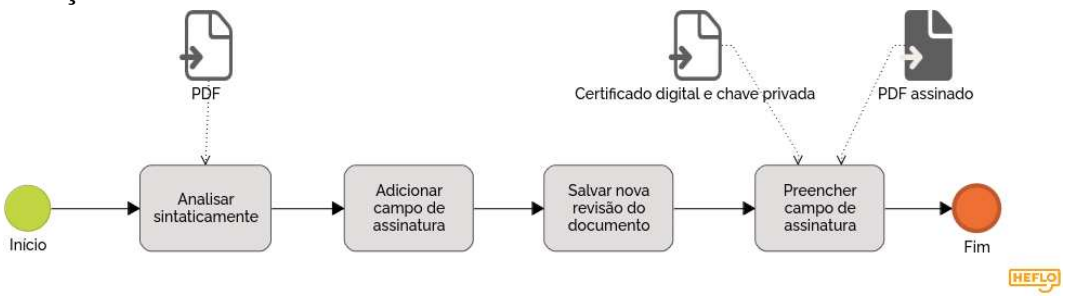
Nessa seção é apresentado em detalhes o funcionamento da biblioteca Rubrik, tanto de uma perspectiva de tarefas e processos, quanto da visão de organização do código.

3.2.1 Atividades

Rubrik funciona em alto nível em quatro passos, de acordo com a Figura 7. Esse fluxo é comum para todas as bibliotecas de assinatura digital e está especificado na Seção 12.8 de ISO 32000-2 (2020).

Nas seções seguintes é explicado como Rubrik realiza essas atividades.

Figura 7 – Diagrama de atividades do processo de assinatura digital utilizando a notação BPMN.



Fonte: Elaborado pelo autor.

3.2.1.1 Análise Sintática

No primeiro passo, é feito a análise sintática do documento pela biblioteca PDF Reader, convertendo seus elementos em sua Representação Intermediária (RI). Essa transformação segue o padrão do *software*, que se baseia na representação de objetos do PDF como estruturas de dados primitivas e objetos da linguagem Ruby.

Desse modo, a estrutura do documento pode ser lida e as mudanças necessárias para a adição de um novo campo de assinatura podem ser calculadas utilizando essa linguagem.

3.2.1.2 Adição dos campos de assinatura

Após o primeiro passo, é necessário criar ou atualizar vários objetos do documento para a inclusão de uma nova assinatura. Rubrik tem a capacidade de fazer esse tipo de alteração em PDFs. Ele aloca novos identificadores para os objetos criados com o intuito de serem referenciados corretamente no documento.

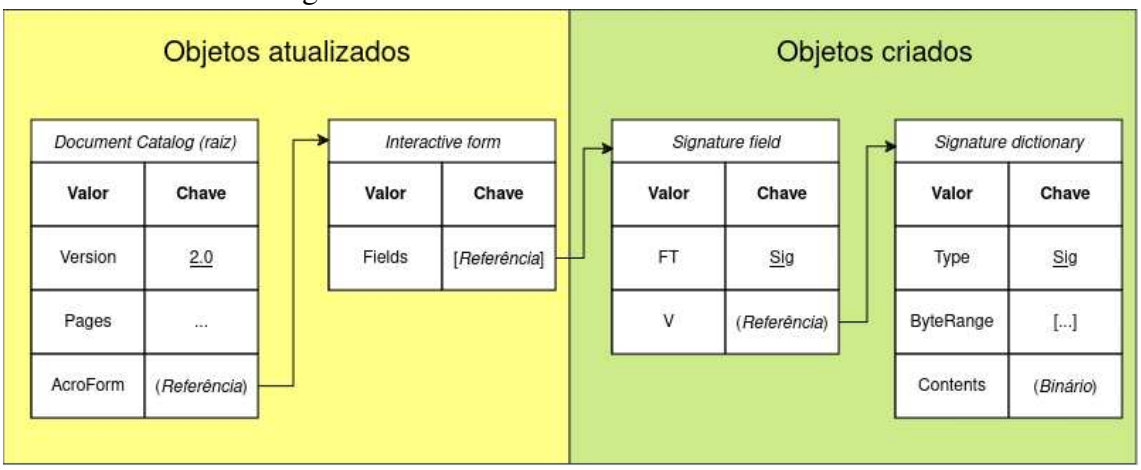
O procedimento para adicionar uma nova assinatura digital envolve a criação ou atualização do *Interactive form* para a inclusão de uma nova referência na lista de campos do formulário. Esse campo é um objeto novo do tipo *Signature field* e referencia um objeto *Signature dictionary*, que armazena informações de como validar a assinatura e o conteúdo dela. Isso está ilustrado na Figura 8.

3.2.1.3 Criação da nova revisão do documento

Nesse estágio,é codificada a nova versão do documento, contendo as modificações aplicadas.

Inicialmente, o documento original é replicado integralmente em um novo arquivo.

Figura 8 – Representação simplificada dos objetos modificados no processo de adicionar uma nova assinatura digital



Fonte: Elaborado pelo autor.
Nota: Cada tabela representa um objeto do tipo dicionário, valores sublinhados representam símbolos e colchetes representam listas.

Posteriormente, conforme as diretrizes estabelecidas por ISO 32000-2 (2020), os novos objetos são serializados de sua representação intermediária em *Ruby* para a sintaxe PDF. Por fim, o documento é concluído com a criação de uma nova *Cross Reference Table* e de um novo *Trailer*.

3.2.1.4 Preenchimento do campo de assinatura recém-criado

Por último, a nova revisão é utilizada para calcular a assinatura digital, e o documento é gravado no disco.

Este processo envolve a leitura do arquivo para extrair os dados a serem assinados e a criação do envelope CMS contendo a assinatura.

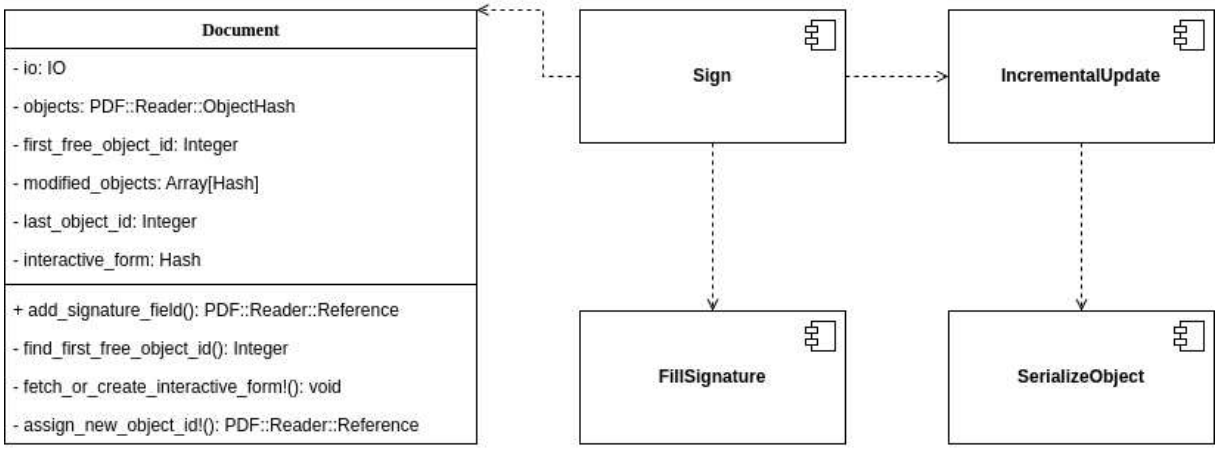
3.2.2 Dependências

- PDF Reader (2007): provê a funcionalidade de analisador sintático para PDFs, convertendo seus componentes para estruturas de dados primitivas da linguagem Ruby, que são usados como RI para a reconstrução dos documentos assinados posteriormente. Apesar de a biblioteca desempenhar uma função central na Rubrik, ela tem licença permissiva, o que permite a sua incorporação em Rubrik, caso seja descontinuada.
- OpenSSL for Ruby (2002): a única parte não nativa da biblioteca, esse pacote fornece a implementação de algoritmos criptográficos, como o RSA.

3.2.3 Organização do código

O Código fonte da biblioteca está organizado em uma classe e cinco componentes, expressos na Figura 9. A responsabilidade de cada unidade está descrita nas seções subsequentes.

Figura 9 – Diagrama dos componentes de Rubrik.



Fonte: Elaborado pelo autor.
Nota: as ligações indicam relações de dependência.

3.2.3.1 Document

No construtor desta classe, é possível passar qualquer objeto que adote a interface IO da linguagem Ruby, proporcionando a capacidade de entrada de arquivos em disco ou carregados em memória. Essa classe representa um arquivo PDF carregado pela Rubrik e tem a responsabilidade de gerenciar o estado das alterações que serão aplicadas ao documento.

Para alcançar esse objetivo, a classe incorpora a variável privada de instância *objects*, que é uma instância de *PDF::Reader::ObjectHash*, uma classe da biblioteca PDF Reader. Essa classe oferece acesso a qualquer objeto no documento digital através da especificação de seu ID, assemelhando-se a um dicionário. Ao fornecer um identificador, o objeto recuperado é analisado sintaticamente e convertido para os tipos primitivos da linguagem Ruby pelo PDF Reader, incluindo dicionários, *strings* e tipos numéricos.

Já os objetos que serão instanciados ou atualizados na criação da nova revisão do documento estão contidos na variável de instância pública *modified_objects*, que é uma *array* que aceita a mesma representação utilizada pela biblioteca PDF Reader.

Finalmente, o método público *add_signature_field* desempenha a responsabilidade de incorporar os objetos essenciais para a realização da assinatura digital.

3.2.3.2 IncrementalUpdate

Este componente converte uma instância de *Document* em um arquivo PDF, incorporando a nova revisão por meio da adição dos campos especificados na propriedade *modified_objects* de uma instância de *Document*.

O componente começa copiando integralmente o conteúdo do arquivo original para um novo arquivo. Em seguida, os novos objetos são serializados dentro do documento, e suas posições iniciais em bytes no novo arquivo são registradas para a construção subsequente da nova *cross reference table*.

Após a criação da nova tabela, também é escrito o novo *trailer*.

3.2.3.3 SerializeObject

Este componente é responsável por realizar o processo inverso da biblioteca *PDF Reader*, transformando a representação intermediária feita com tipos primitivos da linguagem Ruby em objetos PDF, de acordo com a Seção 7.3 de ISO 32000-2 (2020), que aborda a sintaxe desses tipos.

3.2.3.4 FillSignature e Signature

FillSignature desempenha a função de preencher o valor da assinatura no documento, registrando no arquivo tanto o valor da assinatura quanto os intervalos em bytes que foram assinados.

A computação da assinatura em si é delegada para implementações da interface *Signature*. Até o presente momento, a única implementação dessa interface, e consequentemente, o único tipo de assinatura disponível, é do tipo Cryptographic Message Syntax (PKCS #7).

3.2.3.5 Sign

Este módulo representa o ponto principal de entrada da biblioteca, sendo responsável por orquestrar a utilização de outras classes e objetos do software para a realização da assinatura digital de um PDF. Seus parâmetros de entrada consistem em dois objetos que implementam a interface IO da linguagem Ruby: um para a entrada do documento digital e outro para gravar o documento assinado.

Adicionalmente, o método requer o certificado digital, a chave privada associada a

esse certificado e, opcionalmente, a cadeia de certificados a serem incorporados no envelope da assinatura.

3.3 Utilização de Rubrik

Em síntese, é possível instalar Rubrik utilizando o gerenciador de pacotes da linguagem Ruby, o RubyGems.org (2004). Com o pacote instalado, é possível desenvolver *scripts*, utilitários de linha de comando e aplicações *web*. Na documentação disponibilizada junto com o código fonte da biblioteca existem exemplos de como utilizar o *software*. No capítulo seguinte, serão apresentados testes feitos com um pequeno *software* de exemplo desenvolvido com Rubrik.

4 TESTES DE VERIFICAÇÃO

Para testar a biblioteca de assinatura digital desenvolvida, foram definidas três estratégias com o objetivo de garantir que os objetivos definidos foram atingidos: testes automatizados, teste em massa com documentos encontrados em outros softwares livres e testes manuais com a plataforma Validar (2023).

4.1 Testes automatizados

Foi elaborada uma suíte de testes automatizados para a biblioteca utilizando o software minitest (2006), que é uma ferramenta que disponibiliza recursos de teste para a linguagem Ruby. Esse conjunto é constituído de 26 cenários de testes, incluindo testes de unidade para todos os componentes citados na Seção 3.2.3 e também três cenários de testes que exercitam todos os módulos da biblioteca, simulando o uso por um usuário final do *software*.

Nesse contexto, testes de unidade são aqueles que verificam as menores partes de um *software* individualmente (IBM, 2017).

A suíte de testes elaborada executa 100% do código-fonte de Rubrik, além de percorrer todos os caminhos de execução possíveis, passando pelo menos uma vez tanto pelo ramo verdadeiro quanto pelo falso de cada condicional. O resultado de uma execução do conjunto e o relatório resumido de cobertura gerado pela suíte com o *software* SimpleCov (2010) foram reproduzidos na Figura 10 e na Tabela 2.

Figura 10 – Resultado da execução da suite de testes automatizados de Rubrik.

```
Run options: --seed 55234

# Running:

.....................

Fabulous run in 0.044245s, 474.6257 runs/s, 5492.0971 assertions/s.

21 runs, 243 assertions, 0 failures, 0 errors, 0 skips
```

Fonte: Elaborado pelo autor.

A partir da execução da suíte, podemos observar que a sequência de execução dos cenários é pseudo-aleatória e depende do valor de *seed*. Além disso, destaca-se a eficiência do conjunto, que pode ser executado até 475 vezes por segundo em um computador com processador AMD Ryzen 7 5700G, de 8 núcleos, 16 *threads* e 16 *gigabytes* de memória principal. Já a Tabela

Tabela 2 – Relatório da cobertura gerada com a execução da suíte de testes automatizados.

Arquivo	Cobertura	Linhas	Cobertura de Ramos	Ramos
lib/rubrik.rb	100%	10	100%	0
lib/rubrik/document.rb	100%	70	100%	6
lib/rubrik/document/increment.rb	100%	43	100%	2
lib/rubrik/document/serialize_object.rb	100%	25	100%	13
lib/rubrik/fill_signature.rb	100%	37	100%	0
lib/rubrik/pkcs7_signature.rb	100%	9	100%	0
lib/rubrik/sign.rb	100%	11	100%	2

Fonte: elaborada pelo autor.

2 mostra a estrutura de arquivos da biblioteca, linhas de código e cobertura de cada arquivo. Como dito anteriormente, foram atingidos 100% de cobertura de linhas e de ramos em todos os arquivos. Apesar de esse fato sozinho não garantir a qualidade do *software*, a junção dele com a quantidade de testes têm correlação com a efetividade da suíte, como concluem Inozemtseva e Holmes (2014).

Por outro lado, é importante ressaltar que o objetivo dos testes é assegurar a corretude da biblioteca Rubrik, abrangendo a manipulação e a integração bem-sucedida da assinatura digital em arquivos PDF. Vale destacar que, para garantir a validade do documento assinado por terceiros, é necessário utilizar um certificado digital reconhecido por estes e sua chave privada correspondente. No Brasil, um exemplo de certificado reconhecido nacionalmente são os pertencentes à cadeia ICP-Brasil.

Porém, isso não é possível para uma biblioteca com código-fonte livre, já que a disponibilização de uma chave privada pode levar ao roubo de identidade. Para contornar este problema, foi disponibilizado um certificado digital sem nenhuma validação de identidade, somente para que seja possível executar os testes. Esse certificado foi gerado utilizando a biblioteca OpenSSL for Ruby (2002).

4.2 Testes manuais utilizando a plataforma Validar

Devido às limitações inerentes aos testes automatizados para assegurar a validade do documento final perante terceiros, foi formulada a estratégia de testar uma quantidade reduzida de arquivos, provenientes de fontes comuns, por meio de avaliação manual. Durante essa análise, a validade das assinaturas é verificada utilizando a plataforma Validar (2023), do ITI.

No processo de seleção das fontes dos PDFs, conduziu-se uma pesquisa nas categorias mais comuns de softwares que geram arquivos PDF, optando por alguns dos programas

mais populares de cada categoria. A maioria dos arquivos utilizados nos testes não foram feitos para testar a biblioteca, mas são exemplos de aulas, trabalhos, documentos e outros trabalhos acadêmicos.

Após a seleção dos documentos, todos os arquivos foram assinados digitalmente utilizando o Rubrik e enviados para a plataforma Validar (2023). Os resultados da validação das assinaturas dos documentos estão apresentados na Tabela 3, que também inclui os softwares utilizados e suas categorias. Adicionalmente, todos os documentos originais, os documentos assinados e os relatórios de conformidade gerados pela plataforma Validar estão disponíveis no Google Drive.[1]

Tabela 3 – Resultado da validação manual com a plataforma Validar.

Software	Categoria	Resultado da validação
Adobe Acrobat Online	Leitor e editor de PDFs	✓
iLovePDF (Conversor imagem para PDF)	Editor de PDFs	✓
Canva	Design gráfico	✓
Google Chrome (imprimir página como PDF)	Navegador	✓
Mozilla Firefox (imprimir página como PDF)	Navegador	✓
Overleaf	Editor de LaTex	✓
Google Docs	Processador de texto	✓
LibreOffice Writer	Processador de texto	✓
Word (Microsoft 365)	Processador de texto	✓
Google Sheets	Processador de planilhas	✓
LibreOffice Calc	Processador de planilhas	✓
Excel, documento 1 (Microsoft 365)	Processador de planilhas	X*
Excel, documento 2 (Microsoft 365)	Processador de planilhas	✓
Google Slides	Processador de apresentações	✓
LibreOffice Impress	Processador de apresentações	✓
PowerPoint (Microsoft 365)	Processador de apresentações	✓

Fonte: elaborada pelo autor.
Nota: O arquivo marcado com '*' não pode ser assinado devido a um erro de leitura do documento por parte de Rubrik.

Dessa forma, ao analisar a Tabela 3, observa-se que a biblioteca alcançou com sucesso o objetivo de assinar documentos provenientes de fontes comuns. Entretanto, foi necessário corrigir um problema identificado na realização desses testes ao assinar documentos gerados pelo Google Sheets e Google Slides. Isso destaca a relevância dos testes manuais em conjunto com os testes automatizados.

É relevante ressaltar que a falha ao processar o Documento 1 gerado pelo Excel sugere uma falta de tolerância a erros por parte do Rubrik. Embora o PDF em questão apresente inconformidades com a especificação técnica, sua leitura foi bem-sucedida em diversos outros

[1] Link para os resultados: https://drive.google.com/drive/folders/17Q6B8ERMvKT_ElTJqvhcuX6v9TsWw5uu

softwares. A tolerância a falhas é fundamental ao lidar com documentos digitais, dado que o formato de arquivo PDF é o mais popular na internet, resultando em uma probabilidade significativa de encontrar arquivos com grande variedade de erros.

Por fim, dado que esses testes são conduzidos manualmente, a viabilidade de avaliar numerosos arquivos provenientes de diversos geradores de documentos é limitada. Isso impacta a eficácia dos testes, uma vez que, testando apenas 16 documentos, foram identificados e corrigidos dois erros na biblioteca Rubrik.

Considerando as limitações das estratégias de testes descritas nas seções 4.1 e 4.2, foi concebida a estratégia de testes em massa.

4.3 Teste em massa

Essa abordagem envolve a pesquisa pelo maior número possível de PDFs provenientes de diversas fontes, a assinatura desses documentos com o Rubrik e a execução de softwares externos para validar a integridade dos documentos assinados. O objetivo é obter uma compreensão mais abrangente da taxa de PDFs que o Rubrik pode assinar, considerando suas limitações, por meio dessa amostragem.

Os documentos mencionados anteriormente foram retirados da suíte de testes da biblioteca PDF.js, que é o *software* responsável pela visualização de arquivos PDF dentro do navegador Mozilla Firefox. Sendo assim, esses arquivos são bastante diversificados e abrangem uma grande parte da especificação ISO 32000-2 (2020). Eles incluem arquivos íntegros, bem como muitos outros com diversos tipos de erros de sintaxe e corrupções, representando, assim, um conjunto desafiador para a biblioteca Rubrik. Foram selecionados, aleatoriamente, 645 PDFs desse conjunto.

Para verificar a integridade dos arquivos após o processo de assinatura com Rubrik, foram empregadas duas ferramentas distintas: *pdfinfo* e *qpdf*.

pdfinfo é uma utilidade de linha de comando que analisa um documento digital e fornece informações sobre ele. Este programa é parte da biblioteca Poppler (2005), um software cuja finalidade é renderizar PDFs, sendo desenvolvido pela freedesktop.org.

Por outro lado, o *qpdf* é uma ferramenta destinada a aplicar transformações em arquivos PDF que preservem seu conteúdo, como a união de documentos distintos. Essa ferramenta também oferece diversas utilidades para analisar a estrutura do arquivo.

Ademais, dado que vários dos documentos utilizados apresentam erros e corrupções

a priori, as ferramentas *pdfinfo* e *qpdf* também foram utilizadas nos arquivos originais para comparar com o resultado após a assinatura. O resultado desses testes está resumido na Tabela 4.

Tabela 4 – Resultado da análise de integridade dos arquivos originais por *qpdf* e *pdfinfo*

Situação	*qpdf*	*pdfinfo*
Sucessos	428 (66%)	630 (98%)
Falhas	217 (34%)	15 (2%)
Total	645 (100%)	645 (100%)

Fonte: elaborada pelo autor.

Analisando os resultados de *qpdf*, é possível observar uma taxa de erro elevada antes de qualquer processamento por parte de Rubrik. Isso se deve, em grande parte, à elevada exigência de *qpdf* na análise dos documentos. Foi possível constatar essa situação ao observar que, os arquivos considerados falhos pela análise podem ser abertos sem nenhum problema por leitores de PDF. Observar a capacidade de abrir um arquivo em leitores de documentos é importante, uma vez que, se o arquivo não apresenta problemas perceptíveis para o usuário, outras ferramentas de edição também deveriam ser capazes de processar esses arquivos.

Ao analisar os resultados de *pdfinfo*, percebe-se que ele atende o critério descrito anteriormente, pois apenas 15 arquivos não passaram no teste utilizando o *software*, e, de fato, não foi possível abri-los utilizando leitores de PDF.

Após as análises anteriores, todos os documentos de teste foram assinados por Rubrik. A descrição dos resultados das assinaturas está exposto na Tabela 5.

Tabela 5 – Resultado do processamento dos PDFs de teste por Rubrik

Situação	Quantidade de documentos
Processamento sem erros	611 (95%)
Erros de leitura por parte de Rubrik	29 (4,5%)
Motivo desconhecido	1 (0,1%)
PDFs com senha	4 (0,4%)
Total	645 (100%)

Fonte: elaborada pelo autor.

Ao observar a Tabela 5, é possível destacar uma taxa de erro de 5%. Dentre esses erros, quase todos foram causados por falhas de Rubrik ao realizar a análise sintatica dos documentos. Isso evidencia a necessidade de aprimorar o analisador sintático de Rubrik, o

PDF Reader (2007), que é uma dependência de código fonte aberto, como uma melhoria futura. Outrossim, Considerando o total de documentos sem defeitos à priori como sendo 630 (resultado de *pdfinfo*), a taxa de sucesso aumenta para 97%.

Por fim, os 611 documentos assinados foram submetidos novamentes às ferramentas *qpdf* e *pdfinfo*. Os resultados estão descritos a seguir na Tabela 6.

Tabela 6 – Resultado da análise de integridade utilizando *qpdf* e *pdfinfo* dos documentos assinados por Rubrik

Situação	*qpdf*	*pdfinfo*
Sucessos	518 (85%)	607 (99,3%)
Falhas	93 (15%)	4 (0,7%)
Total	611 (100%)	611 (100%)

Fonte: elaborada pelo autor.

Da análise da Tabela 6, destaca-se um resultado surpreendente de após o processamento de Rubrik, a taxa de sucesso com a ferramenta *qpdf* teste aumentou. Isso não significa que Rubrik altera o conteúdo original do PDF ou que faça alguma correção, mas sim ao fato do mecanismo de atualizações incrementais utilizado pela biblioteca torna redundante o processo de linearização desses documentos, que era o motivo de sua falha nos testes. PDFs linearizados são uma funcionalidade opcional para os documentos, que tem como objetivo otimizá-los para transmissão via rede, fazendo com que não seja necessário baixar o arquivo inteiro para acessar uma página específica, por exemplo. (ISO 32000-2, 2020)

Ao analisar os resultados de *pdfinfo*, percebe-se que houveram quatro falhas. Os documentos resultantes também não puderam ser lidos em leitores PDF. Entre essas falhas, há três documentos que foram reprovados por *pdfinfo*, mas que foram assinados com sucesso por Rubrik e continuam apresentando falhas após a assinatura. Não foi possível atribuir uma causa à última falha, que decorreu de um arquivo anteriormente aprovado por *pdinfo*.

4.4 Considerações finais

Os resultados obtidos com o Rubrik nos testes foram suficientes para apontar as deficiências do *software*, especialmente no que diz respeito à análise sintática dos documentos, que foi a principal causa dos erros observados.

Essa imperfeição se deu pelo uso de PDF Reader (2007) como analisador sintático

pré-existente, que foi feito para acelerar o desenvolvimento. Porém, seu código fonte é aberto, podendo ser incorporado e melhorado em Rubrik como trabalho futuro.

Em resumo, Rubrik não apresenta problemas ao lidar com documentos sem erros. No entanto, para garantir seu funcionamento em todos dos casos, é recomendável a realização de uma etapa prévia de pré-processamento para corrigir eventuais problemas nos documentos que poderiam causar erros em Rubrik e outros *softwares* semelhantes.

5 CONCLUSÕES E TRABALHOS FUTUROS

Muitas empresas brasileiras não utilizam assinaturas digitais, mas pretendem adotar essa tecnologia, dada suas vantagens em relação às assinaturas manuais. A construção de *softwares* capazes de suprir essa necessidade pode ser favorecida por bibliotecas que manipulam PDFs. Rubrik foi criada para suprir essa necessidade na linguagem Ruby.

O *software* foi desenvolvido utilizando um analisador sintático preexistente, que possibilitou Rubrik ler documentos digitais. Após isso, modifica-se a estrutura retornada pelo analisador sintático para incluir os elementos da assinatura digital. Então, OpenSSL for Ruby (2002) gera a estrutura binária CMS, já contendo a firma feita com o certificado digital fornecido. Por último, um módulo serializador salva o arquivo já com a assinatura embutida.

Como resultado, foi possível atestar a aderência da assinatura digital ao padrão brasileiro utilizando a plataforma Validar (2023), o que garante validade jurídica aos documentos assinados com Rubrik. Ademais, a conformidade sintática dos PDFs gerados pelo *software* ao padrão ISO 32000-2 (2020) também foi testada por *qpdf* e *pdfinfo*, com boas taxas de sucesso. Por último, a suíte de testes automatizados da biblioteca garantem uma proteção contra a introdução de defeitos em versões futuras.

Desse modo, Rubrik pode ser utilizado junto com uma etapa anterior de pré-processamento dos documentos para gantir seu funcionamento em 100% dos casos, visto a maioria de suas falhas nos testes decorreram de problemas na análise sintática.

Outrossim, durante o desenvolvimento foram percebidos vários erros em Rubrik, que foram prontamente corrigidos. Porém, isso sugere a necessidade de validar Rubrik em *softwares* em cenários reais.

Apenas o perfil de assinatura mais básico, que oferece menos garantias, está disponível. Um exemplo disso é que a assinatura implementada não oferece garantias de tempo sobre quando o documento foi criado ou assinado.

Além disso, a biblioteca depende de código escrito em C, o OpenSSL for Ruby (2002), e não está completamente codificado em Ruby, como se pretendia inicialmente. A pesquisa e a implementação dos padrões necessários para cobrir estas limitações requerem significativo esforço de desenvolvimento, razão pela qual foram deixadas para a continuação em trabalhos futuros.

A partir dos resultados alcançados das limitações do trabalho, elencam-se sugestões para trabalhos futuros:

- Adicionar suporte para PDFs criptografados;
- Incorporação da biblioteca PDF Reader (2007) e a implementação de melhorias na análise sintática;
- Possibilitar o usuário especificar uma aparência ou carimbo para a assinatura;
- Adicionar suporte para o fluxo de assinaturas externas, isso é, fazer com que Rubrik apenas transmita os dados a serem assinados para um processo ou hardware externo, que irá computar e retornar a assinatura a ser embutida no documento;
- Suportar a utilização de perfis de assinatura mais avançados;
- Adicionar suporte a perfis de assinatura da União Europeia;
- Testar Rubrik em sistemas em produção e analisar seu desempenho.

REFERÊNCIAS

Apryse SDK. **Customizable PDF functionalities for all platforms**. 1998. https://apryse.com/products/core-sdk. (Acessado em 31/10/2023).

BRASIL. Lei nº 14.063, de 23 de setembro de 2020. **Diário Oficial [da] República Federativa do Brasil**, Brasília, DF, 2020. ISSN 1677-7042. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2019-2022/2020/lei/l14063.htm.

BUCHMANN, J.; DAHMEN, E.; SZYDLO, M. Hash-based digital signature schemes. In: ____. **Post-Quantum Cryptography**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 35–93. ISBN 978-3-540-88702-7. Disponível em: https://doi.org/10.1007/978-3-540-88702-7_3.

Chilkat. **Chilkat API, SDK, Components, Libs for Windows, MacOS, Linux, iOS, Android, and more**. 2000. https://www.chilkatsoft.com/. (Acessado em 31/10/2023).

Common Crawl Foundation. **MIME Types**. 2023. https://commoncrawl.github.io/cc-crawl-statistics/plots/mimetypes. (Acessado em 24/02/2024).

DOC-ICP-15. **VISÃO GERAL SOBRE ASSINATURAS DIGITAIS NA ICP-BRASIL**. Brasília, BR, 2009. Disponível em: https://www.gov.br/iti/pt-br/central-de-conteudo/doc-icp-15-v-1-0-pdf.

DOC-ICP-15.03. **REQUISITOS DAS POLÍTICAS DE ASSINATURA DIGITAL NA ICP BRASIL: Versão 8.0**. Brasília, BR, 2021. Disponível em: https://www.gov.br/iti/pt-br/assuntos/legislacao/instrucoes-normativas/IN032021_DOC_15.03_assinada.pdf.

DocuSign. **O que é assinatura digital? Tire todas as suas dúvidas!** 2020. https://www.docusign.com/pt-br/blog/o-que-e-assinatura-digital. (Acessado em 27/03/2024).

ETSI TS 319 142-1. **Electronic Signatures and Infrastructures (ESI); PAdES digital signatures; Part 1: Building blocks and PAdES baseline signatures**. [*S. l.*], 2021. v. 2016.

GitHub. **Let's build from here**. 2008. https://github.com/. (Acessado em 09/11/2023).

GOV.UK. **GOV.UK Developer docs - Conventions for Rails applications.** 2024. https://docs.publishing.service.gov.uk/manual/conventions-for-rails-applications.html. (Acessado em 21/08/2024).

HexaPDF. **The versatile PDF creation and manipulation library for Ruby**. 2016. https://gettalong.at/hexapdf/. (Acessado em 31/10/2023).

IBM. **What is software testing?** 2017. https://www.ibm.com/topics/software-testing. (Acessado em 28/11/2023).

INOZEMTSEVA, L.; HOLMES, R. Coverage is not strongly correlated with test suite effectiveness. In: **Proceedings of the 36th International Conference on Software Engineering**. New York, NY, USA: Association for Computing Machinery, 2014. (ICSE 2014), p. 435–445. ISBN 9781450327565. Disponível em: https://doi.org/10.1145/2568225.2568271.

ISO 32000-2. **Document management — Portable document format — Part 2: PDF 2.0**. Geneva, CH, 2020. Disponível em: https://www.iso.org/standard/75839.html.

ITI. **Guia de Orientações aos Desenvolvedores**. 2023. https://validar.iti.gov.br/guia-desenvolvedor.html. (Acessado em 30/10/2023).

ITI. **ICP - Brasil**. 2024. https://estrutura.iti.gov.br/. (Acessado em 27/03/2024).

KATZ, J. **Digital signatures**. [*S. l.*]: Springer, 2010. v. 1.

Kawasaki, Takahiko. **Illustrated X.509 Certificate**. [*S. l.*], 2020. (Acessado em 27/03/2024). Disponível em: https://darutk.medium.com/illustrated-x-509-certificate-84aece2c5c2e.

minitest. **Provides a complete suite of testing facilities supporting TDD, BDD, mocking, and benchmarking.** 2006. https://github.com/minitest/minitest/. (Acessado em 23/11/2023).

OpenSSL for Ruby. **Provides SSL, TLS and general purpose cryptography.** 2002. https://github.com/ruby/openssl/. (Acessado em 31/10/2023).

Origami. **Pure Ruby library to parse, modify and generate PDF documents**. 2011. https://github.com/gdelugre/origami. (Acessado em 31/10/2023).

PDF Reader. **Implements a PDF parser conforming as much as possible to the PDF specification from Adobe**. 2007. https://github.com/yob/pdf-reader/. (Acessado em 31/10/2023).

Poppler. **Poppler is a PDF rendering library based on the xpdf-3.0 code base.** 2005. https://poppler.freedesktop.org/. (Acessado em 21/08/2024).

Ruby on Rails. **Ruby on Rails — A web-app framework that includes everything needed to create database-backed web applications according to the Model-View-Controller (MVC) pattern.** 2004. https://rubyonrails.org/. (Acessado em 06/14/2023).

RubyGems.org. **Find, install, and publish RubyGems.** 2004. https://rubygems.org/. (Acessado em 21/08/2024).

SEBRAE SP. **Gerenciamento de contratos pelos pequenos negócios**. 2023. https://datasebrae.com.br/wp-content/uploads/2023/10/Gerenciamento-de-contratos-pelas-MPEs-vf.pdf. (Acessado em 27/03/2024).

SimpleCov. **Code coverage for Ruby with a powerful configuration library and automatic merging of coverage across test suites.** 2010. https://github.com/simplecov-ruby/simplecov/. (Acessado em 23/11/2023).

The Ruby Toolbox. **Find actively maintained & popular open source software libraries for the Ruby programming language**. 2009. https://www.ruby-toolbox.com/. (Acessado em 24/10/2023).

Validar. **Serviço de validação de assinaturas eletrônicas**. 2023. https://validar.iti.gov.br/. (Acessado em 30/10/2023).

APÊNDICE A – ARQUIVO PDF MÍNIMO

Código-fonte 1 – Arquivo PDF

```
%PDF-1.7
1 0 obj
<<
  /Type /Catalog
  /Pages 2 0 R
>>
endobj

2 0 obj
<<
  /Type /Pages
  /Kids [3 0 R]
  /Count 1
>>
endobj

3 0 obj
<<
  /Type /Page
  /MediaBox [0 0 595.44 841.68]
>>
endobj

xref
0 4
0000000000 65535 f
0000000009 00000 n
0000000063 00000 n
0000000127 00000 n
```

```
30 trailer
31 <<
32   /Size 4
33   /Root 1 0 R
34 >>
35 startxref
36 195
37 %%EOF
```