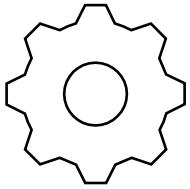
Autor:

Rodrigo Coutinho Corrêa



Documento de Requisitos:
Teste Uppertools

Criado Em:

17/12/2020

Sumário

1. Introdução

Este documento visa fornecer as informações necessárias para desenvolver e manter a solução Teste Uppertools, este solução abrange dois sistemas, um serviço Windows e uma aplicação do tipo formulário Windows.

2. Visão geral do produto

Para o funcionamento dos sistemas de gestão dos clientes, será necessária a importação de dados e quando estes dados já são disponíveis em plataformas digitais, obter e armazenar esses dados da forma mais automatizada possível pode gerar economia de esforço, tempo e dinheiro.

A solução Teste Uppertools deverá obter as informações contidas nas faturas de cada fornecedor, recebidas por um e-mail específico, considerando diferentes formatos de arquivo e diferentes layouts.

Ele irá funcionar como uma aplicação do tipo console, utilizando DOT.NET Core 3.0, podendo ser facilmente implementado com um serviço Windows, para leitura de e-mails e processamento das faturas. Para cadastros de layout de arquivos e fornecedores, ele deverá utilizar uma aplicação Windows Forms com DOT.NET Framework 4.7.2.

3. Abreviações e acrônimos

CRUD: Create, Read, Update e Delete, que define as operações básicas ao manter registros no sistema ou no banco de dados.

PDF: Portable Digital Document, formato de arquivo digital.

ODS: OpenDocument Spreadsheet, formato de arquivo digital.

CSV: Character-separated values, formato de arquivo digital.

JSON: JavaScript Object Notation, formato de arquivo digital.

XML: eXtensible Markup Language, formato de arquivo digital.

4. Stakeholders

4.1. Fornecedor

O fornecedor tem um papel importante na solução proposta pois quanto menor número de layout de fatura, menor o esforço para o recebimento automatizado dos dados, ou seja, é desejável uma padronização desses documentos. Deve-se garantir também o envio das faturas a serem contabilizadas para um único endereço de e-mail do cliente, este e-mail será monitorado pelo solução para automatização do restante do processo e alimentação do banco de dados.

4.2. Cliente

O cliente terá a responsabilidade de dedicar um usuário para monitorar possíveis alertar emitidos pelo serviço, apontando layouts com problema, efetuando o reparo ao utilizar o sistema desktop para reposicionar os campos e/ou linhas esperadas nos arquivos formatados ou cadastrando novos layout com configuração da fatura esperada.

5. Requisitos (Funcionais, Não-Funcionais e Regras de Negócio)

5.1. Requisitos Funcionais

REQ 01: O sistema deverá conhecer os remetentes seguros para percorrer e-mails não lidos na caixa de entrada.

REQ 02: Se for encaminhado um link para download, o sistema deverá realizar o download através deste link. Se for encaminhado como anexo o sistema deverá realizar o download dele. Destinar uma thread a essa funcionalidade.

REQ 03: Percorrer os arquivos recebidos levando em consideração os possíveis formatos (PDF, XLS, XLSX, CSV, XML, JSON, ODS) e extrair o conteúdo com base no formato. Destinar uma thread para essa funcionalidade.

REQ 04: Ler os conteúdos extraídos com base em um layout pré-definido e alimentar um modelo único de dado e salvar em arquivo. Destinar uma thread a essa funcionalidade.

REQ 05: Ler todos os dados interpretados e salvos em arquivo e subir para o banco de dados. Destinar uma thread a essa funcionalidade, se o banco de dados estiver indisponível por exemplo, requisitos anteriores como download e processamento dos conteúdos não sofrerão impacto.

REQ 06: Ao salvar informações, separar por etapa, nomenclatura original e data e hora para evitar substituir arquivos.

REQ 07: Consultar os layouts pré-definidos por fornecedor para realizar a leitura das faturas.

REQ 08: Se o sistema não identificar um layout existente, a informação deve ser armazenada para que seja configurado um layout específico para deste arquivo.

REQ 09: Cadastrar fornecedores no aplicativo desktop.

REQ 10: Cadastrar layout de arquivo dos fornecedores no aplicativo desktop.

REQ 11: Visualizar tela de alertas no aplicativo desktop, gerado pelo serviço, para ver os endereços de e-mail recebidos, que podem ser fornecedores não cadastrados ou tentativas de intepretação de arquivos que não possuem layout cadastrado.

5.2. Requisitos Não Funcionais

RNF 01: Garantir boa performance, separando as principais funcionalidades em threads, permitindo baixar, processar e gravar informações de forma assíncrona.

RNF 02: Permitir baixar arquivos de até 10GB.

RNF 03: Garantir ausência de falhas ao instanciar o serviço mais de uma vez, expandido a capacidade de processamento dos dados.

RNF 04: Funcionar em plataforma Windows, como um serviço e um aplicativo desktop.

6. Roadmap

6.1. Primeiro release

- Inicialmente o sistema deverá realizar a exportação de um layout de arquivo em PDF por ser o formato mais complexo de ser lido e exportado.
- O sistema irá utilizar dados de teste de um Mock Service contendo cadastros de fornecedores e layout definidos para rodar os arquivos de teste, inseridos manualmente na pasta de trabalho do serviço.
- Após extrair o PDF em texto a ler o arquivo extraído, o sistema irá expor na tela os dados obtidos nos campos CNPJ e Valor Total da fatura.

6.2. Evolução em 3 meses

- Deverá ser incluída a opção de leitura de arquivos em CSV, JSON, XML e a conversão dos arquivos XLS em CSV, recebidos por e-mail.
- Deverá ser gravado na pasta de trabalho do serviço, arquivos baixados do e-mail de forma automaticamente.
- Deverá ter o CRUD dos cadastros de Fornecedores e Layouts de arquivo.
- Permitir mais de um layout por fornecedor, usando palavra chave no nome do arquivo enviado para o e-mail.
- Pesquisar a melhorar a biblioteca de leitura de PDFs, que tenha boa performance e melhor compatibilidade com as faturas enviadas pelos fornecedores.
- Projetar uma nova camada API para recebimento dos formatos JSON e XML por HTTP Rest.

7. O Problema: layouts e métodos por tipo de arquivo

Para tratar diferentes layouts de fatura de diferentes fornecedores, a solução proposta tem o objetivo de disponibilizar para o cliente a possibilidade de identificar e cadastrar faturas que não tiveram seus dados extraídos, ou ainda, que a informação extraída não era o esperado, para isso é feito um cadastro do layout e do fornecedor no sistema Windows Forms e o sistema ficará encarregado de receber e extrair as informações das faturas.

7.1. Extensão PDF

O arquivo de extensão PDF é o mais complicado de se trabalhar em relação ao demais pois ele se trata de um documento digital feito para apresentação visual fiel ao documento criado pelo autor. Ele tem características que não prezam pelo posicionamento dos elementos com dados de forma padronizada no arquivo, portanto, para trabalhar com este formato devemos extrair os textos contidos no documento e analisar posições de textos chave como um campo chamado “valor total” de ser uma **condição** para que o módulo interpretador do sistema possa validar os demais dados como por exemplo o valor total da fatura que dever ser do **tipo** numérico.

7.2. JSON e XML

São formatos que definem arquivos texto com uma padronização de troca de dados entre sistemas e podem ser facilmente generalizados para encontrar valor por nomenclatura dos campos que descrevem este valor, tornando fácil o cadastro do layout por nome do campo que será pesquisado e a **condição** para interpretar o arquivo seria encontrar o campo e ele ter o conteúdo do **tipo** esperado.

7.3. XLS, XLSX e ODS

São formatos de arquivos com dados em planilha e para facilitar a interpretação do conteúdo desses arquivos, podemos simplesmente trata-los como CSV fazendo a **conversão**. Para XLS e XLSX temos ferramentas para fazer isso presentes no namespace *Microsoft.Office.Interop.Excel.XlFileFormat.xlCSV*. Para arquivos ODS gerados por aplicativos do tipo Office, na prática são arquivo texto em XML portanto para o serviço interpretador, podemos realizar a **conversão**.

7.4. CSV

CSV é um formato de arquivo texto que tem um certo nível de padronização pois ele utiliza um separador que deve ser configurado pelo usuário para separar os dados em seu conteúdo em colunas, portanto uma possível **condição** de validação ao extrair dados desse formato seria o cabeçalho do arquivo, se existir, logo no cadastro do layout devemos incluir a opção de escolha de arquivo com ou sem cabeçalho. Para arquivos que possuam um cabeçalho, podemos usar sua nomenclatura para identificar os valores das faturas, caso não possuir devemos deixar o usuário cadastrar o posicionamento do dado e este não poderá variar.

8. Bibliotecas de leitura de PDF

8.1. PDFSharp

Essa biblioteca possui uma licença MIT (Instituto de Tecnologia de Massachusetts) e por ser livremente utilizada para fins comerciais, por este motivo foi escolhida como a biblioteca inicial do projeto, no entanto, ela se provou ineficiente para leitura de PDFs gerados em

padrão UNICODE para codificação de caracteres do arquivo, sendo necessária sua substituição para atender qualquer tipo de PDF enviado pelos fornecedores.

8.2. iTextSharp

A iTextSharp é uma adaptação para C# de uma biblioteca já conhecida por outras linguagens chamada iText, ela consegue realizar a extração de texto do PDF inclusive no padrão UNICODE, no entanto, ela é uma biblioteca disponibilizada sob a licença AGPL (*general public license*) que autoriza a utilização comercial do software com a condição da disponibilização de seu código fonte.

9. Arquitetura do Software

9.1. Camadas

O sistema será desenvolvido considerando várias camadas de software, o objetivo da divisão é facilitar a manutenção ao subdividir reponsabilidade e dependências determinadas áreas do sistema, promover o reaproveitamento de código ao compartilhar projetos do tipo *class library* com regras de negócios, classes e métodos já implementados e que podem ser comuns na organização. As principais camadas são:

- Camada de aplicação – onde deve-se alocar serviços e view models relacionado ao aplicativo que se está sendo criado.
- Camada de domínio – que define qual o domínio da organização, normalmente é um reflexo da base de dados e o núcleo do sistema. Grande parte das regras de negócio do sistemas são encontradas aqui assim como as entidades do domínio devem poder se validar conforme suas próprias propriedades.
- Camada de dados – que conhece a tecnologia e possui os métodos para subir registros para o banco de dados.
- Camada de ferramentas – onde aguarda métodos comuns de conversão de dados, criptografia e faz operações em componentes diversos como e-mail. Quanto aos componentes diversos, quanto mais relevante for o componente utilizado, maior a necessidade de criar um camada específica para o mesmo.

9.2. Injeção de Dependências

Uma forma de diminuir o acoplamento entre dependências e entre camadas é o desenho da solução com injeção de dependências ou também chamado Inversão de Controle. Este método permite registrar através das assinaturas (*interfaces*) os componentes do código para que possam ser instanciados de maneira fácil ou removidos sem maiores impactos no código.

9.3. Background Services e Multithreading.

Falando de performance, além de utilizar outros métodos para otimizar o algoritmo do sistema, é necessário extrair o máximo de recursos de processamento da máquina em que a solução estará hospedada e para isso precisamos tarefas assíncronas ou multithreading.

O componente chave para cumprir o objetivo é o Background Service, que utilizar a classe Task para executar serviços de longa duração (*long running services*) e para cada serviço que utiliza outro recurso (exemplos: e-mail, banco de dados, biblioteca externa ao software e etc...), um Background Service deverá ser chamado e ele foi nomeado de Worker.

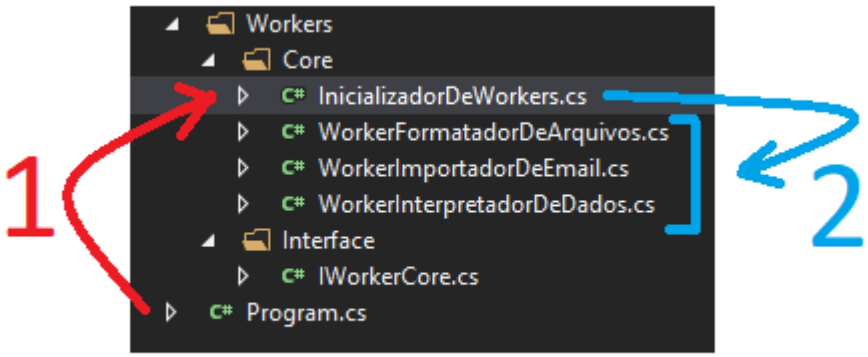


Figura 1 - Fluxo de chamada dos workers

A figura anterior (*Figura 1*) demonstra como ocorre a chamada de um inicializador de workers, que é instanciado com injeção de dependências através de um construtor de serviços (`services.AddSingleton`) da interface *`IHostedService`*. O processo pode ser repetido para instanciar quantos workers forem necessários estar em funcionamento.

9.4. Organização de Arquivos

Está previsto que o sistema irá trabalhar com três principais pastas que irão controlar input e outputs de documentos e dados pelo serviço.

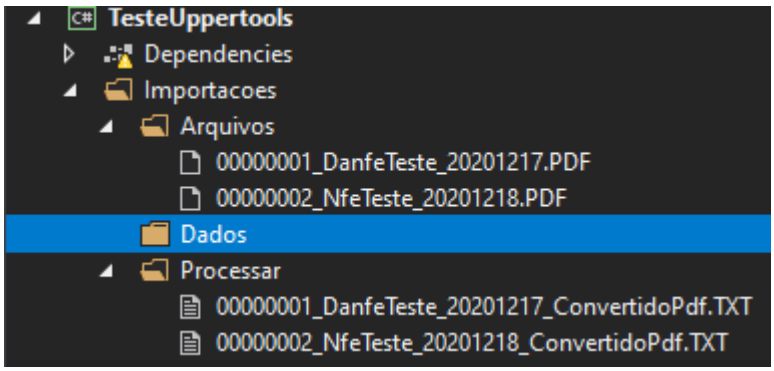


Figura 2- Organização de arquivos

Através da figura anterior (*Figura 2*) é possível ter uma noção de como será estruturado os arquivos do sistema, sendo a pasta Arquivos destinada a armazenas os downloads das faturas, a pasta Processar onde serão armazenados arquivos formatados, convertidos e com dados

extraídos e a pasta Dados onde serão armazenados os registros em um modelo unificado para serem exportados para um banco de dados.

10. Casos de Uso

10.1. Diagrama UML

Em relação ao diagrama, vale mencionar que por mais que o fornecedor não esteja presente pois ele não tem um caso de uso presente dentro da solução, ele presta um papel importante enviando as faturas por e-mail.

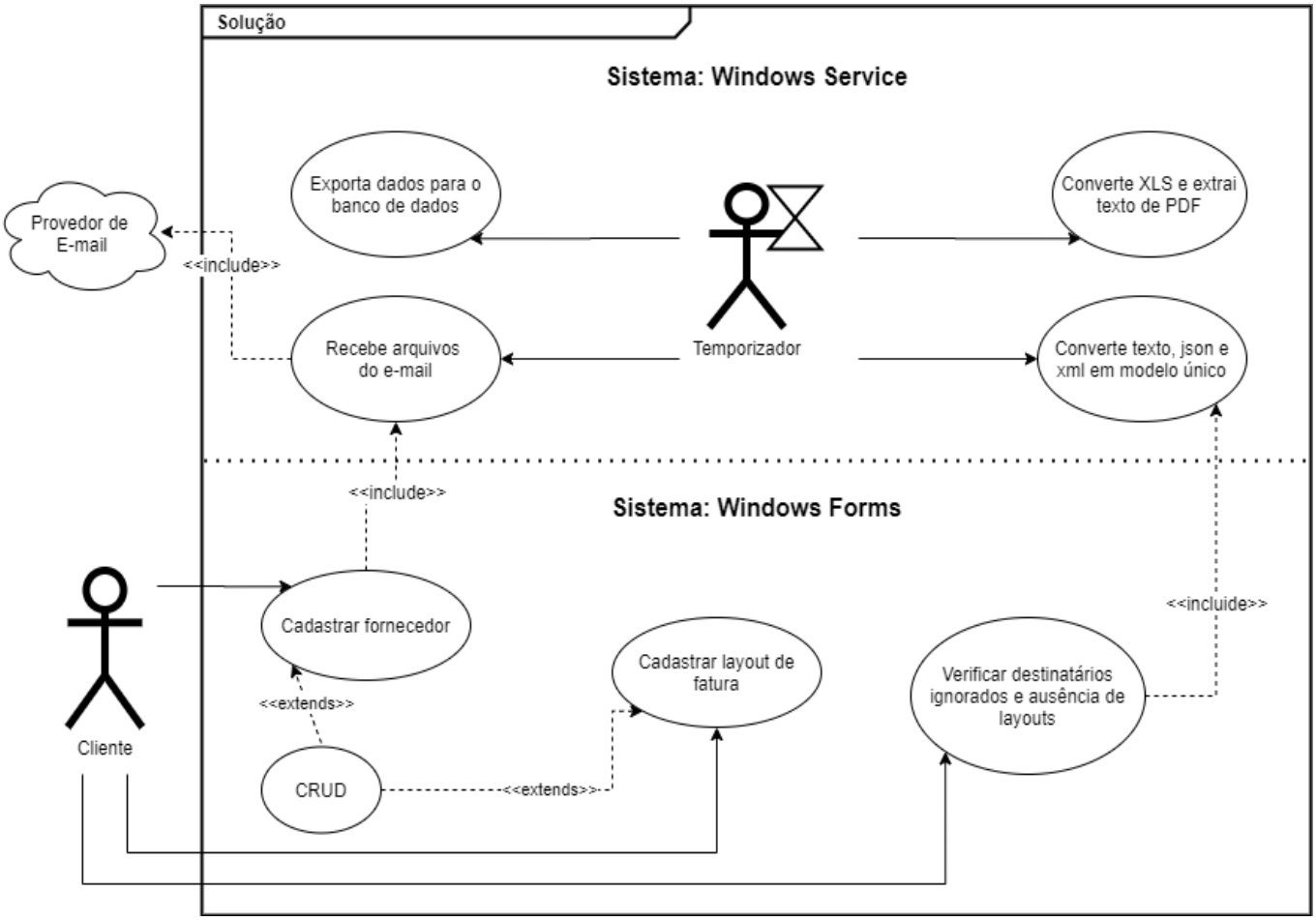


Figura 3- Diagrama de casos de uso

10.2. Descrição dos casos de uso

UC001 - Recebe arquivos do e-mail

Através de um worker importador configurado no serviço, os e-mails recebidos serão lidos e importados para a solução.

UC002 – Converte texto, json, xml em modelo único

Através do worker interpretador, arquivos de um determinado fornecedor serão confrontados com os layouts cadastrados para extrair principais informações como CNPJ, valor da fatura e etc...

UC003 – Converte XLS e extrai texto de PDF

Através do worker formatador, arquivos que possuem uma certa dificuldade em se trabalhar serão convertidos e formatados para outros padrões, como PDF para Texto, ODS para XML, XLS ou XLSX para CSV.

UC004 – Exporta dados para o banco de dados

Após coletar uma certa quantidade de dados, um worker exportador poderá ler um arquivo com dados extraídos das faturas e subir em massa para o banco, simplificando em uma única transação e garantindo desempenho.

UC005 – Cadastrar fornecedor

Manter os fornecedores no sistema para recebimento dos e-mails de forma segura, garantindo endereços remetentes confiáveis.

UC006 – Cadastrar layout de fatura

Manter os layouts de fatura por fornecedores, configurando as possibilidades de receber essas informações.

UC007 – Verificar destinatários ignorados e ausência de layouts

Através de uma tela com alertas, o usuário poderá verificar o possíveis problemas gerados pelos workers, como fornecedores não cadastrados, layouts ausentes ou que estejam gerando dados inválidos.

11. Fluxograma do Windows Service

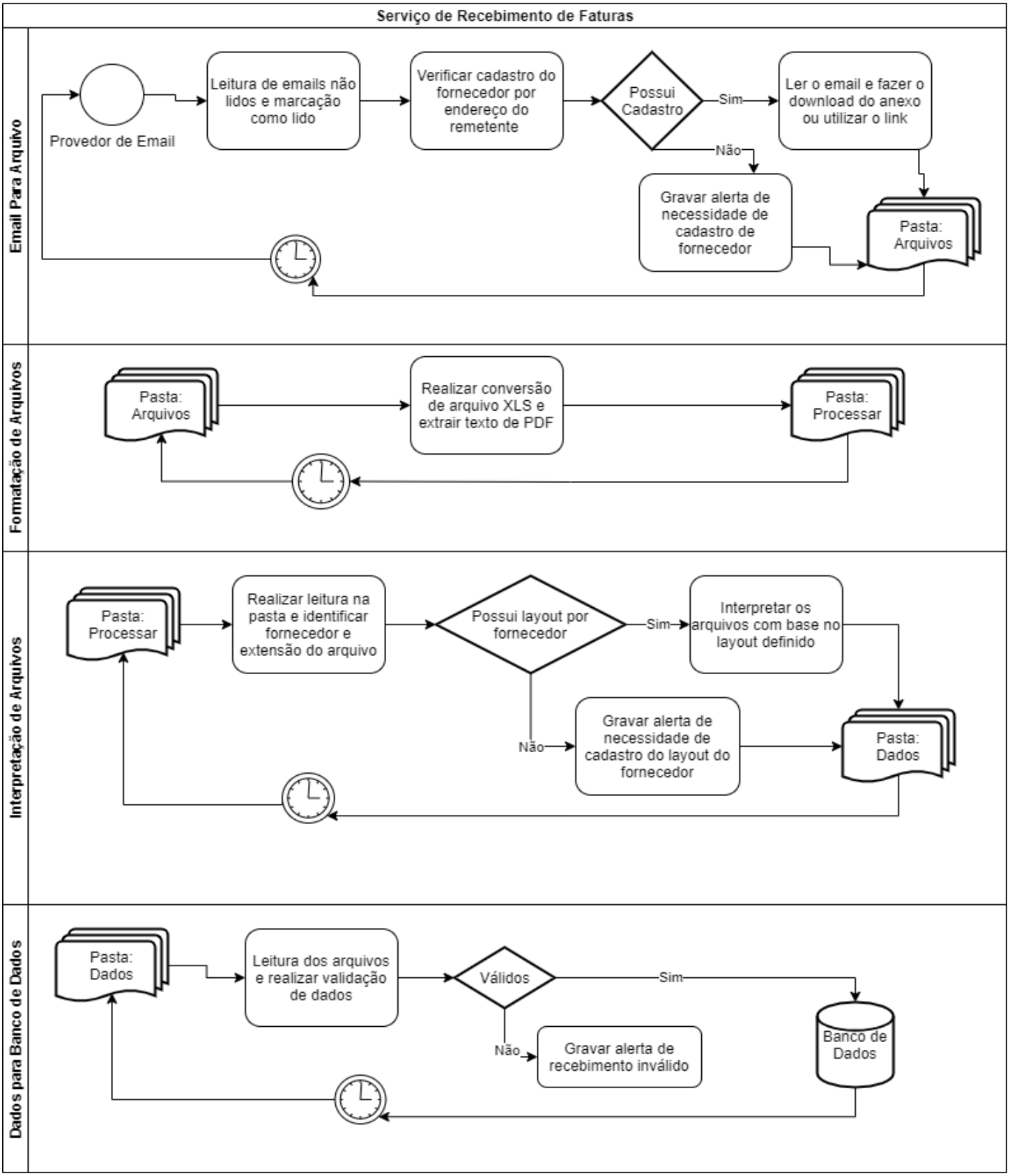


Figura 4- Fluxograma do Windows Service