

---

# **Plano de Especificação: Sistema de Agendamento de Serviços Acadêmicos**

Versão 1.4

2024

---

## Sumário

<b>1. Plano de especificação do ASA.....</b>	<b>3</b>
<b>2. Resumo.....</b>	<b>3</b>
<b>3. Pessoas envolvidas.....</b>	<b>4</b>
<b>4. Recursos necessários.....</b>	<b>4</b>
4.1. Recursos Humanos.....	4
4.2. Recursos Tecnológicos.....	4
4.3. Recursos de Integração.....	5
<b>5. Metodologia.....</b>	<b>5</b>
5.1. Etapas de Especificação.....	5
<b>6. Métodos e Linguagem Formal.....</b>	<b>6</b>
6.1. Método Formal.....	6
6.2. Especificação Formal.....	7
<b>7. Cenários de Uso.....</b>	<b>8</b>
<b>8. Cronograma.....</b>	<b>9</b>

# **1. Plano de especificação do ASA**

Este documento é responsável por apresentar o plano de especificação formal para o desenvolvimento do ASA - Sistema de Agendamento de Serviços Acadêmicos. A especificação garante a precisão e integridade dos requisitos funcionais e não funcionais do sistema.

## **2. Resumo**

O Sistema de Agendamento de Serviços Acadêmicos, desenvolvido em C#, tem como objetivo principal centralizar e automatizar os processos de agendamento de serviços especializados na UFERSA. Atualmente, esses serviços são gerenciados por meio de formulários dispersos no site da universidade. O novo sistema integrará todas essas informações em um único local, simplificando o acesso e tornando o processo mais eficiente e transparente para os usuários.

Como requisitos, o sistema permitirá que os usuários se cadastrem e façam alterações em suas senhas, além de gerenciar os serviços acadêmicos disponíveis. Os usuários poderão visualizar os horários disponíveis para cada serviço e agendar ou cancelar atendimentos de forma prática. Isso visa não apenas facilitar o acesso dos profissionais e estudantes aos serviços acadêmicos, mas também melhorar a transparência e a eficiência do processo de agendamento.

O desenvolvimento do sistema adota uma metodologia híbrida que combina os valores e princípios do XP (Extreme Programming) com as técnicas organizacionais do SCRUM. No XP, o foco será em simplicidade, feedback, comunicação e respeito, enquanto no SCRUM, serão implementados artefatos e eventos, como sprints e reuniões, para garantir uma gestão ágil e colaborativa das entregas contínuas ao longo do projeto. Assim, o sistema será desenvolvido de maneira eficiente e adaptável, atendendo às necessidades específicas dos usuários e do ambiente acadêmico da UFERSA.

### 3. Pessoas envolvidas

- LAVINIA DANTAS DE MESQUITA
- CRISTIANA DE PAULO

### 4. Recursos necessários

#### 4.1. Recursos Humanos

- 4.1.1. **Desenvolvedores de Software (C#):** Aqueles com experiência em desenvolvimento de software utilizando a linguagem C#, capazes de implementar as funcionalidades do sistema.
- 4.1.2. **Especialistas em Metodologias Ágeis:** Aqueles com experiência em XP e SCRUM para gerenciar o desenvolvimento, garantindo a aplicação adequada das práticas ágeis.
- 4.1.3. **Analistas de Sistemas:** Responsáveis por entender os requisitos dos usuários e traduzir esses requisitos em especificações técnicas.
- 4.1.4. **Designers de UI/UX:** Responsáveis pelo design da interface do usuário, garantindo que o sistema seja intuitivo e fácil de usar.
- 4.1.5. **Testadores de Software:** Equipe dedicada à validação e verificação do sistema, garantindo que ele funcione conforme o esperado e esteja livre de falhas.
- 4.1.6. **Escritores:** Responsáveis pela constituição dos documentos que detalham as funcionalidades do sistema, os requisitos, interações e mapeamento do progresso do sistema.

#### 4.2. Recursos Tecnológicos

- 4.2.1. **Ambiente de Desenvolvimento:** Ferramentas e ambiente para desenvolvimento em C#, utilizando o framework Blazor para a criação de interfaces web interativas e responsivas com o auxílio do ASP.NET. Para o frontend, além da edição visual usando o CSS, também usaremos o bootstrap para responsividade. Além desses, o Visual Studio e o repositório disponível no GitHub.
- 4.2.2. **Servidores de Aplicação e Banco de Dados:** Infraestrutura necessária para hospedar o sistema e armazenar dados, garantindo disponibilidade e segurança, no momento, ela não foi escolhida, mas a sugerida foi a hospedagem no site

ElephantSQL, um site gratuito no caso da quantidade de dados ser pequena.

- 4.2.3. **Ferramentas de Modelagem:** Ferramentas que auxiliam na prototipagem e na criação dos diagramas usados. Para a modelagem do sistema, diagramas de caso e uso e diagrama de classe foi usada a Astah UML; Para o modelo lógico do banco de dados, o BRModelo, como ferramenta auxiliar na prototipação, o Figma; E, para a elaboração das Redes de Petri, utilizaremos a PIPE.
- 4.2.4. **Ferramentas de Gestão de Projetos:** Softwares para acompanhar o progresso das tarefas, sprints, e backlog do projeto, no momento a organização está sendo feita através de branches dentro do GitHub e Planilhas do Google.
- 4.2.5. **Ferramentas de Comunicação:** Plataformas para facilitar a comunicação entre a equipe, no momento, para reuniões será utilizado o Google Meet, e para troca de mensagens e documentos, o GitHub e o Whatsapp.

### 4.3. Recursos de Integração

- 4.3.1. **Ferramentas de Automação de Testes:** Para realizar testes automáticos e garantir a qualidade do código ao longo do desenvolvimento, utilizaremos as seguintes: Selenium, NUnit, Lighthouse, SonarLint e entre outras, valendo salientar que os resultados mais importantes serão filtrados e tais recursos podem mudar com o tempo.
- 4.3.2. **Planos de Teste:** Com o uso da ferramenta citada anteriormente, a cada implementação será executado um novo teste e o registro dos seus resultados para futuras alterações e garantia de qualidade.
- 4.3.3. **Revisões de Código:** Toda semana, juntamente com as reuniões, o progresso e estado do código será revisado em caso de erros não identificados pelo programa de testes, no caso, erros referentes a má interpretação e falhas humanas. Isto irá garantir que o objetivo do ASA seja cumprido com excelência.

## 5. Metodologia

Este projeto seguirá uma metodologia formal baseada em etapas claramente definidas para garantir que os requisitos funcionais e não funcionais sejam atendidos.

## 5.1. Etapas de Especificação

- 5.1.1. **Levantamento dos Requisitos:** Coleta e análise de requisitos.
- 5.1.2. **Modelagem com Redes de Petri:** Criação de modelos que representam os processos de agendamento, serviços e gerenciamento de usuário.
- 5.1.3. **Verificação e Validação:** Análise dos modelos com o objetivo de garantir consistência e ausência de conflitos.
- 5.1.4. **Refinamento e Transformação:** Ajustes nos modelos e preparação para a implementação.
- 5.1.5. **Implementação:** Desenvolvimento do sistema com base nos modelos formais.
- 5.1.6. **Teste e Validação:** Testes para garantir que o sistema atenda aos requisitos especificados.
- 5.1.7. **Entrega e Manutenção:** Entrega do sistema e suporte contínuo.

## 6. Métodos e Linguagem Formal

### 6.1. Método Formal

Após a análise das abordagens de especificação apresentadas em sala de aula, o método de especificação formal adotado será com base nas Redes de Petri.

As Redes de Petri oferecem uma representação gráfica intuitiva, o que será essencial para nossa equipe composta de duas disciplinas diferentes e com níveis de experiência em engenharia de software distintos.

A visualização clara das transições de estado e eventos do sistema facilitará a compreensão da lógica do sistema de agendamento de serviços. Além disso, as Redes de Petri são eficazes para representar e analisar a concorrência entre os serviços.

Entre os benefícios das Redes de Petri, um a ser destacado é a modelagem modular, que facilita ajustes e evolução do sistema ao longo do tempo, conforme surgem novos requisitos ou mudanças; o outro é que elas adequadas para análises formais, como a verificação de propriedades e a detecção de possíveis deadlocks ou conflitos, o que assegurará a corretude e a consistência do sistema desde as fases iniciais do desenvolvimento.

Para a especificação formal do nosso sistema de agendamento de serviços usando Redes de Petri, planejamos incluir:

- A criação de uma rede de Petri que modela o processo de cadastro e edição de serviços;
- A especificação formal das restrições e condições que devem ser atendidas pelo sistema, como a prevenção de conflitos de horários.

Por fim, a escolha das Redes de Petri foi fundamentada nas vantagens de visualização intuitiva, modelagem de concorrência, capacidade de análise formal e facilidade de evolução, o que permitirá desenvolver um sistema robusto, eficiente e de alta qualidade, que atenderá às necessidades dos alunos e da instituição de ensino.

## **6.2. Especificação Formal**

### **6.2.1. Requisitos Funcionais**

#### **Agendamento**

[RF] - O sistema deve permitir que usuários do tipo discentes solicitem o agendamento dos serviços.

[RF] - O sistema deve permitir que os usuários visualizem os detalhes do agendamento cadastrado.

[RF] - O sistema deve permitir que usuários do tipo discente e do tipo profissional cancelem um agendamento antes da data marcada.

[RF] - O sistema deve notificar o cancelamento do agendamento para as partes envolvidas na consulta.

[RF] - O sistema deve notificar a confirmação do agendamento para o discente no ato da aceitação e 24 horas antes da data marcada.

[RF] - O sistema deve permitir que usuários do tipo discente visualizem os horários disponíveis para cada serviço ofertado.

#### **Serviço**

[RF] - O sistema deve conter os serviços de psicólogo, pedagogo, nutricionista e esporte para a comunidade acadêmica.

[RF] - Os serviços de psicólogo e nutricionista devem disponibilizar as opções de consulta e retorno.

[RF] - O serviço de esporte deve permitir que interessados realizem matrícula nas modalidades disponíveis.

#### **Usuário**

[RF] - O sistema deve permitir o cadastro de usuário por meio de um email e de uma senha.

[RF] - O sistema deve permitir que usuários, uma vez cadastrados, possam alterar a senha.

### **Recursos**

[RF] - O sistema deve permitir que usuários do tipo profissional cadastrem informações acerca do serviço prestado.

[RF] - O sistema deve permitir que usuários emitam relatórios com as informações do serviço prestado.

#### **6.2.2. Requisitos não funcionais**

##### **Usabilidade**

[RNF] - Estética da interface do usuário: O sistema deve apresentar uma interface responsiva, com cores e tipografia padronizados, além de ícones personalizados, obedecendo ao contexto do uso.

[RNF] - Proteção ao erro do usuário: O sistema deve retornar um alerta/mensagem afirmando sucesso ou fracasso ao finalizar as operações.

##### **Segurança**

[RNF] - Integridade: O sistema deve criptografar a senha dos usuários para armazená-la em seu banco de dados.

[RNF] - Confidencialidade: O sistema deve garantir a segurança dos dados pessoais dos discentes e profissionais.

##### **Portabilidade**

[RNF] - Adaptabilidade: O sistema deve ser acessível por meio de um navegador web, programado para se adaptar a diferentes plataformas (desktop e mobile).

#### **6.2.3. caso precise**

## **7. Cenários de Uso**

### **7.1. Diagramas de Caso de Uso**





## **8. Cronograma**

O cronograma ainda não foi definido.