OpenInsurance

Política de Versionamento de APIs
- Open Insurance Brasil -

- V1.2 –

OpenInsurance

1. Índice

Sumário

2. Referências **3**

3. Histórico de Alterações **3**

4. Introdução **3**

4.1 Objetivo 4

4.2 Conceitos e Terminologia 4

5. Escopo **4**

6. Diretrizes Gerais do Versionamento **4**

6.1 Ciclo de Vida 5

6.2 Tipos e Enquadramento do Versionamento 8

6.3 Quantidade de Versões da mesma API em Produção 9

6.4 Período e Regras de Convivência Versões 9

6.5 Versionamento na URL da API 10

7. Governança de Versionamento **11**

7.1 Modelos de Priorização de Alterações 11

7.2 Análise e Priorização da Solicitações de Alterações 12

7.2.1 Fluxo de Versionamento Padrão (release de melhorias) 13

7.2.2 Fluxo de Versionamento Emergencial (hotfix) 15

2. Referências

Essa política se baseia nas definições obtidas em GT de Dados e nas referências a seguir:

Documentação	Referências
Semantic Versioning (SemVer)	https://semver.org
***Open API Specification* (OAS) - v3**	https://swagger.io/specification
Manual de APIs do Open Insurance (SUSEP)	Link de acesso

Tabela 1 - Referências da política de versionamento

3. Histórico de Versões

Versão	Data	Descrição
1.0	24/09/2025	Versão inicial da política de versionamento do Open Insurance
1.1	27/10/2025	Inclusão de período de preparação do ambiente em casos de ajustes emergenciais major
1.2	10/12/2025	Redefiniçao do periodo de adaptação no fluxo de alteração emergencial major, para periodo de convivência

Tabela 2 – Histórico de versões da política de versionamento

4. Introdução

Este documento estabelece as regras e diretrizes para o versionamento das especificações técnicas (swaggers) utilizados no ambiente, garantindo consistência, rastreabilidade e interoperabilidade entre os sistemas das participantes do Open Insurance (OPIN).

São descritos nesse documento os processos envolvidos, regras e convenções utilizadas pelo ecossistema. Se houver dúvidas sobre os processos, sugestões de melhorias em relação ao documento ou práticas utilizadas, favor endereçar ao Service Desk do Suporte Técnico N2.

4.1. Objetivo

Definir regras e práticas recomendadas para a gestão de versões dos contratos Swagger, facilitando o consumo e manutenção das APIs disponibilizadas no portal do desenvolvedor do Open Insurance, além de gerar visibilidade sob os processos utilizados.

4.2. Conceitos e Terminologia

- **Swagger:** Nome anterior da especificação hoje conhecida como Open API.
- **Versão da API:** Identificador que expressa mudanças compatíveis e incompatíveis.
- **Semantic Versioning (SemVer):** Convenção de versionamento semântico composta por MAJOR.MINOR.PATCH.
- **API (Application Programming Interface**): conjunto de regras, padrões e ferramentas que permite que um software se comunique com outro.
- **Endpoint:** ponto final de comunicação em uma API. URL específica (ou endereço) onde um recurso ou serviço pode ser acessado por meio de requisições.
- **Pattern:** expressão regular que válida a composição e/ou formato de um dado.
- **Convenção*:*** acordo ou regra estabelecida entre pessoas, grupos ou instituições para padronizar práticas, comportamentos ou formas de comunicação.

5. Escopo

Aplica-se a todos os contratos Swagger utilizados na documentação técnica das APIs do Open Insurance disponibilizadas no portal do desenvolvedor, conforme indicado abaixo:

- Swaggers de Fase 1 – Catálogo de produtos (APIs públicas).
- Swaggers de Fase 2 – Movimentação de dados.
- Swaggers de Fase 3 – Iniciação de serviços.
- Swaggers de Monitoramento do ecossistema.

6. Diretrizes Gerais de Versionamento

Nas seções a seguir estão elencadas as regras e diretrizes sobre o versionamento de swaggers no ecossistema OPIN.

6.1. Ciclo de vida

A seguir é apresentado a visão geral do ciclo de vida das versões dos contratos swaggers desenvolvidos no ambiente do Open Insurance, bem como a descrição e especificidades de cada uma das etapas:

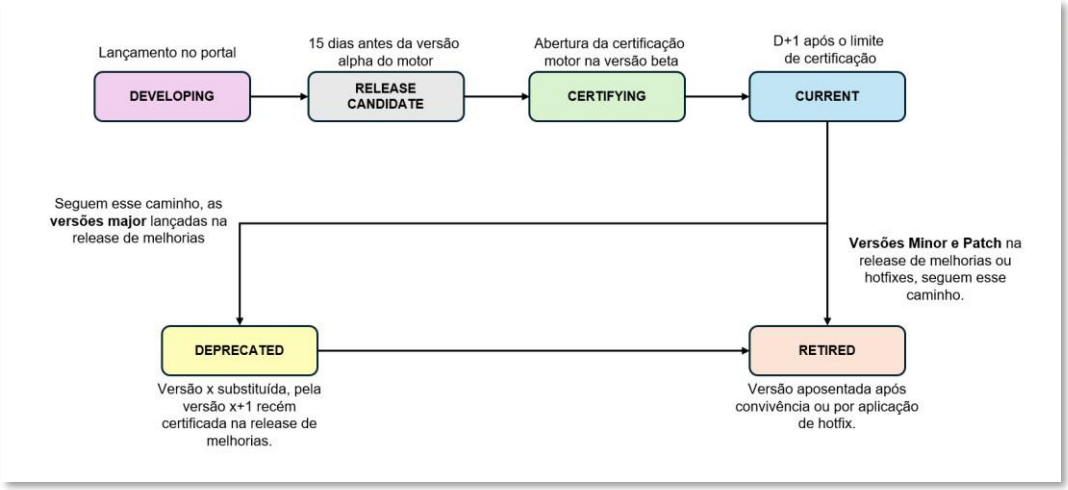

Figura 2 – Ciclo de vida dos swaggers OPIN

- ***Developing*:** etapa em que a Estrutura OPIN está desenvolvendo uma nova versão do swagger. Nesta etapa, as especificações já aprovadas pelo GT de Dados serão disponibilizadas na Área do Desenvolvedor no Portal do Open Insurance.
    - A publicação da versão inicial do swagger no portal do desenvolvedor, marca o início dessa etapa. As participantes podem sugerir melhorias e solicitar correções na especificação caso seja necessário.
    - O swagger é disponibilizado para conhecimento e análise das participantes, dessa forma, ainda não é recomendada a implementação do swagger nesse momento.
- ***Release Candidate*:** etapa em que a nova versão da API estará disponível para implementação.
    - Nessa versão serão incluídas todas as alterações solicitadas e aprovadas pelo GT de Dados, durante a versão “Developing”.

- Essa etapa se inicia em d-15 da liberação do uso do motor de conformidade na versão alpha.
- Poderá ser usada para reportar possíveis *bugs* ainda não corrigidos nas especificações.
- As participantes já podem iniciar a implementação do swagger.
- Poderão ser lançadas diversas versões da API em Release Candidate, à medida que forem identificados pontos de correção. A cada nova versão na etapa release candidate, a versão será incrementada conforme a seguir:
  - Primeira versão: 1.0.0-rc.
  - Segunda versão: 1.0.0-rc2.
  - Terceira versão: 1.0.0-rc3, e assim por diante.

- ***Certifying*:** etapa em que a versão do swaggers estará disponível para certificação.
  - O evento de liberação do uso do motor de conformidade na versão beta, e início do período de certificação, marcará o início desta etapa.
  - Poderá ser usada para reportar possíveis *bugs* nas especificações ou nas ferramentas de teste (Motor de Conformidade e Mock Open Insurance).
  - Como o swagger ainda não consta em produção, ajustes nessa etapa podem ser aprovados pelo GT de Dados sem a necessidade de realizar uma hotfix, assim como nas etapas “Developing” e “Release Candidate”.

- ***Current*:** essa etapa marca o início da operação em produção utilizando a versão recém certificada da API. A versão recém certificada assume o status CURRENT, enquanto a versão anteriormente em produção assume o status DEPRECATED.
  - Esse evento, além de indicar o início da operação em produção com a nova versão, também marca o início do período de convivência das versões DEPRECATED e CURRENT.
  - Ao fim do período de certificação, as participantes deverão cadastrar as APIs recém certificadas no diretório e operá-las em produção. Isso deve ocorrer em d+1 ao último dia de certificação.

- Uma vez que a versão esteja em CURRENT, só poderá receber alterações emergenciais via hotfix. Demais alterações deverão ser incluídas e priorizadas em uma release de melhorias.
- A partir dessa etapa, a versão é considerada estável.

- ***Deprecated*:** etapa em que a versão vigente dos swaggers está sendo substituída pela versão recém certificada no ambiente produtivo.
  - O evento de entrada em produção e fim do período de certificação marcará o início desta etapa.
  - Como as versões estão sendo substituídas pelas participantes, nessa etapa devem ser suportadas iniciações de serviços e compartilhamento de dados nas versões DEPRECATED e CURRENT. Os detalhes de convivência são descritos no tópico *"6.4 Períodos de adequação e Regras de Convivência".*
  - Nesse status, a versão não recebe mais suporte.

- ***Retired*:** Etapa em que a versão da API se torna indisponível em ambiente produtivo e deixa de ser utilizada/exigida pelo ambiente.
  - Ao alcançar esse status, a versão será descontinuada do Motor de Conformidade, Diretório, Mock Open Insurance e FVP.
  - Essa etapa é alcançada em d+1 a convivência estabelecida pelo GT de Dados, entre as versões DEPRECATED e CURRENT, onde "d" é o último dia do período de convivência.
  - Responsabilidades e ações para a remoção do ambiente produtivo:
    i. **Transmissor:** Descadastrar a versão "retired" do Diretório e remover do seu API gateway;
    ii. **Receptor:** Garantir que não haverá mais acionamentos da versão "retired", invalidando eventuais caches internos e acionando apenas o novo **endpoint.**
    iii. **Fornecedores de conformidade e testes:** Remover as versões das ferramentas
    iv. **Suporte Técnico N2:** Atualizar status da versão no portal do desenvolvedor

6.2. Tipos e enquadramento do versionamento

No ecossistema do Open Insurance o versionamento dos swaggers seguem a convenção *Semantic Versioning* (SemVer) na sua versão mais recente, conforme definição do GT de Dados. Já a classificação de alterações entre Major Minor e Patch é realizada de acordo com a descrição do manual de APIs da SUSEP, conforme abaixo:

- **Major:** *"Inclui novas características da implementação, mudanças, correções a serem incorporadas e que podem ser incompatíveis com versões anteriores, por exemplo, v1.0.0 e v2.0.0;"*
- **Minor:** *"Pequenas mudanças nos elementos já existentes, com manutenção da compatibilidade com as versões até a major imediatamente anterior, por exemplo, v1.1.0 e v1.2.0"*
- **Patch:** "*Esclarecimentos às especificações minor, não incluem alterações funcionais, por exemplo, v1.1.1, v1.1.2"*

Dadas as definições, encontra-se a seguir a legenda de alterações utilizada no OPIN:

Alteração	Enquadramento
Inclusão de campo obrigatório	**MAJOR**
Remoção de campo (obrigatório ou não)	**MAJOR**
Alteração do tipo do campo	**MAJOR**
Tornar campo obrigatório	**MAJOR**
Diminuição da quantidade de caracteres permitida	**MAJOR**
Alteração na grafia do nome do campo	**MAJOR**
Alteração de pattern que impacta a validação	**MAJOR**
Alteração no significado/semântica de campo existente	**MAJOR**

Inclusão de campo não obrigatório	**MINOR**
Inclusão de novos valores a um campo do tipo enum	**MINOR**
Tornar campo opcional	**MINOR**
Aumento do número de caracteres permitidos	**MINOR**
Inclusão de endpoint, parâmetro ou funcionalidade sem alterar os já existentes	**MINOR**
Atualizações de pattern que não impacta a validação anterior	**PATCH**
Correções gramaticais, ortográficas ou visuais em descrições da API	**PATCH**
Mudança na ordenação dos campos do JSON	**PATCH**
Ajuste na descrição de campo (sem alterar semântica ou regras)	**PATCH**
Alteração em exemplos de uso (exemplo de payload, header, body etc.)	**PATCH**

Tabela 3 – Legenda de alterações conforme SemVer

6.3. Quantidade de versões da mesma API em Produção

Conforme definição do GT de Dados, as participantes podem manter até duas versões da mesma API em ambiente produtivo. Esse cenário deve ocorrer durante períodos de convivência estabelecidos pelo GT na passagem de versões major, as quais devem ser:

- **Versão x:** versão major com status DEPRECATED que está caindo em desuso pelo ecossistema devido a uma nova versão major em produção (x+1).
- **Versão x+1:** versão major com status CURRENT recém certificada pelas participantes e que substitui a versão major DEPRECATED (x).

Vale ressaltar que fora dos períodos de convivência as participantes devem apresentar somente a versão major mais recente da API em ambiente produtivo.

6.4. Períodos de adequação e Regras de Convivência

Conforme estabelecido em GT de Dados, para que exista uma mudança suave entre as versões major de APIs, é necessário a aplicação de um período de transição no qual duas versões (deprecated e current) coexistam. Esse período pode ser de:

- 45 dias corridos

- 60 dias corridos
- 90 dias corridos

O período de convivência estabelecido pelo GT de Dados será determinado com base na análise do backlog e na complexidade da release de melhorias em andamento, realizada pelo próprio GT.

No caso de alterações emergenciais major **(hotfix major)**, que podem derivar dos seguintes cenários:

- **Requisitos técnicos:** regra técnica descrita no swagger que inviabiliza o funcionamento da API. Ex.: pattern não permite informar todos os valores requisitados pelo campo.

- **Requisitos negociais:** regra negocial apresentada de forma incorreta ou ausente no swagger que inviabiliza o funcionamento da API. Ex.: para um ramo um dado pode ser obrigatório para o fluxo funcionar, mas para outros ramos o dados pode ser opcional.

- **Requisitos de segurança:** Vulnerabilidades de segurança que podem expor dados, clientes, participantes etc.

- **Requisitos regulatórios:** Alteração de normas, regulações, circulares etc., que por força maior, exigem a atualização dos swaggers OPIN. Ex.: alteração da composição do CNPJ

O GT de Dados definirá o **período de convivência entre as versões**, de acordo com a criticidade e ao impacto da mudança, a fim de possibilitar a adequação dos participantes sem comprometer a estabilidade do ambiente.

Para alterações Minor ou Patch lançadas tanto em release de melhorias quanto em hotfix, não há necessidade de períodos de convivência por serem alterações que não quebram o contrato de dados, conforme definições apresentadas na seção *“6.2 Tipos e enquadramento do versionamento”*.

6.5. Versionamento na URL de API

No ecossistema do Open Insurance a versão Major da API e, somente a versão major, deve constar na URL. Vide exemplos abaixo:

- https://api.organizacao.com.br/open-insurance/dynamic-fields/**v1**
- https://api.organizacao.com.br/open-insurance/consents/**v2**
- https://api.organizacao.com.br/open-insurance/quote-patrimonial/**v1**

As versões Minor e Patch, não devem constar na URL. Isso favorece a compatibilidade da URL com versões anteriores dentro da mesma major.

Sendo assim, devem ser atualizadas as rotas de uma API com uma nova versão somente quando essa API, receber uma alteração Major.

7. Governança de Versionamento

Nesse tópico é descrito como a Estrutura do Open Insurance conduz os processos relacionados ao versionamento de APIs, bem como as demais definições relacionadas a Governança de Versionamento.

Vale ressaltar que, o responsável por fazer a manutenção e promover melhorias nesse documento, nos processos aqui elencados e principalmente nas especificações do OPIN é o Grupo Técnico de Dados, no papel de dono das especificações do Open Insurance.

No que se refere a atualização das especificações e demais documentações correlatas, a responsabilidade é incumbida ao Suporte Técnico N2, que deverá suportar o GT de Dados nas demandas operacionais, no papel de Arquiteto de APIs.

7.1 Modelos de priorização de alterações

As alterações solicitadas nos contratos de dados do Open Insurance, devem ser priorizadas conforme os modelos abaixo:

- **Release de melhorias**: esse é um modelo de priorização planejado. As alterações solicitadas que se enquadram aqui, são inseridas no backlog de ajustes devido a possuírem baixa criticidade ou solução de contorno. Os ajustes inseridos no backlog são

priorizados e ajustados nas APIs em cronograma definido pelo GT de Dados e deliberado pelo CDOI.

- **Hotfix:** esse modelo permite que sejam realizadas alterações emergenciais nos swaggers que atualmente estão em produção. São alterações que não possuem contorno e podem impedir que o ecossistema continue funcionando como esperado. Para executar esse fluxo, é necessário que o CDOI delibere a priorização imediata do ajuste.

- **Swaggers em desenvolvimento:** versões em desenvolvimento pelo ecossistema, ou seja, que não estão em produção (Developing, Release Candidate e Certifying), podem sofrer alterações sem a realização de hotfix. Essas alterações necessitam:

    - Ser aprovadas pelo GT de Dados, como dono das especificações.
    - Ser comunicadas as participantes, caso o status da versão seja Release Candidate ou Certifying.

7.2 Análise e Priorização da solicitação de alterações

Para alterar uma especificação é necessário que a solicitação seja feita via Service Desk da Estrutura OPIN, visando garantir a rastreabilidade dos ajustes realizados.

O Suporte Técnico da Estrutura realiza a triagem das solicitações de alteração recebidas e valida se é um ajuste mapeado, ou se é uma alteração a ser discutida no GT de Dados. Vale ressaltar que, alterações que envolvam regras negociais ou revisão de itens regulatórios poderão necessitar do apoio dos subgrupos negociais de dados ou do regulador respectivamente.

O Suporte Técnico da Estrutura, também identifica o impacto e abrangência das alterações solicitadas, que embasam propostas técnicas para atender o ajuste solicitado.

Compete ao GT de Dados definir sobre a pertinência da alteração e a proposta técnica a ser adotada. Quando pertinente, a mudança deve ser alocada em uma release de melhorias ou em um fluxo emergencial, de acordo com o impacto e a urgência. Essa análise também é realizada pelo GT.

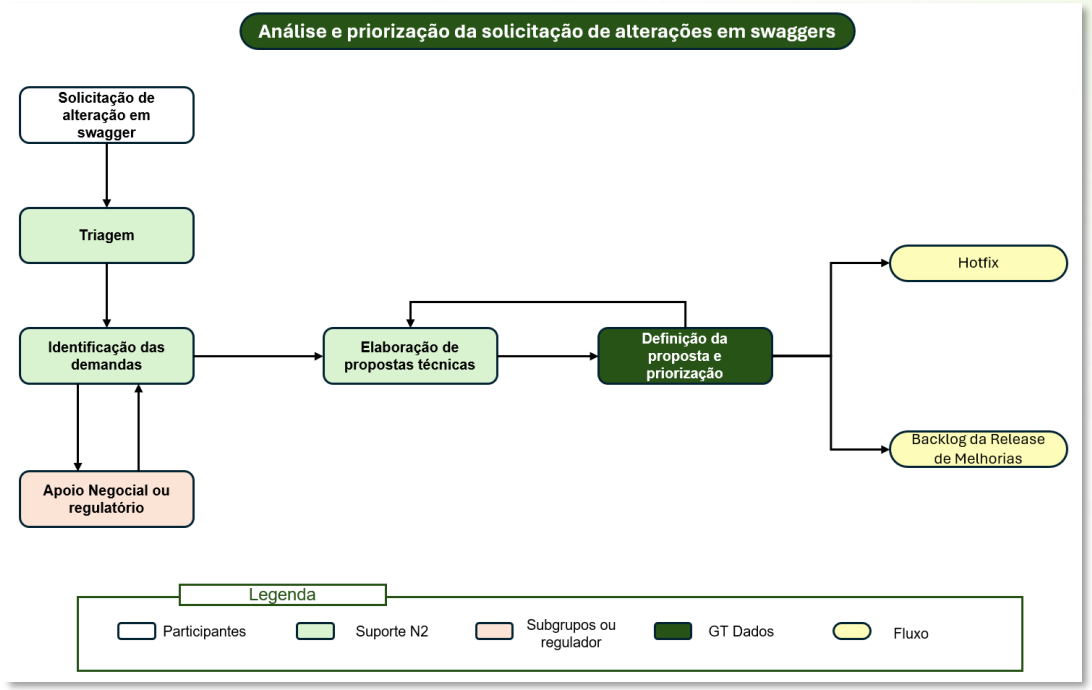

Figura 2 – Macroprocesso análise e priorização de alterações

7.2.1 Fluxo de versionamento padrão (Release de Melhorias)

O fluxo de versionamento padrão, previsto para aprimorar as especificações do Open Insurance é realizado pelo menos 1 vez ao ano. O objetivo é promover alterações em massa nos swaggers do Open Insurance - desde que exista backlog a ser priorizado.

As alterações acatadas pelo GT de Dados que possuem solução de contorno ou que não são urgentes, são incluídas no backlog de ajustes a serem priorizados e disponibilizadas às participantes no backlog no portal do desenvolvedor.

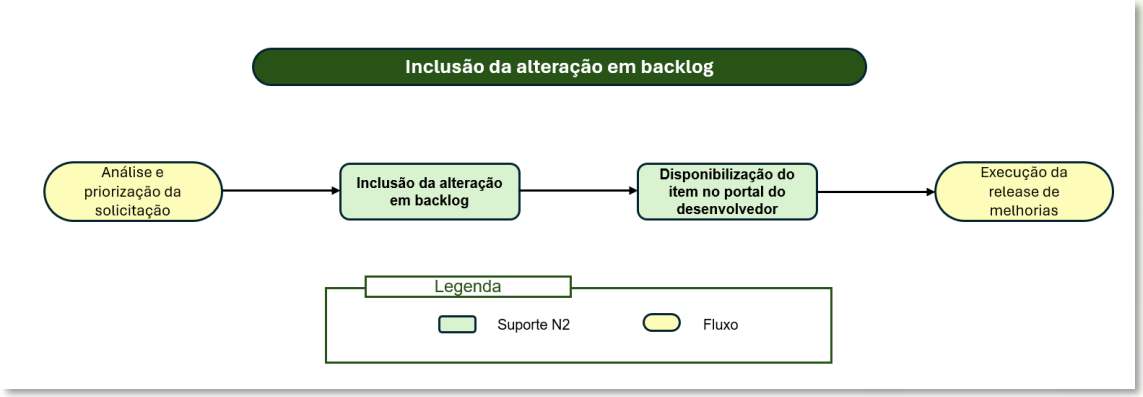

Figura 3 – Macroprocesso inclusão de alterações em backlog

Existindo backlog a ser priorizado, o GT de Dados deverá definir uma data de corte para inclusão de novas alterações na release de melhorias em discussão, isso fornece previsibilidade do bloco de alterações que poderá ser priorizado de acordo com a avalição do Grupo Técnico.

Dada a análise de complexidade e a quantidade dos itens priorizados do backlog, o GT deverá estabelecer o cronograma de atualização das especificações, do ambiente e da certificação das participantes.

Após a definição do GT, o cronograma deverá ser encaminhado ao CDOI (Conselho Deliberativo do Open Insurance) para deliberação e vigência como cronograma oficial da release de melhorias.

O cronograma deve conter no mínimo as seguintes etapas:

- Atualização as especificações – **Suporte Técnico N2**
- Estabilização das ferramentas de conformidade e testes - **Fornecedores**
- Certificação das novas verões - **Participantes**
- Entrada em produção das novas versões - **Participantes**
- Convivência de versões Major – **Participantes**

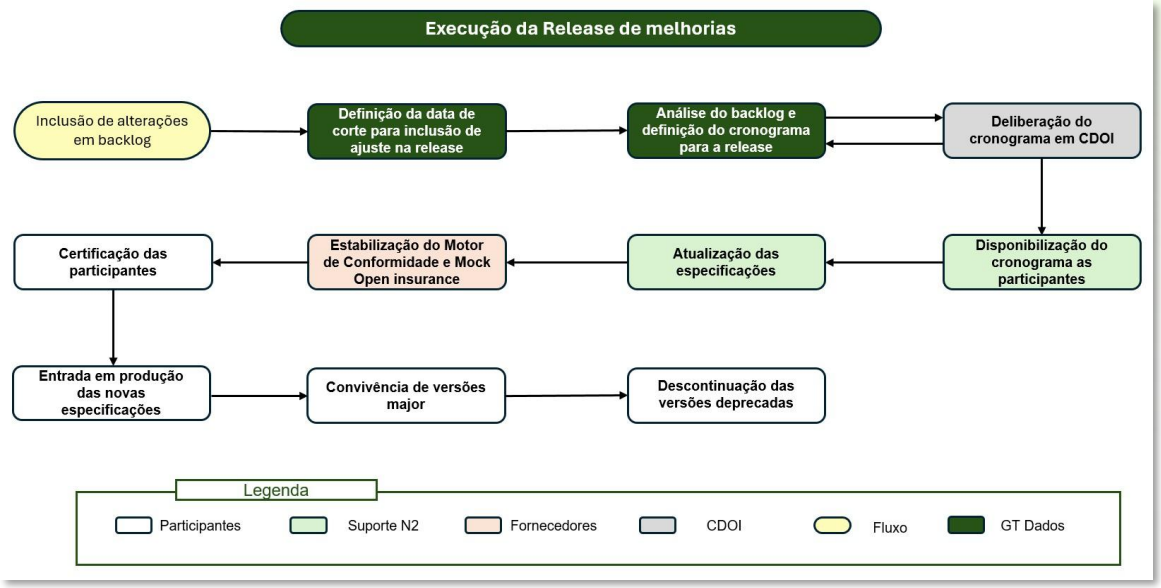

Figura 4– Macroprocesso de execução da release de melhorias

7.2.2 Fluxo de versionamento emergencial (Hotfix)

Se for identificado pelo GT de Dados a necessidade de aplicar uma alteração emergencial em uma ou mais especificações, a proposta técnica definida como solução, deverá ser encaminha para deliberação em CDOI.

Assim que a hotfix for deliberada, o Suporte Técnico N2 deverá atualizar as especificações e disponibilizá-las no portal do desenvolvedor. Os novos swaggers também devem ser encaminhados aos fornecedores das ferramentas de conformidade e testes, para que essas sejam aprimoradas.

Uma vez que o ambiente estiver preparado para certificar as participantes, o Secretariado deverá comunicar as participantes das novas versões de swaggers disponibilizadas, bem como se exigirá a certificação ou não das participantes.

Caso seja uma alteração emergencial major (hotfix major), também deverá ser informado ao ambiente o periodo de convivência estipulado para as versões e APIs envolvidas no fluxo de correção.

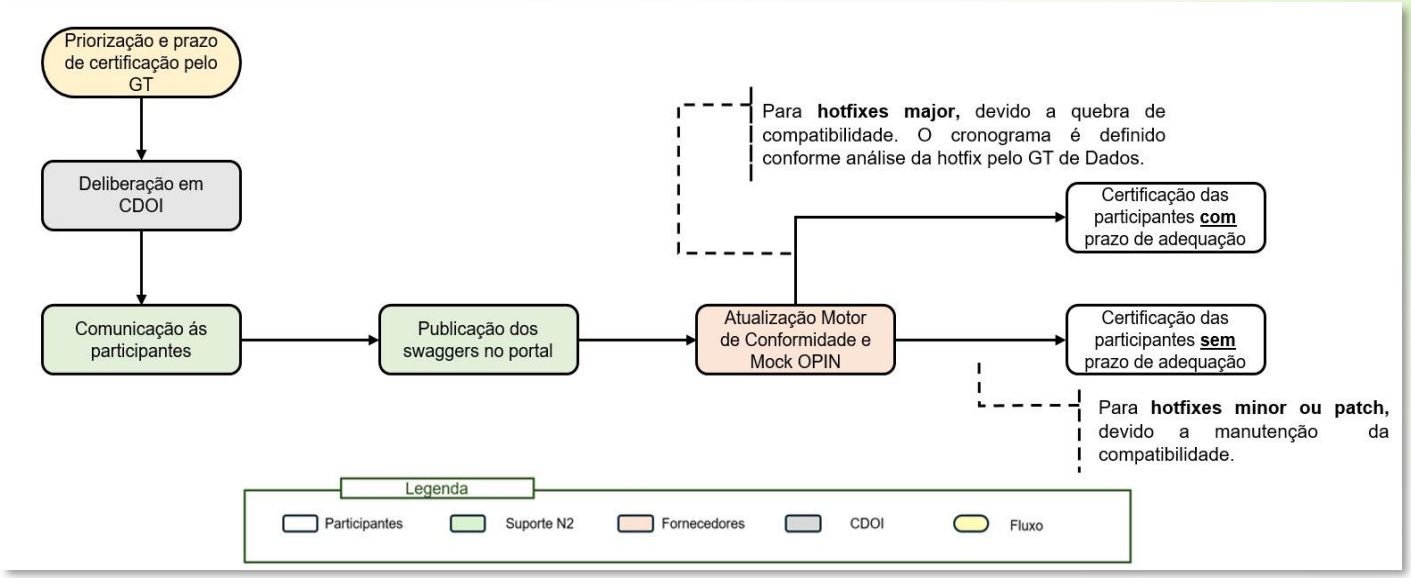

Figura 5 – Macroprocesso de execução de hotfix