INSTITUTO FEDERAL DE RONDÔNIA
***CAMPUS* PORTO VELHO CALAMA**
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JOÃO VICTOR PENHA ALVES

XML FORGE: FERRAMENTA DE APOIO PARA DESENVOLVEDORES COM FUNÇÕES DE FORMATAÇÃO, CONVERSÃO E VALIDAÇÃO DE XML

Porto Velho/RO
2025

JOÃO VICTOR PENHA ALVES

XML FORGE: FERRAMENTA DE APOIO PARA DESENVOLVEDORES COM FUNÇÕES DE FORMATAÇÃO, CONVERSÃO E VALIDAÇÃO DE XML

Artigo entregue como Trabalho de Conclusão de Curso ao Instituto Federal de Educação, Ciência e Tecnologia de Rondônia (IFRO), *Campus* Porto Velho Calama, como requisito parcial para obtenção do grau de Tecnólogo, junto ao Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Celso Guedes Gomes

Porto Velho/RO
2025

Ficha catalográfica elaborada pelo Sistema Gerador de Ficha Catalográfica do IFRO.

Alves, João Victor Penha.
XML Forge: Ferramenta de Apoio para Desenvolvedores com Funções de Formatação, Conversão e Validação de XML / João Victor Penha Alves. - Porto Velho, 2025.
21 f. : il.

Orientador(a): Prof. Celso Guedes Gomes.

Trabalho de Conclusão de Curso (Superior de Tecnologia em Análise e Desenvolvimento de Sistemas) – Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO, Porto Velho, 2025.

1. XML. 2. Extensão Web. 3. Conversão. 4. JSON. 5. XSD. I. Gomes, Celso Guedes (orient.). II. Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO. III. Título.

Bibliotecário(a) Responsável: Miria Santana Veiga, CRB-11/898

JOÃO VICTOR PENHA ALVES

XML FORGE: FERRAMENTA DE APOIO PARA DESENVOLVEDORES COM FUNÇÕES DE FORMATAÇÃO, CONVERSÃO E VALIDAÇÃO DE XML

A banca examinadora, abaixo listada, aprova o Trabalho de Conclusão de Curso “XML Forge: Ferramenta de Apoio para Desenvolvedores com Funções de Formatação, Conversão e Validação de XML” elaborado por “JOÃO VICTOR PENHA ALVES” como requisito parcial para obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas, pelo Instituto Federal de Educação, Ciência e Tecnologia de Rondônia.

Porto Velho/RO, 26/11/2025

Comissão Examinadora

gov.br
Documento assinado digitalmente
CELSO GUEDES GOMES
Data: 01/12/2025 14:17:55-0300
Verifique em https://validar.iti.gov.br

Prof. Celso Guedes Gomes - IFRO
(Orientador)

LEANDRO FERRAREZI VALIANTE:82694095291

Assinado de forma digital por LEANDRO FERRAREZI VALIANTE:82694095291
Dados: 2025.12.01 14:26:38 -04'00'

Prof. Leandro Ferrarezi Valiante - IFRO

gov.br
Documento assinado digitalmente
CLEYTON PEREIRA DOS SANTOS
Data: 01/12/2025 14:26:40-0300
Verifique em https://validar.iti.gov.br

Prof. Cleyton Pereira dos Santos - IFRO

XML FORGE: FERRAMENTA DE APOIO PARA DESENVOLVEDORES COM FUNÇÕES DE FORMATAÇÃO, CONVERSÃO E VALIDAÇÃO DE XML

RESUMO: O presente trabalho aborda os desafios persistentes na manipulação de documentos XML, como a baixa legibilidade de arquivos desestruturados, a necessidade de interoperabilidade com JSON e a garantia de conformidade via XSD. O objetivo geral foi projetar e implementar a extensão ”XML Forge”para o Google Chrome, visando apoiar desenvolvedores, analistas e estudantes nesta tarefa. Utilizando uma metodologia de prototipagem evolutiva e a arquitetura *Manifest V3* , a ferramenta implementou com sucesso os objetivos específicos, incluindo um formatador de código (*pretty print*) , um conversor bidirecional XML e JSON, um validador de esquemas XSD e um módulo de comparação diferencial (*diff*). Os resultados demonstram que a ”XML Forge”constitui uma solução funcional e prática que resolve os problemas identificados, consolidando o aprendizado do ciclo de engenharia de software.

PALAVRAS-CHAVE: XML. Extensão Web. Conversão. JSON. XSD.

ABSTRACT: This paper addresses the persistent challenges in handling XML documents, such as the low readability of unstructured files, the need for interoperability with JSON, and ensuring compliance via XSD. The general objective was to design and implement the ”XML Forge”extension for Google Chrome, aiming to support developers, analysts, and students in this task. Using an evolutionary prototyping methodology and the Manifest V3 architecture, the tool successfully implemented the specific objectives, including a code formatter (pretty print), a bidirectional XML and JSON converter, an XSD schema validator, and a differential comparison (diff) module. The results demonstrate that ”XML Forge”constitutes a functional and practical solution that solves the identified problems, consolidating the learning of the software engineering cycle.

KEYWORDS: XML. Web Extension. Conversion. JSON. XSD.

1 INTRODUÇÃO

A interconexão crescente dos sistemas de tecnologia da informação impõe desafios persistentes na troca de dados estruturados entre plataformas heterogêneas, especialmente para desenvolvedores e analistas. O XML (Extensible Markup Language), concebido como uma linguagem flexível para a representação hierárquica e semântica de informações, estabeleceu-se como um padrão indispensável em ambientes legados e integrações críticas, incluindo serviços web SOAP e documentos fiscais eletrônicos no contexto brasileiro (Bray *et al.*, 2008). Sua duradoura relevância em setores como o financeiro, governamental e de saúde, nos quais a precisão e a conformidade dos dados são imperativas, persiste apesar do surgimento de tecnologias emergentes. Contudo, na rotina profissional, os arquivos XML são comumente recebidos em formatos desestruturados, caracterizados por indentação inadequada, quebras de linha irregulares e ausência de realces visuais, o que obstrui a leitura manual e a identificação de erros no processo de desenvolvimento ou manutenção de sistemas (Pressman; Maxim, 2021). Tal desorganização resulta em ineficiências temporais substanciais, necessidade de retrabalho e, em situações extremas, falhas de interoperabilidade que comprometem a qualidade integral do software, elevando custos e diminuindo a confiabilidade sistêmica.

Ademais, o avanço acelerado das arquiteturas de software contemporâneas tem favorecido a preferência pelo JSON (JavaScript Object Notation) em APIs RESTful e interfaces web *front-end*, graças à sua estrutura compacta, clareza conceitual e compatibilidade direta com *parsers* em JavaScript (Crockford, 2006). Essa coexistência entre o XML, predominante em infraestruturas legadas, e o JSON, central em plataformas atuais, impõe obstáculos extras às equipes responsáveis pela migração e integração de dados, demandando soluções robustas para conversões em ambas as direções.

Nesse cenário, a verificação de documentos XML conforme esquemas XSD (XML Schema Definition) configura-se como um elemento indispensável, ao assegurar que os dados se alinhem integralmente às exigências normativas e aos modelos esquemáticos definidos, mitigando assim eventuais desvios que coloquem em risco a solidez de bases de dados ou das rotinas operacionais (Fallside; Walmsley, 2004). A carência de recursos técnicos adequados para enfrentar essas exigências agrava a complexidade inerente ao desenvolvimento, fomentando um maior número de inconsistências e comprometendo a produtividade da equipe envolvida.

Nesse contexto, o presente estudo delineia o desenvolvimento da extensão ”XML Forge”para o navegador Google Chrome, concebida com o intuito de apoiar estudantes, desenvolvedores e analistas no processamento de arquivos XML. Essa solu-

ção incorpora um repertório abrangente de recursos fundamentais, abrangendo desde a formatação via *pretty print* aliada ao realce sintático, que eleva substancialmente a clareza de leitura, até a conversão mútua entre XML e JSON, viabilizando a integração entre estruturas de dados e a adaptação a diversos ambientes tecnológicos.

Ademais, contempla a verificação em face de esquemas XSD, com emissão de relatórios exaustivos sobre inconsistências, o que facilita a detecção apurada de não conformidades, o acesso a arquivos locais nos formatos XML e JSON, assegurando versatilidade no tratamento de informações, mecanismos de exportação para a área de transferência ou salvamento em extensões originais .xml e .json, otimizando as etapas operacionais a opção de alternância entre temas escuro e claro, promovendo ergonomia visual conforme as preferências individuais e de maneira particularmente original, um módulo de análise diferencial (*diff*) para dois documentos XML, apresentado em visualização paralela, permitindo a localização imediata de alterações. Projetados para uma interação intuitiva e reativa, esses componentes operam integralmente no âmbito do navegador, dispensando instalações externas, o que confere à ferramenta uma acessibilidade notável e uma eficácia comprovada em cenários de desenvolvimento iterativo, em consonância com as orientações de Sommerville (2011) quanto à relevância de protótipos personalizados na superação de entraves operacionais.

Ao longo deste Trabalho de Conclusão de Curso, tem como foco o objetivo geral de criar uma extensão que simplifique o tratamento de XML, alinhando-se aos objetivos específicos de implementar um formatador eficiente, um conversor preciso, um validador confiável e uma interface amigável. Essa iniciativa não só atende a demandas práticas do mercado, mas também enriquece discussões acadêmicas sobre qualidade de dados, interoperabilidade em sistemas de informação e a aplicação de ferramentas de suporte no ciclo de vida do software, conforme destacado por Pressman e Maxim (2021). A extensão “XML Forge” configura-se como solução prática. e didática, contribuindo para a modernização e eficiência no manuseio de dados estruturados.

1.1 Objetivos

1.1.1 Objetivo geral

Projetar e implementar uma ferramenta de software na forma de uma extensão para o navegador Google Chrome, para apoiar desenvolvedores, analistas e estudantes na manipulação de documentos XML.

1.1.2 Objetivos específicos

Como objetivos específicos, buscam-se

1. Desenvolver um módulo de formatação de código XML que melhore a legibilidade por meio da indentação hierárquica e da aplicação de destaque de sintaxe;
2. Implementar um mecanismo de conversão bidirecional e automatizada entre os formatos XML e JSON, a fim de garantir a interoperabilidade entre sistemas que utilizam diferentes padrões de dados estruturadoss;
3. Projetar um sistema de validação de documentos XML com base em esquemas XSD (XML Schema Definition), fornecendo ao usuário relatórios detalhados sobre erros de conformidade e inconsistências estruturais;
4. Implementar uma interface gráfica para a análise comparativa entre duas versões de um documento XML, destacando visualmente as operações de inserção, remoção e alteração, de modo que o trabalho de verificação se torne mais rápido e menos suscetível a erros;
5. Desenvolver a extensão utilizando a arquitetura Manifest V3, em conformidade com as diretrizes do Google Chrome que visam aprimorar a segurança, a privacidade e o desempenho das extensões para publicação.

2 FUNDAMENTAÇÃO TEÓRICA

O desenvolvimento de um projeto de software requer um roteiro bem definido, pois o processo de desenvolvimento exige um conjunto estruturado de atividades básicas, incluindo comunicação, planejamento, modelagem, construção e entrega (Pressman; Maxim, 2021). Trilhando este princípio, e para contextualizar o desenvolvimento da ferramenta XML Forge, esta seção aborda os conceitos fundamentais que sustentam suas funcionalidades e justificam sua relevância. A seguir, são discutidos a linguagem XML e sua contínua importância, o desafio da interoperabilidade de sistemas frente à ascensão do JSON, a necessidade de validação para garantia da qualidade de dados, a arquitetura de extensões web como plataforma para ferramentas de produtividade e por fim a análise comparativa de documentos estruturados.

2.1 XML (Extensible Markup Language) e Sua Relevância Contínua

De acordo com Harold e Means (2004), em sua obra “XML in a Nutshell” frequentemente referida como a base essencial sobre o tema, o XML (Extensible Markup

Language) configura-se como uma metalinguagem de marcação destinada a descrever, armazenar e transportar dados de maneira estruturada, acessível tanto a máquinas quanto a usuários humanos. Sua essência está na extensibilidade, que possibilita a elaboração de vocabulários personalizados para domínios específicos, adaptando-se a necessidades variadas sem rigidez imposta. Essa visão é complementada por Ramalho (1999), que destaca, como traço distintivo herdado do SGML (Standard Generalized Markup Language), a separação inequívoca entre o conteúdo e sua apresentação visual. Tal princípio direciona o foco exclusivo para a estrutura semântica da informação, independentemente de sua renderização, o que confere aos documentos XML uma natureza autodescritiva. Essa propriedade os posiciona como solução privilegiada para a intercâmbio de dados entre sistemas heterogêneos e aplicações diversas, promovendo uma representação independente de plataformas.

Mesmo com o advento de formatos mais concisos e ágeis, o XML preserva sua proeminência em integrações críticas de sistemas, particularmente no contexto brasileiro. Dada a área, sua aplicação é vista no Sistema Público de Escrituração Digital (SPED), onde serve de base para documentos fiscais eletrônicos a exemplo da Nota Fiscal Eletrônica (NF-e), conforme regulamentações da Secretaria da Fazenda (BRASIL, 2005). O Projeto NF-e tem como objetivo instituir um padrão nacional de emissão de documentos fiscais digitais, substituindo o modelo em papel por um processo validado por assinatura digital do emissor, o que confere a validade jurídica. Essa iniciativa não apenas simplifica as obrigações acessórias dos contribuintes, mas também habilita o monitoramento em tempo real das transações comerciais pela autoridade fiscal, otimizando o controle tributário e reduzindo fraudes (BRASIL, 2005). Assim, o XML demonstra sua resiliência como tecnologia madura, alinhada a demandas regulatórias e operacionais contemporâneas.Entretanto, apesar dessa relevância, profissionais ainda enfrentam dificuldades para manipular documentos XML desestruturados, reforçando a necessidade de ferramentas de apoio.

2.2 Interoperabilidade de Sistemas e a Ascensão do JSON

A interoperabilidade é uma propriedade de dois ou mais sistemas interagirem e trocarem dados, baseados em métodos definidos, com a finalidade de obterem os resultados esperados (Santos, 2010).

Segundo Mucheroni e Modesto (2011), a interoperabilidade desempenha um papel essencial ao aprimorar a conectividade entre diferentes agentes, como empresas, consumidores, instituições financeiras e órgãos públicos, possibilitando um fluxo de informações eficiente e apoiando operações relacionadas ao comércio eletrônico.

A necessidade de integrar sistemas baseados em XML com novas soluções que

utilizam JSON é uma realidade constante para desenvolvedores no cenário atual. A migração de arquiteturas SOAP para microsserviços baseados em REST, por exemplo, frequentemente exige uma camada de tradução entre os dois formatos. Com isso, a conversão de dados não é apenas uma conveniência, mas um requisito fundamental para garantir a comunicação, a coexistência e a evolução gradual em arquiteturas de sistemas modernas.

2.3 Validação e Qualidade de Dados com XSD

A natureza flexível do XML, embora vantajosa para a representação de dados complexos, impõe a necessidade de controles rigorosos para que os documentos permaneçam alinhados a padrões normativos, salvaguardando a integridade e a confiabilidade das informações manipuladas. Nesse panorama, o XML Schema Definition (XSD), formalizado como recomendação pelo World Wide Web Consortium (W3C), emerge como um mecanismo essencial, funcionando a exemplo de uma planta baixa que traça os contornos arquiteturais de um documento XML abrangendo a definição de elementos, atributos e os tipos de dados tolerados. Tal abordagem não apenas impõe uma disciplina estrutural, mas também eleva a robustez dos sistemas que dependem de trocas padronizadas.

Pressman e Maxim (2021) enfatizam que a validação de insumos representa um pilar fundamental na garantia de qualidade, atuando como barreira contra a propagação de dados falhos, os quais poderiam desencadear instabilidades operacionais, falhas em processamento e resultados enviesados. Diante disso, submeter um documento XML à análise de seu esquema XSD associado torna-se imperativo em contextos de integração delicados, onde a detecção precoce de discrepâncias não só atenua riscos latentes, mas também assegura que as interações de dados honrem fielmente as especificações formais e as demandas contratuais do negócio, fomentando assim uma interoperabilidade sustentável.

2.4 Arquitetura de Extensões Web como Ferramenta de Produtividade

Como observado por Pantelaios e Kapravelos (2024), extensões de navegador são componentes integrais da experiência de navegação que aprimoram a produtividade e a segurança do usuário. A arquitetura de uma extensão é composta por arquivos JavaScript (JS), HTML e CSS, além de um arquivo de configuração central conhecido como `manifest.json`, que especifica as permissões da extensão. A principal vantagem de encapsular ferramentas de desenvolvimento em uma extensão reside na integração direta com o fluxo de trabalho do profissional. Os autores explicam que essas extensões são de particular interesse devido ao seu acesso a APIs de alto privi-

légio, que as capacitam a modificar páginas web. No padrão mais recente, o Manifest V3, a arquitetura foi alterada com o objetivo de reforçar a privacidade, a segurança e o desempenho. Conforme detalhado por (Pantelaios; Kapravelos, 2024), uma mudança fundamental foi a troca dos *scripts* que rodavam em segundo plano pelos chamados trabalhadores de serviço, o que impacta diretamente a forma como os desenvolvedores constroem suas ferramentas.

2.5 Análise Comparativa de Documentos Estruturados (*Diff*)

A identificação de diferenças entre duas versões de um documento estruturado é uma tarefa que transcende a comparação textual linha a linha. Para dados hierárquicos como o XML, uma abordagem eficaz deve operar sobre a representação em árvore dos documentos. Os usuários não se interessam somente pelos valores atuais dos dados, mas também pelas mudanças (Santos; Hara, 2007).

Cecchin e Hara (2009) destacam que, devido às constantes mudanças nos dados publicados na web, ferramentas de *diff* tornam-se fundamentais para que usuários e administradores monitorem e controlem as diferenças entre versões de documentos XML. O resultado é uma comparação semanticamente relevante, essencial para depurar alterações em arquivos de configuração, rastrear mudanças em respostas de serviços ou auditar transações de dados.

3 METODOLOGIA

O presente capítulo apresenta o fluxo de desenvolvimento adotado no Trabalho de Conclusão de Curso. Foi adotada uma estratégia metodológica mista, fundindo conceitos da engenharia de software clássica a métodos ágeis, distribuída em dois momentos principais: a coleta de requisitos e o progresso gradual na construção. Essas divisões mostraram-se cruciais para validar que o resultado final se adequasse às expectativas concretas dos usuários, sobretudo em um cenário de criação de extensão para navegador. Priorizaram-se rodadas rápidas de ajuste para elevar capacidades como o embelezamento de XML e a verificação através de XSD, em um protótipo erguido de modo independente. Essa opção metodológica fundamenta-se na prototipagem evolutiva defendida por Pressman e Maxim (Pressman; Maxim, 2021), a qual favorece adaptações baseadas em avaliações práticas. Tal abordagem encontra-se alinhada ao escopo da ferramenta e ao paradigma de desenvolvimento incremental delineado por Sommerville (2011), que prioriza a montagem gradual de módulos para minimizar incertezas.

O processo de levantamento de requisitos baseou-se em uma análise qualita-

tiva de fontes bibliográficas e na identificação de desafios cotidianos no tratamento de XML. Destacaram-se necessidades como a conversão mútua entre XML e JSON e a melhoria da legibilidade, visto que arquivos XML mal estruturados representam um obstáculo significativo para os profissionais. Para a execução do projeto, procedeu-se a uma adaptação da metodologia padrão, onde a fase tradicional de comunicação foi substituída pela análise técnica, definindo o escopo pela observação direta de demandas de desenvolvimento e não por interações com *stakeholders* externos. As demais fases do ciclo de engenharia permaneceram inalteradas.

Foi realizada uma análise, da documentação oficial da API de Extensões do Google Chrome, com foco no Manifest V3. Com base nesse estudo, elaborou-se um catálogo contendo 15 requisitos funcionais e não funcionais que orientaram o desenvolvimento da solução. Na etapa de *design*, foi implementado uma estrutura arquitetural modular, composta por camadas de interface frontal, processamento lógico e conexão com o navegador, empregando um *popup* interativo construído com HTML5, CSS3 e JavaScript puro para promover fluxos de uso naturais. Quanto à conversão de formatos, optou-se por bibliotecas leves como xml2js, enquanto a validação de XSD recorreu ao parser DOMParser nativo, suplementado pela libxmljs adaptada para web.

O desenvolvimento avançou por meio de iterações semanais, utilizando o Visual Studio Code para codificação e o GitHub para controle de versões. Iniciou-se pelo módulo de formatação, que aplica recursão sobre a árvore DOM para indentação. Após ser avaliado em documentos XML, prosseguiu-se com a conversão bidirecional, incorporando ajustes para namespaces. Posteriormente, foi implementado o comparador diferencial baseado no algoritmo de *Longest Common Subsequence*, que exibe diferenças em painéis paralelos com marcações visuais em cores distintas para inserções, exclusões e modificações. Todo o desenvolvimento manteve a conformidade com o Manifest V3 por meio de permissões restritas que preservam a privacidade e executam operações exclusivamente no navegador. As atividades de teste abrangeram verificações unitárias em cenários diversos, como XML mal formatados, além de testes de integração no ambiente de depuração do *Chrome DevTools*.

A aferição da efetividade da ”XML Forge”, alcançada nesta dinâmica iterativa de três meses, não se limitou aos testes funcionais. A validação foi conduzida de forma empírica por meio da aplicação direta da ferramenta diretamente contra os desafios cotidianos que motivaram sua criação. Isso incluiu a submissão de arquivos XML complexos e desestruturados ao formatador e a verificação da integridade dos dados em conversões bidirecionais de XML para JSON, dentre as outras funcionalidades. O sucesso na resolução desses casos de uso práticos, alinhados aos requisitos levantados e observados no fluxo de trabalho, serviu como métrica principal. Dessa forma, a efetividade foi demonstrada concretamente, com a ferramenta provando ser uma solução

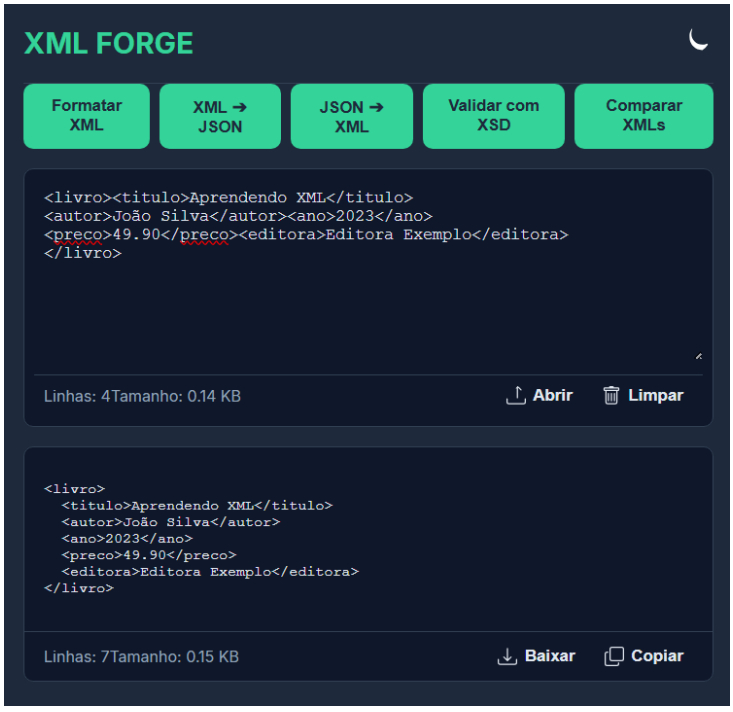
viável baseada em navegador para superar as barreiras de legibilidade e interoperabilidade no tratamento de dados.

4 RESULTADOS E DISCUSSÃO

Este capítulo apresenta os resultados concretos obtidos com o desenvolvimento da extensão ”XML Forge”, demonstrando como as funcionalidades implementadas atendem diretamente aos problemas e objetivos delineados neste trabalho. A discussão foca na validação dos requisitos através da própria entrega funcional da ferramenta, alinhada à metodologia de prototipagem evolutiva adotada.

A Figura 1 demonstra a capacidade central de formatação e realce de sintaxe. Esta funcionalidade aborda diretamente o problema da baixa legibilidade de arquivos XML em formatos desestruturados, uma ineficiência comum na rotina de desenvolvimento. Ao aplicar a indentação hierárquica e o realce de cores, a ferramenta cumpre o objetivo específico (1), transformando um documento de difícil leitura manual em uma estrutura clara e organizada, facilitando a depuração e a análise.

Figura 1 – Formatação de XML



Fonte: Elaborado pelo autor.

Seguindo para o desafio da interoperabilidade, a Figura 2 exibe o módulo de conversão bidirecional entre XML e JSON. Esta ferramenta foi projetada para resolver a coexistência entre sistemas legados, que frequentemente utilizam XML, e arquiteturas

modernas baseadas em APIs RESTful, que favorecem o JSON. A implementação de um mecanismo de conversão automatizado é preciso, que permite ao usuário carregar, converter e copiar o resultado, atende integralmente ao objetivo específico (2).

Figura 2 – Conversão Bidirecional

(a) XML para JSON

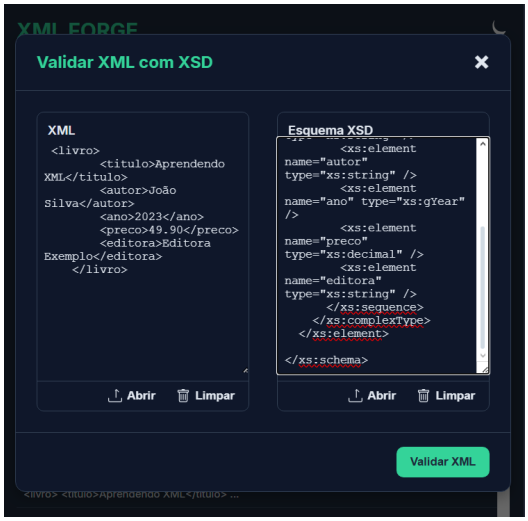
(b) JSON para XML

Fonte: Elaborado pelo autor.

Para garantir a qualidade e a conformidade dos dados, foi implementado o sistema de validação de documentos XML com base em esquemas XSD, como ilustrado na Figura 3 . Esta funcionalidade é crucial, pois assegura que os dados estejam alinhados às exigências normativas e modelos definidos, mitigando riscos de interoperabilidade. Ao fornecer ao usuário um relatório claro sobre inconsistências ou erros de conformidade, a extensão cumpre o objetivo específico (3), atuando como um mecanismo de controle de qualidade no fluxo de desenvolvimento.

Figura 3 – Validação XSD

(a) XML e XSD



(b) Resultado da Validação

Fonte: Elaborado pelo autor.

Um dos recursos mais distintivos da ”XML Forge”é o módulo de análise comparativa, apresentado na Figura 4. A interface exibe dois documentos XML em visualização paralela, destacando visualmente as operações de inserção, remoção e alteração. Esta funcionalidade, que atende ao objetivo específico (4), oferece um ganho de produtividade substancial, automatizando a tarefa de auditar mudanças entre versões de arquivos, que manualmente é demorada e suscetível a erros.

Figura 4 – Comparação de XML

(a) XML Original e Modificado

(b) Resultado da Comparação

Fonte: Elaborado pelo autor.

Finalmente, todo o projeto foi desenvolvido em conformidade com as diretrizes do Google Chrome, utilizando a arquitetura Manifest V3. Isso garante que a extensão não apenas atenda aos seus requisitos funcionais, mas também esteja alinhada com os padrões modernos de segurança, privacidade e desempenho exigidos para publicação. Em suma, os resultados demonstram que cada objetivo geral e específico foi traduzido em uma funcionalidade tangível e operacional dentro da extensão ”XML Forge”. A ferramenta desenvolvida constitui uma solução prática e eficaz para os desafios diários no manuseio de XML, validando a proposta deste estudo através de sua própria implementação e capacidade de resolver os problemas identificados.

5 CONCLUSÃO

A elaboração do presente Trabalho de Conclusão de Curso mostrou uma etapa fundamental para a consolidação das competências do graduando, possibilitando a aplicação prática do conhecimento adquirido desde o levantamento de requisitos até o desenvolvimento final da extensão web. Este processo ofereceu uma valiosa reflexão sobre os pontos de evolução e limitações, resultando assim em um sistema que, embora cumpra os objetivos propostos de facilitar o tratamento e a validação de documentos XML por meio de uma extensão acessível, leve e integrada ao navegador Google Chrome, também evidencia o potencial de aplicação de tecnologias web modernas na resolução de desafios reais enfrentados por profissionais da área de desenvolvimento de software.

A experiência obtida durante o desenvolvimento da extensão ”XML Forge”, resultou em uma compreensão abrangente do ciclo de engenharia de software, da concepção de requisitos à validação dos resultados. Esta experiência prática confirmou o valor da prototipagem incremental e das metodologias ágeis para a otimização de processos técnicos. Dessa forma, este Trabalho de Conclusão de Curso não representa apenas o fim de uma jornada acadêmica, mas o início de uma carreira profissional fundamentada na inovação, no aprendizado constante e no compromisso com a qualidade de software.

omo continuidade do estudo, está prevista a publicação da extensão XML Forge na Chrome Web Store, deixando-a disponível para um público mais amplo e consolidando seu uso no meio acadêmico e profissional. Além disso, também estão previsto melhorias de acessibilidade e usabilidade, como suporte a leitores de tela e otimizações de desempenho para o processamento de arquivos xml volumosos. Essas ações visam não apenas expandir o alcance da ferramenta, mas também reforçar seu papel como uma contribuição prática e evolutiva para o ecossistema de desenvolvimento web e o ensino de tecnologias de marcação estruturada.

REFERÊNCIAS

BRASIL. CONSELHO NACIONAL DE POLÍTICA FAZENDÁRIA (CONFAZ). **Ajuste SINIEF n. 07, de 30 de setembro de 2005: Institui a Nota Fiscal Eletrônica – NF-e**. Brasília, DF: Secretaria da Fazenda, 2005. Publicada no Diário Oficial da União de 3 de outubro de 2005. Disponível em: https://www.confaz.fazenda.gov.br/legislacao/ajustes-sinief/2005/as07_05. Acesso em: 15 set. 2024.

BRAY, T. *et al.* **Extensible Markup Language (XML) 1.0 (Fifth Edition)**. Nov. 2008. W3C Recommendation. Disponível em: https://www.w3.org/TR/xml/. Acesso em: 30 abr. 2025.

CECCHIN, Frantchesco; HARA, Carmem Satie. Uma interface gráfica para algoritmos de detecção de diferenças entre documentos XML. *In:* ESCOLA Regional de Banco de Dados (ERBD). 2009. Disponível em: https://www.inf.ufpr.br/carmem/pub/erbd09.pdf. Acesso em: 12 out. 2025.

CROCKFORD, D. **The application/json media type for JavaScript Object Notation (JSON)**. Jul. 2006. RFC 4627. Disponível em: https://datatracker.ietf.org/doc/html/rfc4627. Acesso em: 30 abr. 2025.

FALLSIDE, D. C.; WALMSLEY, P. **XML Schema Part 0: Primer (Second Edition)**. Out. 2004. W3C Recommendation. Disponível em: https://www.w3.org/TR/xmlschema-0/. Acesso em: 30 abr. 2025.

HAROLD, E. R.; MEANS, W. S. **XML in a Nutshell**. 3. ed. [S. l.]: O'Reilly Media, 2004.

MUCHERONI, Marcos; MODESTO, Fernando. A interoperabilidade dos sistemas de informação sob o enfoque da análise sintática e semântica de dados na WEB. **Ponto de Acesso**, v. 5, n. 1, p. 3–18, 2011. Disponível em: https://www.eca.usp.br/acervo/producao-academica/002183369.pdf. Acesso em: 8 set. 2025.

PANTELAIOS, Nikolaos; KAPRAVELOS, Alexandros. Manifest v3 unveiled: Navigating the new era of browser extensions. **arXiv preprint**, 2024. eprint: 2404.08310. Disponível em: https://arxiv.org/abs/2404.08310.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de Software: Uma Abordagem Profissional**. 9. ed. São Paulo: McGraw-Hill, 2021.

RAMALHO, José Carlos Leite. Extensible Markup Language (XML): a promessa e a esperança? *In:* ENCONTRO Nacional de Informática na Educação – EINE'99. Fortaleza; João Pessoa; Salvador, 1999. [S.l.: s.n.] Disponível em: https://hdl.handle.net/1822/680. Acesso em: 7 set. 2025.

SANTOS, E. M. Desenvolvimento e implantação da arquitetura e-Ping. *In:* MESQUITA, C. S. F.; BRETAS, N. L. (ed.). **Panorama da interoperabilidade no Brasil**. Brasil: Ministério do Planejamento / SLTI, 2010.

SANTOS, Rodrigo C.; HARA, Carmem. Utilização de Chaves em Algoritmos de Detecção de Diferenças para XML. *In:* ANAIS do Simpósio Brasileiro de Banco de Dados (SBBD). João Pessoa, PB: Sociedade Brasileira de Computação (SBC), 2007. Disponível em: `https://www.inf.ufpr.br/carmem/pub/sbbd07.pdf`. Acesso em: 12 out. 2025.

SOMMERVILLE, Ian. **Engenharia de Software**. 9. ed. Boston: Pearson Addison-Wesley, 2011.

Apêndices

APÊNDICE A – DOCUMENTO DE ESPECIFICAÇÃO DE REQUISITOS

Apresenta-se a seguir o Documento de Especificação de Requisitos, que detalha o catálogo de funcionalidades e restrições técnicas definidos na fase de concepção, servindo de base para o desenvolvimento e a validação da extensão XML Forge citados no decorrer do trabalho.

Figura 4 – Requisitos Funcionais

3. Requisitos Funcionais (RF)

Identificador	Nome do Requisito	Descrição
RF01	Formatação de XML (Pretty Print)	O sistema deve reordenar documentos XML desestruturados aplicando indentação hierárquica para melhorar a legibilidade.
RF02	Realce de Sintaxe	O sistema deve aplicar cores distintas aos elementos, atributos e valores do código (syntax highlighting) para facilitar a leitura visual.
RF03	Conversão XML para JSON	O sistema deve converter automaticamente estruturas XML para o formato JSON, tratando namespaces e atributos adequadamente.
RF04	Conversão JSON para XML	O sistema deve realizar a conversão inversa, transformando objetos JSON em estruturas XML válidas.

(a) Parte 01

RF05	Validação via XSD	O sistema deve permitir a validação de um código XML contra um esquema XSD (XML Schema Definition) fornecido pelo usuário.
RF06	Relatório de Inconsistências	O sistema deve exibir mensagens de erro detalhadas caso o XML não esteja em conformidade com o XSD.
RF07	Comparação de Arquivos (Diff)	O sistema deve comparar duas versões de um documento XML e identificar inserções, remoções e alterações.
RF08	Visualização Paralela	O sistema deve exibir os arquivos comparados lado a lado, destacando as diferenças visualmente com cores (verde para adição, vermelho para remoção).
RF09	Carregamento de Arquivos	O sistema deve permitir a abertura e leitura de arquivos locais com extensões .xml e .json.
RF10	Download de Resultados	O sistema deve permitir que o usuário salve o código processado (formatado ou convertido) em um arquivo local.
RF11	Cópia Rápida	O sistema deve possuir um botão para copiar o conteúdo da área de texto diretamente para a área de transferência do sistema operacional.
RF12	Alternância de Tema	O sistema deve oferecer modos de visualização Claro e Escuro para ergonomia visual.

(b) Parte 02

Fonte: Elaborado pelo autor.

Figura 5 – Requisitos Não Funcionais

4. Requisitos Não Funcionais (RNF)

Identificador	Nome do Requisito	Descrição
RNF01	Arquitetura Manifest V3	A extensão deve ser construída seguindo a arquitetura Manifest V3 do Google Chrome, utilizando serviços workers em vez de scripts de fundo persistentes.
RNF02	Privacidade e Execução Local	Todo o processamento de dados deve ocorrer no lado do cliente (browser), sem envio de dados para servidores externos.
RNF03	Usabilidade e Acessibilidade	A interface deve ser apresentada em formato de popup leve, dispensando instalações complexas ou configurações de ambiente.

Fonte: Elaborado pelo autor.