

PROTÉGÉ 6.0.0-ALPHA-1: ANÁLISE DA MIGRAÇÃO DO EDITOR DE ONTOLOGIAS PARA A OWL API 5 E DE SUAS IMPLICAÇÕES PARA O ECOSISTEMA DE *PLUGINS*

PROTÉGÉ 6.0.0-ALPHA-1: ANÁLISE DA MIGRAÇÃO DO EDITOR DE ONTOLOGIAS PARA A OWL API 5 E DE SUAS IMPLICAÇÕES PARA O ECOSISTEMA DE *PLUGINS*

 ARK: 44123/multi.v7i14.1798

Henderson Matsuura Sanches

 <https://orcid.org/0000-0003-2354-3393>

 <http://lattes.cnpq.br/8112131443508423>

UniProcessus – Centro Universitário Processus, DF, Brasil

E-mail: henderson.sanches@uniprocessus.edu.br



Resumo:

O *Protégé* é um dos editores de ontologias OWL mais utilizados em pesquisa e na indústria, e sua linha 5.x manteve-se, por cerca de uma década, apoiada na versão 4 da OWL API. Em 28 de setembro de 2026, o projeto publicou a versão 6.0.0-alpha-1, primeira de uma nova linha de desenvolvimento cuja principal mudança é a adoção da OWL API 5. Este artigo tem como objetivo caracterizar essa versão e analisar as implicações da migração para usuários e desenvolvedores de *plugins*. Trata-se de um estudo de caso exploratório, de abordagem qualiquantitativa, baseado em análise documental das notas de versão e em mineração do repositório de código-fonte, comparando as etiquetas 5.6.9 e 6.0.0-alpha-1. Os resultados mostram que a nova versão deriva diretamente da 5.6.9 e que a transição envolveu 33 *commits* e 154 arquivos alterados, dos quais 136 pertencem ao módulo *protege-editor-owl*. A OWL API passou da versão 4.5.29 para a 5.5.1, enquanto o requisito de *Java* 11 e o contêiner *OSGi Apache Felix* 7.0.5 foram mantidos. A migração concentrou-se na substituição do tipo *Optional* da biblioteca *Guava* pelo *java.util.Optional*, na adaptação a métodos baseados em *streams* e na reescrita de implementações do padrão *Visitor*. Foram ainda removidos a infraestrutura de classes definidas anônimas, as preferências de escrita XML e as dependências *JAXB* e *CORBA*. A distribuição alfa empacota apenas o raciocinador *HermiT*, contra 13 *plugins* na versão 5.6.9. Conclui-se que a versão 6.0.0-alpha-1 representa uma modernização arquitetural relevante, porém ainda inadequada para uso em produção, sendo seu sucesso dependente da migração do ecossistema de *plugins*.

Palavras-chave: *Protégé*; OWL API; Ontologias; Evolução de Software; Software Livre.

Abstract:

Protégé is one of the most widely used OWL ontology editors in research and industry, and its 5.x line relied on version 4 of the OWL API for roughly a decade. On September 28, 2026, the project published version 6.0.0-alpha-1, the first release of a new development line whose main change is the adoption of OWL API 5. This article aims to characterize this release and to analyze the implications of the migration for users and plugin developers. It is an exploratory case study with a mixed qualitative and quantitative approach, based on documentary analysis of the release notes and on mining of the source code repository, comparing the 5.6.9 and 6.0.0-alpha-1 tags. The results show that the new release derives directly from 5.6.9 and that the transition involved 33 commits and 154 changed files, 136 of which belong to the protege-editor-owl module. The OWL API moved from version 4.5.29 to 5.5.1, while the Java 11 requirement and the Apache Felix 7.0.5 OSGi container were kept. The migration focused on replacing Guava's Optional type with java.util.Optional, adapting to stream-based methods, and rewriting Visitor pattern implementations. The anonymous defined class infrastructure, the XML writer preferences, and the JAXB and CORBA dependencies were also removed. The alpha distribution bundles only the HermiT reasoner, compared with 13 plugins in version 5.6.9. It is concluded that version 6.0.0-alpha-1 represents a relevant architectural modernization, yet it is not suitable for production use, and its success depends on the migration of the plugin ecosystem.

Keywords: *Protégé; OWL API; Ontologies; Software Evolution; Open Source Software.*

1. Introdução

Ontologias computacionais, entendidas como especificações explícitas e formais de conceitualizações compartilhadas (GRUBER, 1993; STUDER; BENJAMINS; FENSEL, 1998), tornaram-se artefatos centrais na representação de conhecimento, na integração de dados e, mais recentemente, na fundamentação semântica de sistemas de Inteligência Artificial. A linguagem OWL 2, padronizada pelo *World Wide Web Consortium* (W3C), é o formalismo predominante para sua publicação (HITZLER *et al.*, 2012).

A construção dessas ontologias depende de ferramentas especializadas. Entre elas, destaca-se o *Protégé*, editor livre desenvolvido pelo *Stanford Center for Biomedical Informatics Research* cuja origem remonta a 1987 e cuja trajetória acompanhou a evolução da própria área de engenharia do conhecimento (GENNARI *et al.*, 2003; MUSEN, 2015). O *Protégé* é amplamente adotado no ensino e na pesquisa, sendo frequentemente o primeiro contato de estudantes com a modelagem em OWL (NOY; MCGUINNESS, 2001). No contexto brasileiro, a evolução sucessiva do editor tem sido acompanhada por uma série de trabalhos que descrevem os recursos e as aplicações de cada versão, da 5.5.0 à 5.6.9 (SANCHES; DUTRA, 2022; SANCHES, 2024; SANCHES; GOMES, 2024, 2025b; SANCHES, 2026a, 2026b).

Do ponto de vista técnico, o *Protégé* apoia-se na OWL API, biblioteca Java de referência para a manipulação de ontologias OWL (HORRIDGE; BECHHOFFER, 2011). Embora a versão 5 da OWL API esteja disponível há anos, a própria documentação da biblioteca registra que a versão 4 permaneceu em uso por ferramentas como o *Protégé* e que seria mantida enquanto a comunidade não migrasse (OWLAPI, [2026a]). Essa dependência de uma linha mais antiga representava uma dívida técnica para o editor e para seu ecossistema de *plugins*.

Em 28 de setembro de 2026, o projeto publicou a versão 6.0.0-alpha-1, apresentada como a primeira versão alfa da linha de desenvolvimento *Protégé* 6, voltada a testes iniciais e trabalhos de migração e baseada na OWL API 5 (PROTÉGÉ PROJECT,

2026a). Por se tratar de uma mudança de versão principal, que, segundo o versionamento semântico, sinaliza alterações incompatíveis na interface pública (PRESTON-WERNER, [2013]), é razoável supor que ela tenha efeitos sobre usuários, autores de *plugins* e sobre o ensino de engenharia de ontologias.

Diante disso, formula-se a seguinte questão de pesquisa: quais mudanças técnicas a versão 6.0.0-alpha-1 introduz em relação à linha 5.6, e quais as implicações dessas mudanças para usuários e para o ecossistema de *plugins* do *Protégé*?

O objetivo geral deste artigo é caracterizar a versão 6.0.0-alpha-1 do *Protégé* e analisar as implicações de sua migração para a OWL API 5. Os objetivos específicos são: (a) identificar as alterações de dependências e de arquitetura entre as versões 5.6.9 e 6.0.0-alpha-1; (b) quantificar a extensão das modificações no código-fonte e identificar os principais padrões de migração; e (c) analisar os efeitos sobre a distribuição de *plugins* e sobre a adoção da nova versão.

A relevância do estudo decorre do fato de que versões alfa raramente são objeto de análise acadêmica, embora sejam o momento em que decisões arquiteturais se tornam visíveis e em que a comunidade ainda pode influenciá-las. Além desta introdução, o artigo apresenta o referencial teórico (seção 2), os procedimentos metodológicos (seção 3), os resultados (seção 4), a discussão (seção 5) e as considerações finais (seção 6).

2. Fundamentação Teórica

2.1 O *Protégé* e sua Arquitetura de *Plugins*

Gennari *et al.* (2003) descrevem a evolução do *Protégé* desde uma ferramenta voltada à aquisição de conhecimento para sistemas especialistas até um ambiente extensível de modelagem. A partir da linha 4, o editor passou a ser construído especificamente para OWL e sobre a plataforma OSGi, que organiza a aplicação em módulos (*bundles*) carregados dinamicamente em tempo de execução (OSGI ALLIANCE, 2018). Essa arquitetura permite que funcionalidades como raciocinadores, mecanismos de consulta e visualizações sejam distribuídas como *plugins* independentes do núcleo.

Musen (2015) destaca que a comunidade de usuários e desenvolvedores é um dos principais ativos do projeto, e que grande parte das funcionalidades valorizadas pelos usuários provém de *plugins* mantidos por terceiros. Entre eles, figuram raciocinadores como o Hermit, baseado em cálculo de hipertableau (GLIMM *et al.*, 2014), e o ELK, especializado no perfil OWL 2 EL (KAZAKOV; KRÖTZSCH; SIMANČÍK, 2014), além de ferramentas de consulta, de visualização e de regras SWRL. Um panorama das ferramentas de desenvolvimento de ontologias, no qual o *Protégé* figura ao lado de outras alternativas, é apresentado por Sanches (2022), e sua variante colaborativa baseada na web, o *WebProtégé*, é discutida por Sanches (2026c).

2.2 A OWL API e a Transição para a Versão 5

A OWL API oferece uma interface de programação para criar, manipular, serializar e raciocinar sobre ontologias OWL 2, separando as interfaces de modelo das implementações de analisadores, serializadores e raciocinadores (HORRIDGE; BECHHOFFER, 2011). Por servir de base comum a editores, raciocinadores e aplicações, alterações em sua interface pública repercutem por todo o ecossistema de ferramentas que dela dependem.

Segundo o guia de migração oficial, a principal mudança da versão 5 é o suporte ao Java 8, com o surgimento de muitos métodos que retornam *streams* em lugar de coleções (OWLAPI, [2026b]). O mesmo guia relaciona outras alterações relevantes: argumentos do tipo Set substituídos por Collection e Stream; o tipo Optional da biblioteca Guava substituído por java.util.Optional; a centralização das preferências de carga e gravação no componente

OntologyConfigurator, com a remoção de classes como XMLWriterPreferences; e o tratamento de literais alinhado ao RDF 1.1. A documentação registra ainda que as versões 5.5.0 e posteriores exigem Java 11 ou superior (OWLAPI, [2026c]).

2.3 Evolução de *Software* e Mudanças Incompatíveis

As leis de evolução de *software* formuladas por Lehman (1980) estabelecem que sistemas em uso contínuo precisam ser adaptados ao ambiente para não se tornarem progressivamente menos úteis, e que essa adaptação tende a elevar sua complexidade, a menos que haja esforço deliberado de simplificação. A migração de uma dependência central, como a OWL API, é um caso típico de adaptação ao ambiente.

No ecossistema Maven, Raemaekers, Van Deursen e Visser (2017) mostraram que mudanças incompatíveis em bibliotecas são frequentes, inclusive em versões que, pelo versionamento semântico, não deveriam contê-las, e que elas impõem custos de adaptação aos sistemas clientes. No caso de plataformas extensíveis como o *Protégé*, esse custo recai sobre dois níveis: o núcleo da aplicação e cada *plugin* que dependa da interface alterada.

3 PROCEDIMENTOS METODOLÓGICOS

Esta pesquisa caracteriza-se como exploratória e descritiva quanto aos objetivos, e quali quantitativa quanto à abordagem (GIL, 2017). O delineamento adotado é o estudo de caso único (YIN, 2015), tendo como unidade de análise a versão 6.0.0-alpha-1 do *Protégé*, comparada à versão 5.6.9, última versão estável da linha 5 até a data da coleta. O recorte comparativo entre versões segue a abordagem adotada em Sanches (2026b), que confrontou a versão 5.6.9 com as versões anteriores do editor.

Foram utilizadas duas técnicas de coleta. A primeira foi a análise documental das notas de versão publicadas no GitHub, do arquivo README da linha 6, do guia de publicação (RELEASING.md) e dos arquivos de configuração Maven (pom.xml) dos repositórios *protégé* e *protege-distribution*. A segunda foi a mineração de repositório de software, com uso do sistema de controle de versões Git para comparar as etiquetas 5.6.9 e 6.0.0-alpha-1. Foram obtidos o ancestral comum entre as versões, o número de *commits* e de autores, as estatísticas de alteração por arquivo e por módulo, e a contagem de linhas adicionadas e removidas que correspondem a padrões característicos da migração para a OWL API 5, identificados por expressões regulares aplicadas ao *diff* dos arquivos Java.

Os padrões investigados foram derivados do guia oficial de migração (OWLAPI, [2026b]). A coleta foi realizada em 9 de outubro de 2026. Em atenção às limitações apontadas por Kalliamvakou *et al.* (2014) para a mineração de dados do GitHub, as contagens foram feitas diretamente sobre o histórico Git, e não sobre metadados da plataforma, e os resultados foram interpretados em conjunto com a documentação do projeto. Ressalta-se que o estudo não incluiu testes de desempenho nem avaliação com usuários da versão alfa.

4 RESULTADOS

4.1 Caracterização da Versão

A versão 6.0.0-alpha-1 foi publicada em 28 de setembro de 2026 no repositório *protege-distribution*, marcada como pré-lançamento. Sua nota de versão a descreve como a primeira versão alfa da linha *Protégé* 6, informa que ela atualiza o editor para a OWL API 5 e que se destina a testes iniciais e a trabalhos de migração. A mesma nota esclarece que apenas o raciocinador HermiT, em versão *snapshot* 1.5.0 compatível com o *Protégé* 6, foi incluído, e que os demais *plugins* normalmente empacotados foram temporariamente omitidos até a conclusão de suas migrações (PROTÉGÉ PROJECT, 2026a).

O README da linha 6 reforça esse caráter, ao declarar que o ramo está em desenvolvimento ativo, que ainda não constitui uma versão estável e que *plugins* construídos para o *Protégé* 5 podem exigir alterações para manter a compatibilidade (PROTÉGÉ PROJECT, 2026b). A configuração de montagem prevê pacotes para macOS, Linux, Windows e uma versão independente de plataforma. A licença permanece a BSD de duas cláusulas. A Figura 1 sintetiza a arquitetura em camadas do editor, destacando os pontos alterados na nova versão.

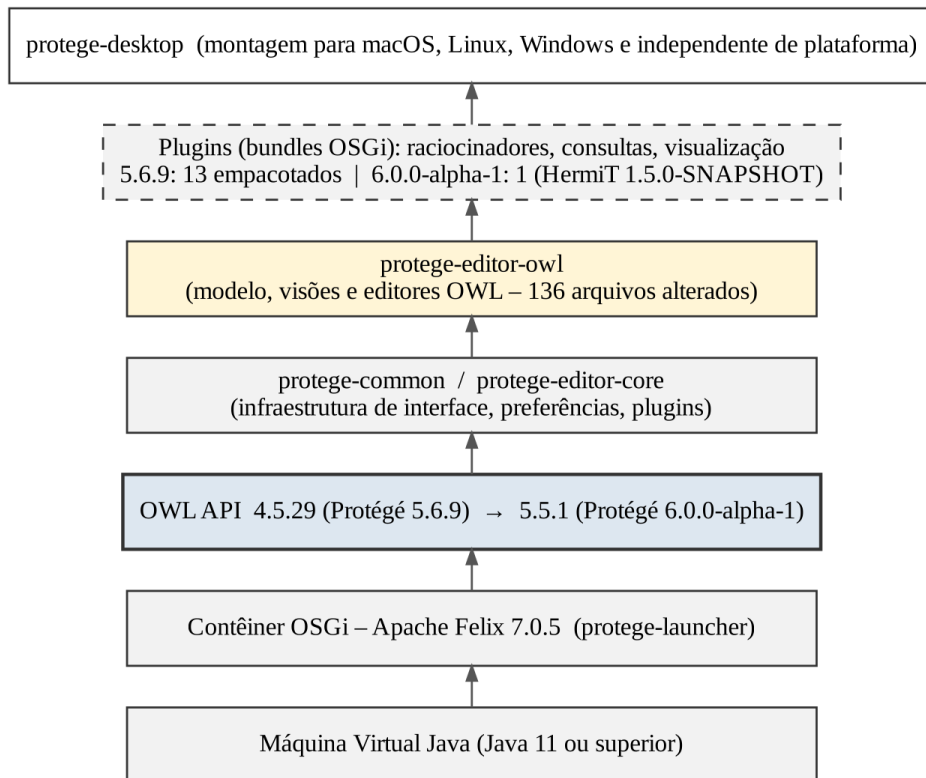


Figura 1: Arquitetura em Camadas do *Protégé* e Alteração na Versão 6.0.0-*alpha-1*.

4.2 Alterações de Dependências

A comparação dos arquivos pom.xml das duas versões, apresentada na Tabela 1, mostra que a mudança central é a substituição da OWL API 4.5.29 pela 5.5.1. O nível de compilação Java permaneceu fixado na versão 11, o que é coerente com o requisito da OWL API 5.5 (OWLAPI, [2026c]), e o contêiner OSGi Apache Felix manteve-se na versão 7.0.5. Foram retiradas do núcleo as dependências da API JAXB e do ORB CORBA do GlassFish; esta última, segundo comentário no próprio arquivo de configuração da versão 5.6.9, era exigida pelo *plugin* SWRLTab. Bibliotecas auxiliares e de teste também foram atualizadas, e a distribuição desktop passou a incluir a biblioteca Apache Commons Codec.

Tabela 1: Principais Dependências do Núcleo do *Protégé* nas versões 5.6.9 e 6.0.0-*alpha-1*.

Componente	<i>Protégé</i> 5.6.9	<i>Protégé</i> 6.0.0- <i>alpha-1</i>	Situação
OWL API (owlapi-osgidistribution)	4.5.29	5.5.1	Versão principal alterada
Nível de compilação Java	11	11	Mantido
Apache Felix (contêiner OSGi)	7.0.5	7.0.5	Mantido

Componente	<i>Protégé 5.6.9</i>	<i>Protégé 6.0.0-alpha-1</i>	Situação
API JAXB e implementações	Presentes	Ausentes	Removido
GlassFish CORBA ORB	5.0.0	Ausente	Removido
Apache Commons IO	2.14.0	2.21.0	Atualizado
Apache Commons Codec (desktop)	Ausente	1.21.0	Adicionado
Mockito / Hamcrest (testes)	2.28.2 / 2.2	3.12.4 / 3.0	Atualizado

4.3 Extensão das Modificações no Código-fonte

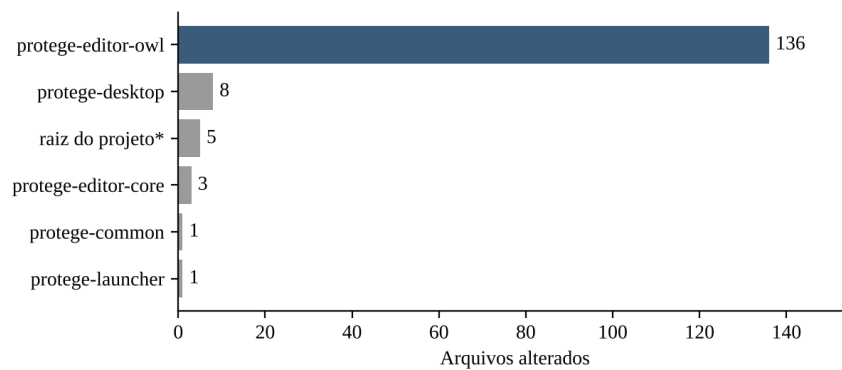
A análise do histórico revelou que o ancestral comum entre as duas versões é o próprio *commit* etiquetado como 5.6.9, ou seja, a linha 6 foi derivada diretamente da última versão estável, e não de um ramo paralelo antigo. Entre as duas etiquetas, foram registrados 33 *commits*, dos quais 10 são de integração de *pull requests*. Participaram cinco autores, mas 27 *commits* foram realizados por um único desenvolvedor. Observou-se, ainda, forte concentração temporal: 29 dos 33 *commits* ocorreram a partir de 15 de setembro de 2026, e os quatro anteriores correspondem a manutenções da linha 5.6 incorporadas ao novo ramo. A Tabela 2 resume as métricas de alteração.

Tabela 2: Métricas de Alteração entre as Etiquetas 5.6.9 e 6.0.0-alpha-1

Métrica	Todos os arquivos	Arquivos Java
Arquivos alterados	154	137
Linhas adicionadas	2.031	1.767
Linhas removidas	1.594	1.464
<i>Commits</i> (total / integrações)	33 / 10	–
Autores distintos	5	–

Fonte: Dados da Pesquisa (2026).

A distribuição das alterações por módulo, apresentada no Gráfico 1, mostra que a migração está quase inteiramente concentrada no módulo *protege-editor-owl*, responsável pelo modelo, pelas visões e pelos editores específicos de OWL e, portanto, pelo uso direto da OWL API. Os módulos de infraestrutura (*protege-common*, *protege-editor-core* e *protege-launcher*), que não dependem da semântica OWL, sofreram alterações mínimas.

Gráfico 1: Arquivos Alterados por Módulo entre as Versões 5.6.9 e 6.0.0-alpha-1.

Fonte: Dados da Pesquisa (2026). *Nota:* (*) arquivos de documentação, integração contínua e configuração Maven na raiz do repositório.

4.4 Padrões de Migração para a OWL API 5

A Tabela 3 apresenta a contagem de linhas adicionadas e removidas nos arquivos Java que correspondem aos padrões de migração descritos no guia oficial da OWL API. O padrão mais expressivo é a eliminação completa das referências ao Optional da biblioteca Guava: foram removidas 40 linhas que o referenciavam, sem nenhuma adição, enquanto 30 linhas passaram a referenciar `java.util.Optional`. O histórico confirma a intenção, com *commits* intitulados, por exemplo, como conclusão da migração para o Optional do Java no módulo `protege-editor-owl`.

Tabela 3: Ocorrências de Padrões de Migração nas Linhas Alteradas de Código Java.

Padrão (expressão buscada)	Linhas adicionadas	Linhas removidas
Optional da Guava (<code>com.google.common.base.Optional</code>)	0	40
Optional do Java (<code>java.util.Optional</code>)	30	0
Uso de <i>streams</i> (<code>Stream</code> , <code>.stream()</code> , <code>.collect()</code>)	33	4
Pesquisa de entidades (<code>EntitySearcher</code>)	15	17
Configuração de carga (<code>OntologyConfigurator</code> e similares)	14	7

Um segundo padrão relevante é a reescrita de implementações do padrão de projeto *Visitor*. Na versão 5.6.9, diversas classes estendiam adaptadores da OWL API 4, como `OWLObjectVisitorAdapter` e `OWLObjectVisitorExAdapter`. Na nova versão, essas classes passaram a implementar diretamente as interfaces de visitante, que na OWL API 5 oferecem métodos padrão, com a sobrescrita explícita de um método genérico de tratamento padrão quando necessário. O arquivo com maior volume de alterações, `OWLIconProviderImpl.java`, com 258 linhas modificadas, reflete justamente essa mudança. A Figura 2 ilustra os dois padrões com trechos extraídos do *diff* entre as versões.

```
// (a) Substituição do Optional da Guava pelo Optional do Java
- return Optional.fromNullable(subject);
+ return Optional.ofNullable(subject);
- ...getLocationFromName(name).or(FreshAxiomLocation.getDefaultValue());
+ ...getLocationFromName(name).orElse(FreshAxiomLocation.getDefaultValue());

// (b) Visitante: de adaptador da OWL API 4 para interface da OWL API 5
- public class OWLIconProviderImpl extends OWLObjectVisitorAdapter
- implements OWLIconProvider {
```

```

+ public class OWLIconProviderImpl implements OWLObjectVisitor,
+     OWLIconProvider {
- public void visit(OWLDatatype owlDatatype) {
+ @Override
+ public void visit(@Nonnull OWLDatatype owlDatatype) {

```

Figura 2: – Trechos Representativos da Migração (linhas com “-” removidas e com “+” adicionadas) (Protégé Project (2026c).

Além das adaptações, a migração foi acompanhada de remoções de funcionalidades e de acréscimo de testes. Foram excluídas a classe *XMLWriterPrefs*, coerente com a eliminação das preferências de escrita XML na OWL API 5, e toda a infraestrutura de classes definidas anônimas (*anonymous defined classes*), composta por *ADCFactory*, *ADCRewriter*, *AnonymousDefinedClassManager* e pela visão *AnonymousClassesView*. Em contrapartida, foram adicionados dez novos arquivos de teste, a maior parte voltada a verificar as implementações de visitantes migradas, a hierarquia de propriedades e o painel de criação de classes definidas. O projeto também passou a contar com um fluxo automatizado de publicação, que só publica artefatos quando a versão declarada no *pom.xml* coincide exatamente com a etiqueta Git (PROTÉGÉ PROJECT, 2026).

4.5 Situação do Ecossistema de Plugins

O Tabela 4 compara os *plugins* empacotados nas duas distribuições. Na versão 5.6.9, 13 *plugins* acompanhavam o instalador. Na versão 6.0.0-alpha-1, as declarações desses *plugins* foram mantidas no arquivo de configuração, porém comentadas, acompanhadas da observação de que permanecem desativados até a conclusão de suas migrações para o Protégé 6; apenas o HerMiT foi incluído, em uma versão *snapshot* com novo identificador de artefato (PROTÉGÉ PROJECT, 2026d).

Tabela 4: *Plugins* Empacotados nas Distribuições 5.6.9 e 6.0.0-alpha-1.

<i>Plugin</i>	Finalidade	Protégé 5.6.9	Protégé 6.0.0-alpha-1
HerMiT	Raciocinador OWL 2 DL	1.4.3.456	1.5.0-SNAPSHOT
ELK	Raciocinador OWL 2 EL	0.6.0	Omitido
DL Query	Consulta em lógica de descrição	4.0.1	Omitido
SPARQL Query	Consulta SPARQL	3.0.0	Omitido
Existential Query	Consulta por restrições existenciais	2.0.0	Omitido
Explanation Workbench	Explicação de inferências	3.0.1	Omitido
SWRLTab	Edição e execução de regras SWRL	2.0.10	Omitido
OWLViz	Visualização de hierarquias	5.0.3	Omitido
OntoGraf	Visualização de grafos	2.0.3	Omitido
Cellfie	Importação de planilhas	2.1.0	Omitido

<i>Plugin</i>	Finalidade	<i>Protégé</i> 5.6.9	<i>Protégé</i> 6.0.0-alpha-1
OWLDoc	Geração de documentação HTML	3.0.3	Omitido
Code Generation	Geração de código Java	2.0.0	Omitido
RDF Library	Biblioteca de ontologias RDF	3.0.1	Omitido

5 DISCUSSÃO

Os resultados indicam que a versão 6.0.0-alpha-1 é, essencialmente, uma versão de infraestrutura. Ela não acrescenta funcionalidades visíveis ao usuário final e, ao contrário, retira algumas. O que muda é a base sobre a qual o editor e seus *plugins* são construídos. Sob a perspectiva de Lehman (1980), trata-se de uma adaptação ao ambiente que o projeto vinha adiando: manter a OWL API 4 preservava a compatibilidade com os *plugins* existentes, mas mantinha o *Protégé* preso a uma linha de biblioteca que a própria comunidade da OWL API tratava como sustentada apenas enquanto houvesse dependentes (OWLAPI, [2026a]).

A estratégia adotada apresenta escolhas de engenharia que reduzem o risco da transição. A primeira é o isolamento da mudança: manteve-se a versão do Java e do contêiner OSGi, de modo que a migração da OWL API não se somasse a outras alterações estruturais. A segunda é a derivação direta da versão 5.6.9, que preserva as correções acumuladas na linha estável. A terceira é a combinação entre remoção de código obsoleto, novos testes automatizados e publicação controlada por etiqueta, práticas que favorecem a rastreabilidade. A concentração das alterações no módulo *protege-editor-owl* confirma, ainda, que a separação em camadas da arquitetura funcionou como esperado, limitando o impacto da troca da biblioteca aos componentes que de fato manipulam ontologias.

O principal custo da mudança, contudo, recai sobre o ecossistema. Como observam Raemaekers, Van Deursen e Visser (2017), mudanças incompatíveis em bibliotecas transferem esforço de adaptação para os sistemas clientes. No *Protégé*, os clientes são os *plugins*, muitos mantidos por terceiros e com ritmos de manutenção distintos. A omissão de 12 dos 13 *plugins* na distribuição alfa torna esse custo explícito. Para o usuário, significa que recursos de uso cotidiano, como as abas DL Query e SPARQL Query, as visualizações e o editor de regras SWRL, não estão disponíveis nesta versão. Para o desenvolvedor de *plugins*, significa a necessidade de revisar ao menos o uso de Optional, de coleções e de visitantes, padrões que, como mostra a Tabela 2, foram os mais afetados no próprio núcleo.

A remoção da infraestrutura de classes definidas anônimas merece atenção específica. Por não constar no guia de migração da OWL API, trata-se de uma decisão do próprio projeto *Protégé*, aproveitando a mudança de versão principal para eliminar código. Qualquer *plugin* que dependesse dessas classes precisará ser reescrito, e não apenas adaptado.

No ensino, os resultados recomendam cautela. Em disciplinas de engenharia de ontologias, nas quais o raciocínio automatizado e as consultas *DL Query* costumam ser usados para verificar a modelagem dos estudantes, a versão 5.6.9 continua sendo a escolha adequada. O mesmo vale para projetos aplicados que dependem desses recursos, como a construção de ontologias de domínio na área da saúde, a exemplo da ontologia para o Transtorno do Espectro Autista desenvolvida no *Protégé* 5.6.7 por Sanches e Gomes

(2025a). A versão alfa é, por outro lado, um objeto de estudo pertinente em disciplinas de engenharia e evolução de software, por oferecer um caso real e recente de migração de dependência central em uma plataforma extensível.

6. Conclusão

Este artigo caracterizou a versão 6.0.0-alpha-1 do *Protégé* e analisou as implicações de sua migração para a OWL API 5. Em relação aos objetivos específicos, verificou-se que a principal alteração de dependência é a passagem da OWL API 4.5.29 para a 5.5.1, com manutenção do *Java* 11 e do **Apache Felix** 7.0.5 e remoção das dependências *JAXB* e *CORBA*; que a migração envolveu 33 *commits* e 154 arquivos, concentrados no módulo *protege-editor-owl*, com destaque para a substituição do *Optional* da *Guava*, a adoção de *streams* e a reescrita de visitantes; e que a distribuição alfa empacota apenas o *HermiT*, em contraste com os 13 *plugins* da versão 5.6.9.

Em resposta à questão de pesquisa, conclui-se que a versão 6.0.0-alpha-1 representa uma modernização arquitetural necessária e conduzida de forma controlada, mas que ainda não é adequada para uso em produção ou em sala de aula como ferramenta principal. O êxito da linha 6 dependerá menos do núcleo, cuja migração já foi realizada, e mais da velocidade com que os mantenedores de *plugins* adaptarem seus projetos.

Como trabalhos futuros, propõe-se: (a) acompanhar as próximas versões da linha 6, medindo a recomposição do conjunto de *plugins* ao longo do tempo; (b) avaliar empiricamente o desempenho de carga e de raciocínio das duas versões sobre ontologias de referência; e (c) investigar, junto a desenvolvedores de *plugins*, o esforço efetivo de migração para a OWL API 5.

Referências

- GENNARI, J. H. *et al.* The evolution of Protégé: an environment for knowledge-based systems development. **International Journal of Human-Computer Studies**, [s. l.], v. 58, n. 1, p. 89-123, 2003.
- GIL, A. C. **Como elaborar projetos de pesquisa**. 6. ed. São Paulo: Atlas, 2017.
- GLIMM, B. *et al.* HermiT: an OWL 2 reasoner. **Journal of Automated Reasoning**, [s. l.], v. 53, n. 3, p. 245-269, 2014.
- GRUBER, T. R. A translation approach to portable ontology specifications. **Knowledge Acquisition**, [s. l.], v. 5, n. 2, p. 199-220, 1993.
- HITZLER, P. *et al.* **OWL 2 Web Ontology Language primer**. 2. ed. [S. l.]: W3C, 2012. W3C Recommendation. Disponível em: <https://www.w3.org/TR/owl2-primer/>. Acesso em: 9 out. 2026.
- HORRIDGE, M.; BECHHOFFER, S. The OWL API: a Java API for OWL ontologies. **Semantic Web**, [s. l.], v. 2, n. 1, p. 11-21, 2011.
- KALLIAMVAKOU, E. *et al.* The promises and perils of mining GitHub. In: WORKING CONFERENCE ON MINING SOFTWARE REPOSITORIES, 11., 2014, Hyderabad. **Proceedings** [...]. New York: ACM, 2014. p. 92-101.
- KAZAKOV, Y.; KRÖTZSCH, M.; SIMANČÍK, F. The incredible ELK: from polynomial procedures to efficient reasoning with EL ontologies. **Journal of Automated Reasoning**, [s. l.], v. 53, n. 1, p. 1-61, 2014.
- LEHMAN, M. M. Programs, life cycles, and laws of software evolution. **Proceedings of the IEEE**, [s. l.], v. 68, n. 9, p. 1060-1076, 1980.
- MUSEN, M. A. The Protégé project: a look back and a look forward. **AI Matters**, New York, v. 1, n. 4, p. 4-12, 2015.

- NOY, N. F.; MCGUINNESS, D. L. **Ontology development 101**: a guide to creating your first ontology. Stanford: Stanford Knowledge Systems Laboratory, 2001. (Technical Report KSL-01-05).
- OSGI ALLIANCE. **OSGi core**: release 7. San Ramon: OSGi Alliance, 2018.
- OWLAPI. **The OWL API**: the main repository. [S. l.], [2026a]. Disponível em: <https://owlcs.github.io/owlapi/>. Acesso em: 9 out. 2026.
- OWLAPI. **Migrate to version 5**. [S. l.], [2026b]. GitHub Wiki. Disponível em: <https://github.com/owlcs/owlapi/wiki/Migrate-to-version-5>. Acesso em: 9 out. 2026.
- OWLAPI. **owlapi**: repositório de código-fonte, ramo version5. [S. l.], [2026c]. Disponível em: <https://github.com/owlcs/owlapi>. Acesso em: 9 out. 2026.
- PRESTON-WERNER, T. **Semantic Versioning 2.0.0**. [S. l.], [2013]. Disponível em: <https://semver.org/>. Acesso em: 9 out. 2026.
- PROTÉGÉ PROJECT. **Protégé 6.0.0-alpha-1**. [S. l.]: GitHub, 28 set. 2026a. Nota de versão. Disponível em: <https://github.com/protegeproject/protege-distribution/releases/tag/protege-6.0.0-alpha-1>. Acesso em: 9 out. 2026.
- PROTÉGÉ PROJECT. **README**: Protégé 6 development. [S. l.]: GitHub, 2026b. Disponível em: <https://github.com/protegeproject/protege/tree/protege-6>. Acesso em: 9 out. 2026.
- PROTÉGÉ PROJECT. **protege**: repositório de código-fonte, etiquetas 5.6.9 e 6.0.0-alpha-1. [S. l.]: GitHub, 2026c. Disponível em: <https://github.com/protegeproject/protege>. Acesso em: 9 out. 2026.
- PROTÉGÉ PROJECT. **protege-distribution**: pom.xml, etiqueta protege-6.0.0-alpha-1. [S. l.]: GitHub, 2026d. Disponível em: <https://github.com/protegeproject/protege-distribution>. Acesso em: 9 out. 2026.
- PROTÉGÉ PROJECT. **Releasing Protégé 6**. [S. l.]: GitHub, 2026e. Disponível em: <https://github.com/protegeproject/protege/blob/protege-6/RELEASING.md>. Acesso em: 9 out. 2026.
- PROTÉGÉ PROJECT. **protege-distribution**: pom.xml, etiqueta protege-5.6.9. [S. l.]: GitHub, 2026f. Disponível em: <https://github.com/protegeproject/protege-distribution>. Acesso em: 9 out. 2026.
- RAEMAEKERS, S.; VAN DEURSEN, A.; VISSER, J. Semantic versioning and impact of breaking changes in the Maven repository. **Journal of Systems and Software**, [S. l.], v. 129, p. 140-158, 2017.
- SANCHES, H. M. Ferramentas de desenvolvimento e criação de ontologias. In: **Ciências exatas e da terra e engenharias**: conhecimento e informação. Ponta Grossa: Atena, 2022. p. 129-134. DOI: <https://doi.org/10.22533/at.ed.53622061211>.
- SANCHES, H. M. Protégé 5.6.3: ontology management with efficiency and accuracy. **Journal of Engineering Research**, Ponta Grossa, v. 4, n. 19, p. 2-5, 2024. DOI: <https://doi.org/10.22533/at.ed.3174192401078>.
- SANCHES, H. M. Protégé 5.6.9: avanços e aplicações no desenvolvimento de ontologias. **Revista Processus Multidisciplinar**, Brasília, 2026a. Disponível em: <https://periodicos.processus.com.br/index.php/multi/article/view/1628>. Acesso em: 9 out. 2026.
- SANCHES, H. M. Protégé 5.6.9: novidades, processos e comparativo com versões anteriores. In: **Ciência, tecnologia e inovação**: horizontes e transformações científicas. Ponta Grossa: Atena, 2026b. p. 84-95. DOI: <https://doi.org/10.22533/at.ed.939112621016>.
- SANCHES, H. M. WebProtégé: uma plataforma colaborativa baseada na web para desenvolvimento de ontologias. In: **Ciência, tecnologia e inovação**: horizontes e transformações científicas. Ponta Grossa: Atena, 2026c. p. 158-170. DOI: <https://doi.org/10.22533/at.ed.939112621019>.

- SANCHES, H. M.; DUTRA, P. P. Protégé 5.5.0 ferramenta open source para criação de ontologia. *In: Engenharia biomédica: desenvolvimento e inovação*. Ponta Grossa: Atena, 2022. p. 11-20. DOI: <https://doi.org/10.22533/at.ed.2252209112>.
- SANCHES, H. M.; GOMES, M. M. F. Protégé 5.6.4 ferramenta para criação de ontologia em Web Ontology Language (OWL). **Revista ft**, Rio de Janeiro, v. 29, n. 141, p. 29-30, 2024. DOI: <https://doi.org/10.69849/revistaft/ar10202412240029>.
- SANCHES, H. M.; GOMES, M. M. F. Construção de uma ontologia para Transtorno do Espectro Autista (TEA) utilizando Protégé 5.6.7 e OWL. **Revista ft**, Rio de Janeiro, v. 29, n. 153, p. 45-46, 2025a. DOI: <https://doi.org/10.69849/revistaft/ch10202512240645>.
- SANCHES, H. M.; GOMES, M. M. F. Protégé 5.6.5 tool for creating ontologies in Web Ontology Language (OWL). **Journal of Engineering Research**, Ponta Grossa, v. 5, n. 4, p. 1-7, 2025b. DOI: <https://doi.org/10.22533/at.ed.317542517038>.
- STUDER, R.; BENJAMINS, V. R.; FENSEL, D. Knowledge engineering: principles and methods. **Data & Knowledge Engineering**, [s. l.], v. 25, n. 1-2, p. 161-197, 1998.
- YIN, R. K. **Estudo de caso: planejamento e métodos**. 5. ed. Porto Alegre: Bookman, 2015.