UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

FERRAMENTA WEB DE SUPORTE A AVALIAÇÃO DE SOFTWARE COM A METODOLOGIA CERTICS

VINÍCIUS FERNEDA DE LIMA

BLUMENAU
2014

2014/1-25

VINÍCIUS FERNEDA DE LIMA

FERRAMENTA WEB DE SUPORTE A AVALIAÇÃO DE SOFTWARE COM A METODOLOGIA CERTICS

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciência da Computação — Bacharelado.

Prof. Everaldo Artur Grahl, Mestre - Orientador

BLUMENAU
2014

2014/1-25

FERRAMENTA WEB DE SUPORTE A AVALIAÇÃO DE SOFTWARE COM A METODOLOGIA CERTICS

Por

VINÍCIUS FERNEDA DE LIMA

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

Presidente:	Prof. Everaldo Artur Grahl, Mestre – Orientador, FURB
Membro:	Prof. Marcel Hugo, Mestre – FURB
Membro:	Prof. Jacques Robert Heckmann, Mestre – FURB

Blumenau, 11 de julho de 2014.

Dedico este trabalho a toda a minha família, especialmente aos meus pais, à minha namorada, aos meus amigos e aos meus professores de todo o curso.

AGRADECIMENTOS

A Deus, pelo seu imenso amor, graça e por ter me iluminado durante todo esse processo.

À minha família que sempre me motivaram a estudar, especialmente meus pais, Rosiley e Décio, que incessantemente se esforçaram para oferecer a mim e a meu irmão a melhor educação e quando mais precisei estavam ao meu lado.

À minha namorada, Natana, por ter sido compreensiva durante este trabalho, por me apoiar, amar e sempre estar ao meu lado nas alegrias e também nas dificuldades.

Aos meus amigos que me ajudaram e me apoiaram nos momentos em que necessitei de auxílio no decorrer do curso.

Ao meu orientador, Everaldo Artur Grahl, por dedicar parte de seu tempo a me auxiliar e me orientar durante todo o desenvolvimento deste trabalho.

O importante não é justificar o erro, mas impedir que ele se repita.

Ernesto Che Guevara

RESUMO

Este trabalho apresenta o desenvolvimento de uma ferramenta web que possibilita a avaliação de softwares utilizando como base a metodologia da Certificação de Software Resultante de Desenvolvimento e Inovação Tecnológica para Software (CERTICS). Foi utilizada a linguagem Java, o *framework Demoiselle*, que oferece um ambiente de desenvolvimento, e diversos componentes que auxiliam na construção da ferramenta utilizando a arquitetura *Model View Controller* (MVC), o *framework* Primefaces, que apresenta os componentes visuais com suporte a *Java Server Faces* (JSF), e a biblioteca JasperReports, que disponibiliza suporte à geração de relatórios. Os resultados demonstraram que a ferramenta web apoia os consultores e as organizações solicitantes das avaliações, facilitando o registro das evidências. Além disso, pode-se utilizar a ferramenta para dinamizar o ensino de qualidade de software.

Palavras-chave: Metodologia CERTICS. Avaliação de software. Qualidade de software.

ABSTRACT

This paper presents a web tool development which allows the software assessment based on the Software Certification Resultant Development and Technological Innovation for Software (CERTICS) methodology. For this, propose the Java language was used, the Demoiselle framework, that provides a development environment, and various components that assist in the of the tool construction using the Model View Controller (MVC) architecture, Primefaces framework, that presents visual components to support Java Server Faces (JSF), and the JasperReports library, that provides support for report generation. The results showed that the web tool supports consultants and the evaluations applicant organizations, facilitating the evidence recording. Moreover, may be used to dynamize software quality teaching.

Key-words: CERTICS methodology. Software assessment. Software quality.

LISTA DE ILUSTRAÇÕES

LISTA DE SIGLAS

ABNT – Associação Brasileira de Normas Técnicas

API – *Application Programming Interface*

BP – *Base Practice*

CenPRA – Centro de Pesquisas Renato Archer

CERTICS – Certificação de Software Resultante de Desenvolvimento e Inovação Tecnológica para Software

CRUD – *Creat, Read, Update and Delete*

CSS – *Cascading Style Sheets*

CTI - Centro de Tecnologia da Informação

HTML – *HyperText Markup Language*

ISO – *International Organization for Standardization*

IEC – *International Electrotechnical Commission*

JSF – *Java Server Faces*

PAM – *Process Assessment Model*

PRM – *Process Reference Model*

MCTI – Ministério da Ciência, Tecnologia e Inovação

MER – Modelo Entidade Relacionamento

MPS.BR – Melhoria de Processos do Software Brasileiro

MVC – *Model View Controller*

SEPIN – Secretaria de Política de Informática

SGBD – Sistema Gerenciador de Bancos de Dados

SGBDOR – Sistema Gerenciador de Banco de Dados Objeto Relacional

SPICE – *Software Process Improvement and Capability Determination*

UML – *Unified Modeling Language*

WP – *Working Products*

XML – *eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

A metodologia de avaliação CERTICS surgiu da necessidade de verificar se um software é resultado de desenvolvimento e inovação tecnológica realizada no Brasil. Com a aplicação desta metodologia, pretende-se dar preferência às instituições que participam de compras públicas e contribuem para o desenvolvimento nacional sustentável (CTI RENATO ARCHER, 2013a).

A metodologia CERTICS foi criada em 2013, sendo mantida pelo governo federal no sentido de privilegiar empresas nacionais e que tenham produtos inovadores e de qualidade. Esta metodologia é considerada inclusiva, ágil e acessível, focada em analisar um conjunto de evidências relacionadas às quatro áreas de competências fundamentais: desenvolvimento tecnológico, gestão de tecnologia, gestão de negócios e melhoria contínua (CTI RENATO ARCHER, 2013a).

A metodologia define que a avaliação é do software, não é da empresa, e é baseada na análise dos processos utilizados no software ou na construção do software. Ela propõe um novo conceito para que o software seja o resultado de desenvolvimento e inovação tecnológica realizados no país, com isso é exigido um novo modelo de referência e um novo método de avaliação de um software. É apresentado um conjunto mínimo de resultados esperados para a caracterização de desenvolvimento e inovação tecnológica realizados no país e exige a demonstração da obtenção destes resultados (CTI RENATO ARCHER, 2013a).

A implantação desta certificação no país visa uma maior contribuição para o desenvolvimento nacional de software e um maior investimento em inovação tecnológica nacional, com isso estimulando a produção caseira de bens e serviços, consolidando o poder e induzindo o desenvolvimento de compras governamentais nas redes produtivas nacionais. O desenvolvimento desta metodologia atende a uma demanda da Secretaria de Política de Informática (SEPIN), do Ministério da Ciência, Tecnologia e Inovação (MCTI), dirigida ao Centro de Tecnologia da Informação Renato Archer (CTI) para que seja possível certificar os softwares das organizações solicitantes de todo o Brasil (CTI RENATO ARCHER, 2013a).

Por ser recente, ainda são restritas soluções para adoção desta metodologia. Existe inicialmente a aplicação oficial usada pelo próprio Centro de Tecnologia da Informação Renato Archer na condução de toda logística usada no processo de certificação. Diante deste cenário, neste trabalho foi desenvolvida uma ferramenta web que permita apoiar o processo de avaliação de um software utilizando a metodologia CERTICS. Este software visa apoiar a avaliação de um consultor na organização solicitante, promovendo uma maior facilidade para

o avaliador identificar as evidências do software da organização solicitante que estará sendo avaliado. Além disso, a ferramenta permite aos professores da área de qualidade de software uma opção para aulas práticas.

1.1 OBJETIVOS DO TRABALHO

O objetivo geral deste trabalho é desenvolver uma ferramenta web que possa apoiar a avaliação de software usando a metodologia CERTICS.

Os objetivos específicos podem ser detalhados da seguinte forma:

a) disponibilizar um ambiente que permita configurar a metodologia de avaliação da CERTICS;
b) agilizar o cálculo dos resultados de uma avaliação e possibilitar a emissão de relatórios detalhados com a consequente resultância das avaliações do software;
c) permitir que os avaliadores consigam avaliar as evidências registradas pela empresa que está sendo analisada.

1.2 ESTRUTURA DO TRABALHO

O trabalho está estruturado em quatro capítulos. O próximo capítulo contém a fundamentação teórica necessária para proporcionar um melhor entendimento sobre o trabalho desenvolvido.

O capítulo três apresenta o desenvolvimento da ferramenta, disponibilizando os principais requisitos do problema e a especificação, com os casos de uso, os diagramas de classes, o modelo de entidade relacionamento e o diagrama de atividades. Neste mesmo capítulo são apresentadas as ferramentas utilizadas na especificação e na implementação, além da operacionalidade da ferramenta desenvolvida e os resultados obtidos.

O quarto capítulo refere-se às conclusões do presente trabalho e às sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

A seção 2.1 descreve a metodologia CERTICS, definindo os seus conceitos e características de avaliação. A seção 2.2 menciona a Norma ABNT NBR ISO/IEC 15504-2 (2008) – "tecnologia da informação - avaliação de processo - parte 2: realização de uma avaliação". Por fim, na seção 2.3 estão expostos os trabalhos correlatos.

2.1 METODOLOGIA CERTICS

A metodologia CERTICS tem como principal objetivo avaliar se um software foi resultado de um determinado desenvolvimento e inovação tecnológica no país, tendo como base as evidências relacionadas ao software. O modelo de referência para avaliação da CERTICS está estruturado em quatro camadas conceituais hierárquicas. A primeira camada trata do conceito de um software ser resultado do desenvolvimento e inovação tecnológica realizados no país e de sua correlação com o desenvolvimento nacional. A segunda é composta por quatro áreas de competência que detalham os conceitos especificados na primeira. As áreas são dividas em Desenvolvimento Tecnológico (DES), Gestão de Tecnologia (TEC), Gestão de Negócios (GNE) e Melhoria Contínua (MEC). A terceira camada é composta por resultados esperados, que detalham cada uma das áreas de competência. Por fim, a quarta e última camada é composta por conjuntos de orientações e indicadores, que detalham a terceira camada (CTI RENATO ARCHER, 2013a).

A metodologia de avaliação da CERTICS para software e o seu desenvolvimento atendem às seguintes diretrizes (CTI RENATO ARCHER, 2013b):

a) a avaliação é do software, não da empresa, e é baseada na análise dos processos utilizados no software;
b) a metodologia é baseada na Norma ABNT NBR ISO/IEC 15504-2 (2008) para avaliação de processo e na experiência do CTI Renato Archer e de seus parceiros;
c) um novo conceito (software ser resultado de desenvolvimento e inovação tecnológica realizados no país) demanda um novo modelo de referência e um novo método para avaliação;
d) a metodologia apresenta um conjunto mínimo de resultados esperados para a caracterização de desenvolvimento e inovação tecnológica realizados no país e exige a demonstração da obtenção destes resultados;
e) nenhuma forma específica de estruturação, operação e documentação são exigidas da organização solicitante.

CTI Renato Archer (2013b) define que o processo de avaliação é composto por seis fases sequenciais, que podem ser definidas como fase 1 (exploração) – permite que a organização solicitante entenda melhor a relação do software avaliado e a metodologia; fase 2 (contratação) – é estabelecido o contrato da avaliação; fase 3 (preparação) – preparação da organização solicitante e da equipe de avaliação para a visita de avaliação; fase 4 (visita) – é realizada a visita da equipe de avaliação para realizar a análise das evidências e pontuar os resultados esperados; fase 5 (validação) – assegura que a avaliação foi realizada seguindo as conformidades da metodologia de avaliação da CERTICS e fase 6 (conclusão) – é a conclusão do processo, são emitidos os relatórios com os resultados da avaliação e caso sejam positivos a organização solicitante pode requerer a certificação CERTICS à SEPIN do MCTI.

Uma avaliação gera como resultado principal um relatório, denominado relatório da avaliação, com informações sobre o contexto da avaliação, a pontuação atribuída aos resultados esperados do modelo de referência, a justificativa para cada pontuação e o resultado da avaliação. Para garantir que os resultados da avaliação sejam objetivos, imparciais, consistentes, repetíveis e representativos, a metodologia de avaliação da CERTICS para software segue os requisitos estabelecidos na Norma ABNT NBR ISO/IEC 15504-2 (2008), inclusive quanto à sua pontuação.

CTI Renato Archer (2013b) define que a metodologia de avaliação é composta por quatro áreas de competência, que são definidas da seguinte forma:

a) desenvolvimento tecnológico: refere-se ao domínio do conhecimento nas tecnologias relevantes presentes no software, para que seja possível o seu desenvolvimento tecnológico, manutenção, suporte e evolução. Esta competência está dividida em seis resultados esperados. Utiliza-se a sigla DES;
b) gestão de tecnologia: envolve o estabelecimento de ações direcionadoras para a pesquisa e desenvolvimento de novas tecnologias, a absorção de tecnologias e/ou aquisição de tecnologias existentes a serem incorporados no software, levando em consideração a autonomia e inovação tecnológicas como fatores relevantes. Está competência está divida em quatro resultados esperados. Utiliza-se a sigla TEC;
c) gestão de negócios: refere-se à administração de ações voltadas para a promoção e o aumento de negócios baseados em conhecimento, a partir do software. Esta competência está dividida em três resultados esperados. Utiliza-se a sigla GNE;
d) melhoria contínua: abrange um conjunto de atividades, coerentes entre si, que apóiam e potencializam de forma integrada as outras áreas de competência do modelo de referência, objetivando a melhoria contínua do software. Esta

competência está dividida em três resultados esperados. Utiliza-se a sigla MEC.

Cada área de competência envolve, com ênfases diferentes, tanto aspectos de competências tecnológicas quanto de competências correlatas. Cada uma das quatros áreas de competência são compostas por resultados esperados, que tem por objetivo detalhar cada uma, e possuem uma pergunta-chave para que se tenha uma melhor visão da competência que deve ser atingida (CTI RENATO ARCHER, 2013b).

O resultado de uma avaliação nesta metodologia é definido como um valor binário, sim ou não, obtido a partir da pontuação de cada resultado esperado. A seguir segue a escala de pontuação ordinal definida pelo CTI Renato Archer (2013c) que tem por objetivo expressar o alcance de cada resultado esperado em uma avaliação:

a) completamente atendido (*fully* – F): possui evidências suficientes relacionadas ao software que possam identificar o total atendimento do resultado esperado;
b) largamente atendido (*largely* – L): possui evidências suficientes para demonstrar o atendimento dos aspectos mais importantes do resultado esperado, porém existem pontos fracos relacionados ao resultado esperado, mas que não comprometem o atendimento;
c) parcialmente atendido (*partially* – P): possui evidências que possam demonstrar o atendimento do resultado esperado, porém existem pontos fracos que comprometem o atendimento;
d) não atendido (*not* – N): todas as evidências apresentadas são insuficientes ou não foram apresentadas de forma adequada para que se possa demonstrar o atendimento do resultado esperado.

Juntamente com a atribuição da pontuação F, L, P e N são acompanhadas de uma justificativa do avaliador para que fiquem evidenciados quais foram os critérios utilizados para a avaliação. Quando a avaliação for diferente de F deve ser atribuído juntamente da justificativa os pontos fracos encontrados.

CTI Renato Archer (2013c) define que a realização da avaliação produz e necessita de evidências para a realização da avaliação do software. As principais evidências são:

a) base de dados de avaliação: representa o conjunto de dados necessários para que se realize a avaliação do software da organização solicitante. Estes dados são compostos por evidências que possam ajudar ao apoio da definição do grau de atendimento e justificativas para cada resultado esperado;
b) *checklist* da validação: formulário que contém itens a serem validados pelo avaliador em uma avaliação;

c) estimativa de sucesso: informação apresentada no final da fase de exploração que indica o sucesso da avaliação do software e de conseguir a certificação;
d) estimativa do grau de prontidão: esta informação é apresentada no final da fase de preparação e indica o quanto a organização solicitante está preparada para a realização da fase de visita;
e) formulário de validação da avaliação: relaciona as validações realizadas e um parecer sobre a validação com a justificativa do parecer;
f) plano de avaliação: plano acordado entre a organização solicitante e a equipe de avaliação sobre como irá proceder a avaliação do software;
g) relatório de resultado da avaliação: contém o resultado da avaliação e as pontuações, justificativas e pontos fracos de cada resultado esperado dividido pelas áreas de competência.

2.2 NORMA ABNT NBR ISO/IEC 15504-2

A Norma NBR ISO/IEC 15504-2:2003 (ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS, 2008) trata da avaliação de processo e de sua aplicação para a melhoria e determinação da capacidade. Ela define o conjunto mínimo de requisitos para a realização de uma avaliação. Os resultados de avaliações de processos podem ser comparados quando os escopos das avaliações são considerados semelhantes.

Segundo Koscianski e Soares (2006), o ínicio da norma 15504 remonta ao ano de 1991, quando o JTC1 iniciou um estudo sobre a necessidade de uma norma para a avaliação de processos de software. Em 1993 teve início o projeto SPICE (*Software Process Improvement and Capability Determination*), com os seguintes objetivos:

a) auxiliar o início do projeto com a finalidade de executar testes de campo;
b) obter dados de experiências práticas;
c) despertar o mercado para o surgimento da futura norma.

A envergadura do projeto SPICE acabou associando definitivamente o acrônimo à norma ISO/IEC 15504. Esse projeto foi oficialmente encerrado em março de 2003, sendo substituído pelo SPICE *Network* (KOSCIANSKI; SOARES, 2006).

A norma 15504 define um conceito chamado modelo de referência de processo (PRM – *Process Reference Model*). Um PRM contém uma descrição de escopo e uma descrição de requisitos. Tais requisitos estabelecem os resultados esperados da execução de cada processo. Permitem avaliar se os objetivos do processo serão alcançados (KOSCIANSKI; SOARES, 2006).

Segundo Koscianski e Soares (2006), para realizar uma medição, define-se um modelo de medição (PAM – *Process Assessment Model*), que identifica elementos da organização a serem exeminados. Para cada processo ele define dois indicadores: práticas base (BP – *Base Practice*) e artefatos produzidos (WP - *Working Products*). Tais elementos são utilizados na chamada dimensão de processo e permitem verificar se os processos são ou não executados (*assessment of process performance*). Outra dimensão de avaliação é a dimensão de capacidade (*assessment of process capability*). O PAM define seis níveis de capacidade para os processos, conforme apresenta o Quadro 1.

Quadro 1 – Níveis de capacitação da ISO/IEC 15504

NÍVEL	NOME	DESCRIÇÃO
0	Incompleto	O processo não é implementado ou falha em atingir seus objetivos
1	Executado	O processo essencialmente atinge os objetivos, mesmo se de forma pouco planejada ou rigorosa
2	Gerenciado	O processo é implementado de forma controlada (planejado, monitorado e ajustado); os produtos por ele criados são controlados e mantidos de forma apropriada
3	Estabelecido	O processo é implementado de forma sistemática e consistente
4	Previsível	O processo é executado e existe um controle que permite verificar se ele se encontra dentro dos limites estabelecidos para atingir os resultados
5	Otimizado	O processo é adaptado continuamente para, de uma forma mais eficiente, atingir os objetivos de negócio definidos e projetados

Fonte: Koscianski e Soares (2006, p. 159).

Koscianski e Soares (2006) definem que a dimensão de capacidade permite uma avaliação mais detalhada dos processos executados por uma organização, enquanto a dimensão de processo se limita à verificação de execução ou não dos processos. Para avaliar sob essa ótica, o PAM inclui a definição de atributos de processo, que permitem avaliá-los em uma escala. Há um total de nove atributos agrupados em níveis de capacidade, que são aplicáveis a todos os processos. Os atributos são apresentados no Quadro 2.

Quadro 2 – Atributos de processos

NÍVEL	ATRIBUTO
1	1.1 Execução
2	2.1 Administração do processo 2.2 Administração dos produtos obtidos do processo
3	3.1 Definição 3.2 Implementação
4	4.1 Medição 4.2 Controle
5	5.1 Inovação 5.2 Otimização

Fonte: Koscianski e Soares (2006, p. 161).

Os atributos de um processo podem ser avaliados em quatro níveis qualitativos. Cada nível é mapeado a uma faixa de valores porcentuais variando de 0%, quando o atributo não é

alcançado, a 100%, quando é complementamente satisfeito. O Quadro 3 apresenta os níveis e as faixas porcentuais correspondentes (KOSCIANSKI; SOARES, 2006).

Quadro 3 – Escala de capacidade dos atributos de processos

PORCENTAGEM	NÍVEL	DESCRIÇÃO
0% a 15%	Não atendido (*not* – N)	Existe pouca ou nenhuma evidência de que o atributo do processo seja alcançado
16% a 50%	Parcialmente atendido (*partially* – P)	Existe evidência de uma abordagem sistemática significativa para atingir o atributo. Alguns aspectos, como resultados, podem ser imprevisíveis
51% a 85%	Largamente atendido (*largely* – L)	O desempenho do processo pode variar em algumas áreas
86% a 100%	Completamente atendido (*fully* – F)	Não há nenhuma falta ou falha significativa

Fonte: adaptado de Koscianski e Soares (2006, p. 163).

CTI Renato Archer (2013b) define que a norma tem por objetivo definir os requisitos mínimos para a realização de uma avaliação que garanta coerência e boa granularidade para o processo. Esta norma define seis níveis de capacidade, porém o método de avaliação da CERTICS avalia somente o nível 1 de capacidade (processo executado).

A metodologia da CERTICS utilizou como base a Norma ISO/IEC 15504-2 (2008) para garantir que os resultados da avaliação sejam objetivos, imparciais, consistentes, repetíveis e representativos. A metodologia de avaliação da CERTICS para software segue os requisitos estabelecidos pela norma inclusive quanto à sua pontuação (CTI RENATO ARCHER, 2013b).

2.3 TRABALHOS CORRELATOS

Existem disponíveis ferramentas de avaliações de softwares utilizando outras metodologias de avaliação, mais voltadas à avaliação de processos como Melhoria de Processos do Software Brasileiro (MPS.BR). É possível encontrar diversos trabalhos de conclusão de curso relacionados a avaliação de software, como “WISE: Uma Abordagem de Ferramenta de Software Para o Auxílio no Modelo de Avaliação do MPS.BR” de Albuquerque Júnior, Santos e Furtado (2008) e “Ambiente Web de Suporte ao Processo de Avaliação da Qualidade de Produtos de Software” de Borges (2006). Para o processo de avaliação utilizando a metodologia CERTICS somente foi encontrada a ferramenta “CERTICSys” de CTI Renato Archer (2013d). Todos estes trabalhos serão brevemente descritos a seguir.

2.3.1 WISE: Uma Abordagem de Ferramenta de Software Para o Auxílio no Modelo de Avaliação do MPS.BR

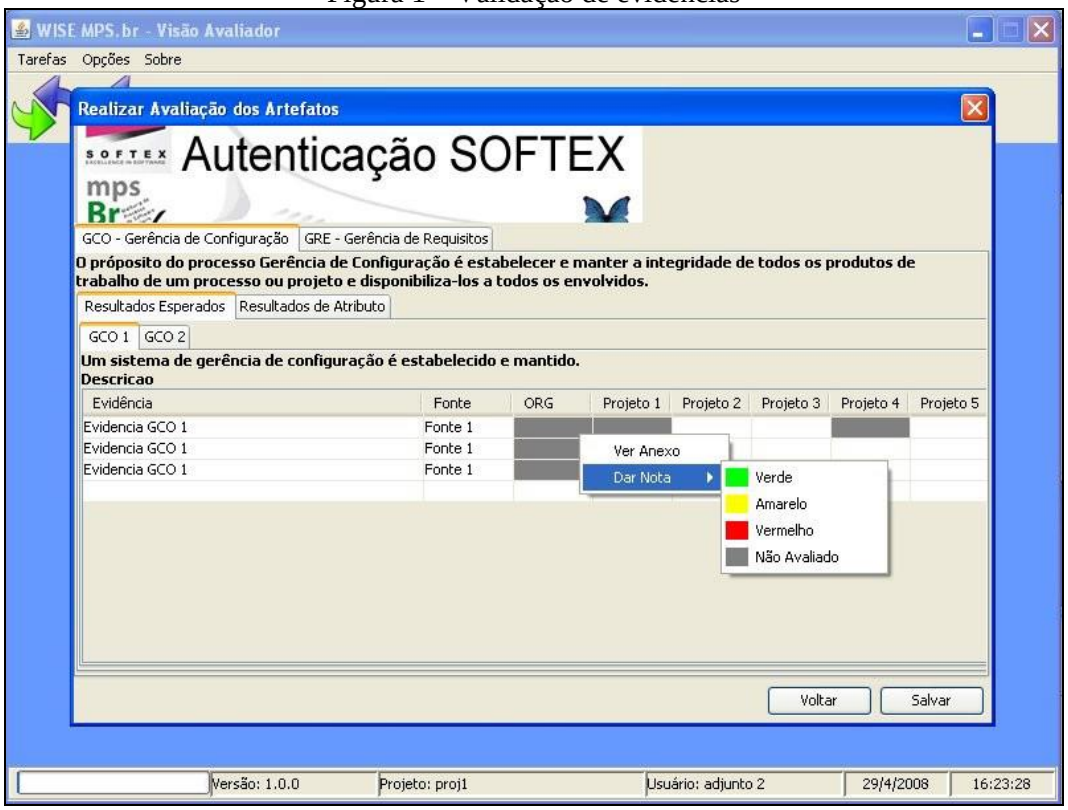
A ferramenta proposta por Albuquerque Júnior, Santos e Furtado (2008) tem como objetivo atender as necessidades dos *stakeholders* do processo de avaliação seguindo o modelo MPS.BR. O propósito desta ferramenta é tanto ajudar os avaliados como os avaliadores envolvidos no processo de certificação.

Para o desenvolvimento da ferramenta foi utilizada a tecnologia JAVA como linguagem de programação e foi criada uma arquitetura voltada para acesso em ambiente *desktop*.

Na ferramenta foi definido um cenário com três perfis, sendo eles o administrador, que possui a responsabilidade de definir a estrutura de avaliação imposta pelo MPS.BR, o avaliador, que possui a tarefa de realizar as avaliações com as organizações e, por fim, os avaliados, que são os representantes da organização avaliada e que possuem permissões para cadastrar as evidências necessárias para a avaliação.

A tela apresentada na Figura 1 identifica o objetivo principal do avaliador que é a validação das evidências cadastradas pela organização avaliada. Nesta validação são atribuídas cores às evidências, que depois são utilizadas como base para a análise da avaliação na geração da nota final.

Figura 1 – Validação de evidências



Fonte: Albuquerque Júnior, Santos e Furtado (2008, p. 55).

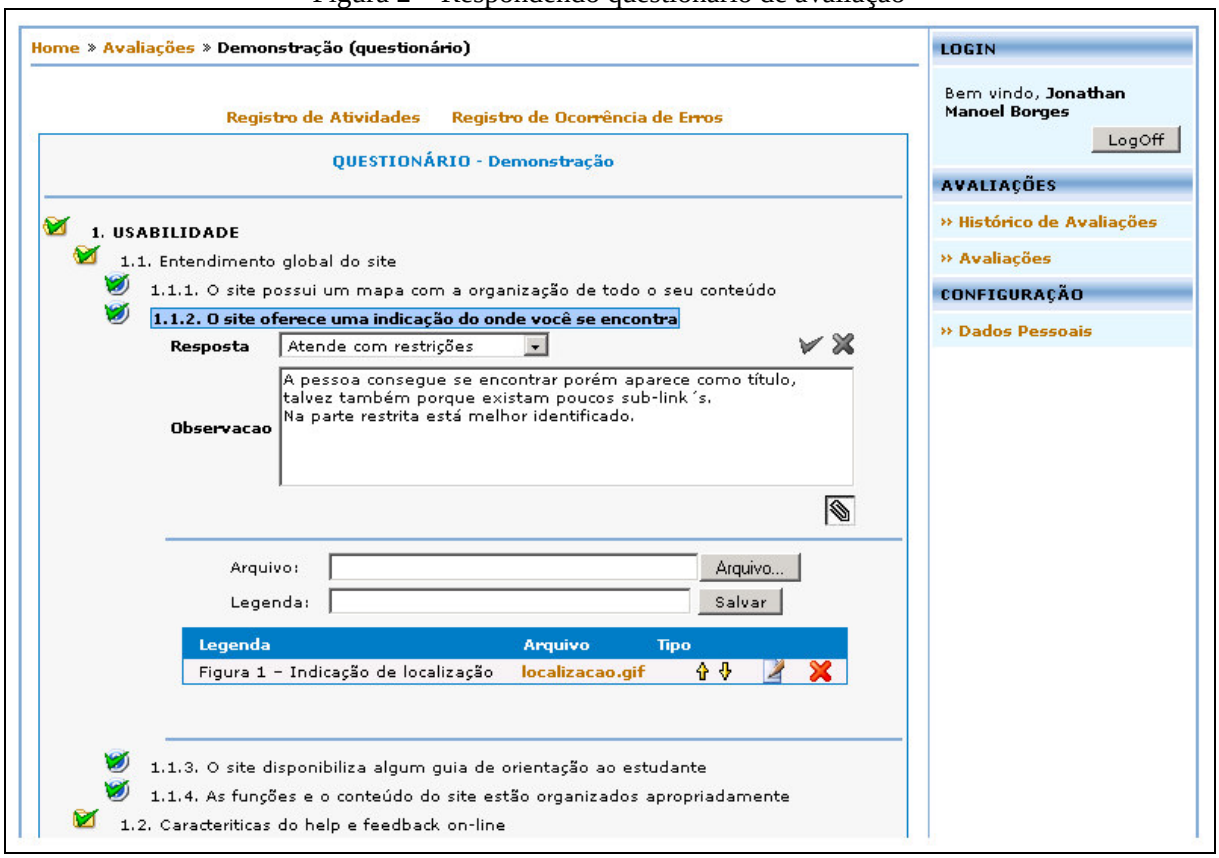
2.3.2 Ambiente Web de Suporte ao Processo de Avaliação da Qualidade de Produtos de Software

O trabalho de Borges (2006) teve por objetivo desenvolver uma ferramenta para auxiliar no processo de avaliação de produtos de software. Como estrutura da avaliação é utilizada a norma NBR ISO/IEC 14598-5 e o método de avaliação da qualidade de produto de software MEDE-PROS, definido pelo Centro de Pesquisas Renato Archer (CenPRA). A ferramenta disponibiliza funcionalidades como o registro de evidências, avaliação do software baseado nas evidências registradas e relatórios com os resultados das avaliações.

Para o desenvolvimento da ferramenta foi utilizada como linguagem de programação PHP na versão 5, banco de dados MySQL e foram utilizados *HyperText Markup Language* (HTML), Javascript e Ajax para a implementação de interfaces com os usuários.

A tela apresentada na Figura 2 ilustra a funcionalidade de preenchimento das respostas nos questionários disponibilizados pelo avaliador.

Figura 2 – Respondendo questionário de avaliação



Fonte: Borges (2006, p. 93).

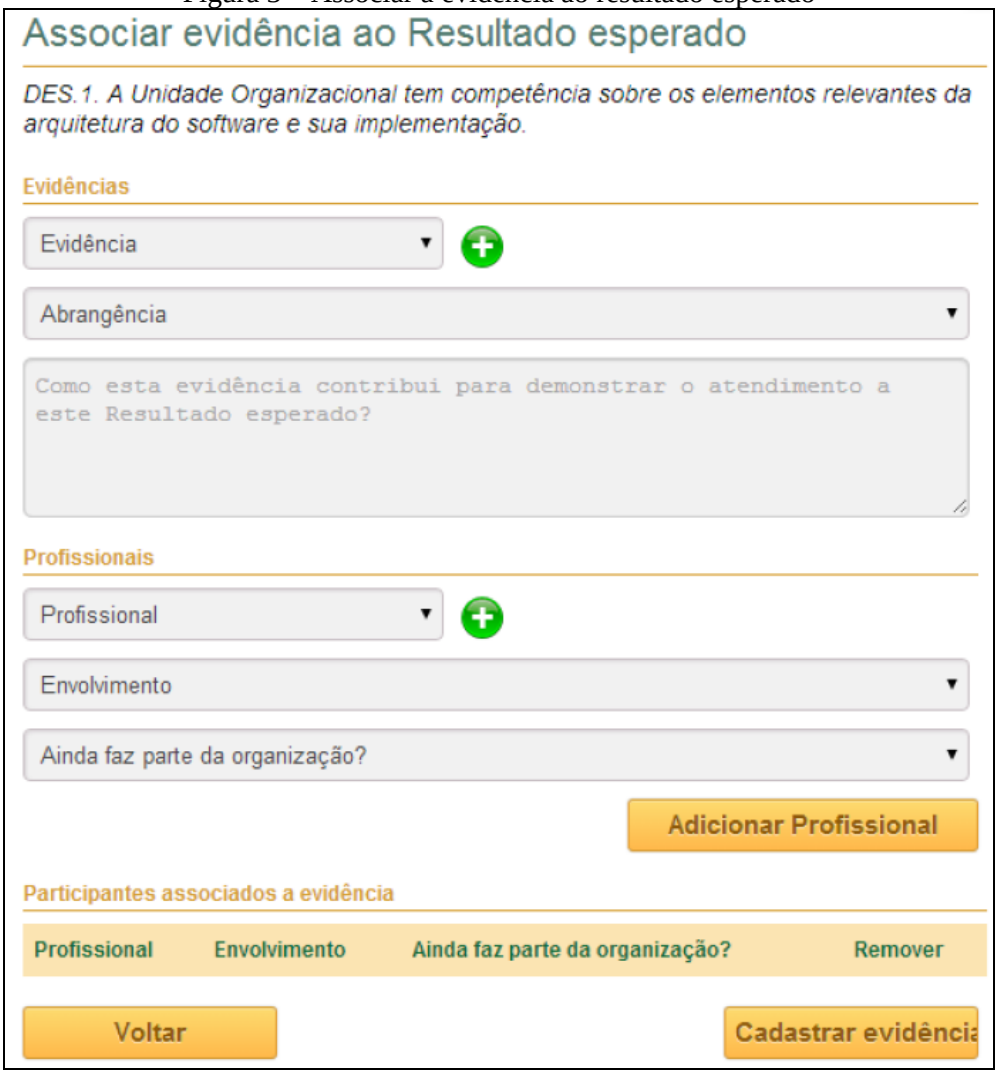
2.3.3 CERTICSys

CERTICSys é a plataforma de apoio responsável pelo suporte a todas as fases do método de avaliação da CERTICS. Ela disponibiliza que parte do processo de avaliação seja realizada pela *web*, o que diminui o tempo de avaliação do software (CTI Renato Archer, 2013d).

Criada para gerenciar todas as etapas de um processo de avaliação, permitindo o armazenamento, o monitoramento e o compartilhamento das informações cadastrais e das evidências de uma avaliação, a plataforma facilita a comunicação entre os envolvidos e gera os resultados da avaliação. São disponibilizadas funcionalidades que permitem desde cadastrar um software para avaliação e simular o uso da metodologia CERTICS, disponibilizando informações que mostram qual a possibilidade do software ser aprovado na avaliação CERTICS (CTI Renato Archer, 2013d).

A tela apresentada na Figura 3 ilustra a funcionalidade de associar a evidência ao resultado esperado.

Figura 3 – Associar a evidência ao resultado esperado



Fonte: CTI RENATO ARCHER (2013d, p. 22).

3 DESENVOLVIMENTO

Neste capítulo serão apresentadas as etapas do desenvolvimento da ferramenta web de suporte a avaliação de software com a metodologia CERTICS. Na seção 3.1 são enumerados os principais requisitos da ferramenta a ser desenvolvida. A seção 3.2 apresenta sua especificação, a seção 3.3 mostra os detalhes da implementação e por fim, a seção 3.4 exibe os resultados obtidos na ferramenta.

3.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

Baseado nos trabalhos correlatos pesquisados e na descrição da metodologia CERTICS foram identificados os seguintes requisitos:

a) permitir o cadastro de usuário (Requisito Funcional – RF);
b) permitir a definição de áreas de competência (RF);
c) permitir a definição dos resultados esperados (RF);
d) permitir o cadastro de avaliador (RF);
e) permitir o cadastro de organização (RF);
f) permitir o cadastro de profissionais da organização (RF);
g) permitir o cadastro de softwares (RF);
h) permitir o cadastro de evidências (RF);
i) permitir o cadastro de anexos (RF);
j) permitir a definição de escalas de pontuação (RF);
k) permitir a criação de uma avaliação de software (RF);
l) permitir o vínculo de evidências em uma avaliação (RF);
m) permitir o registro de observações sobre uma avaliação (RF);
n) permitir o cálculo do resultado de uma avaliação (RF);
o) permitir a geração de um gráfico de atendimento das áreas de competência (RF);
p) permitir a geração de um relatório do resultado final da avaliação detalhado (RF);
q) utilizar a linguagem de programação Java 1.7 (Requisito Não-Funcional - RNF);
r) utilizar o ambiente de desenvolvimento Eclipse 4.3 (RNF);
s) utilizar JSF 2.0 com o *framework* PrimeFaces para que a ferramenta ofereça suporte web (RNF);
t) utilizar JasperReports (RNF);
u) utilizar iReport (RNF);
v) utilizar Banco de dados PostgreSQL (RNF).

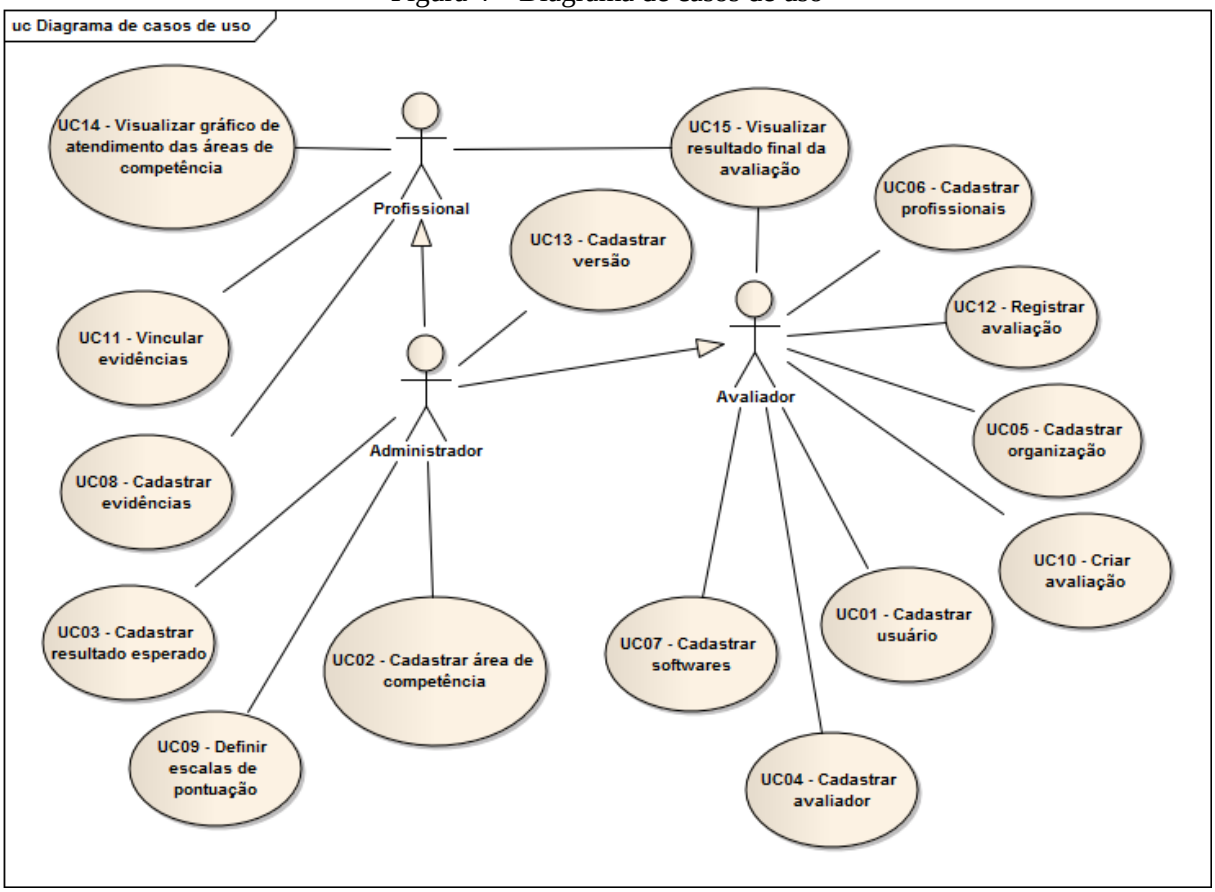
3.2 ESPECIFICAÇÃO

A seguir será apresentada a especificação da ferramenta, representada por diagramas da *Unified Modeling Language* (UML) e modelada utilizando-se a ferramenta Enterprise Architect 7.5. Foram utilizados os diagramas de casos de uso, classes e atividades. Também foi especificado o Modelo Entidade Relacionamento (MER) utilizado na modelagem dos dados, com a ferramenta DBDesigner 4.

3.2.1 Diagrama de casos de uso

Nesta seção são descritos os casos de uso com as funcionalidades da ferramenta. Foram identificados três atores: o `administrador`, o `avaliador` e o `profissional`. O `administrador` é responsável por cadastrar as regras definidas na CERTICS. O `avaliador`, sendo o responsável por aplicar as avaliações para a organização solicitante e avaliar as evidências vinculadas. O `profissional`, que será representante da organização solicitante no processo de avaliação do software e o responsável por cadastrar e vincular as evidências. Na figura 4 é apresentado o diagrama de casos de uso e o detalhamento dos três principais casos de uso encontra-se no Apêndice A.

Figura 4 – Diagrama de casos de uso



A seguir são descritos os casos de uso exibidos na Figura 4:

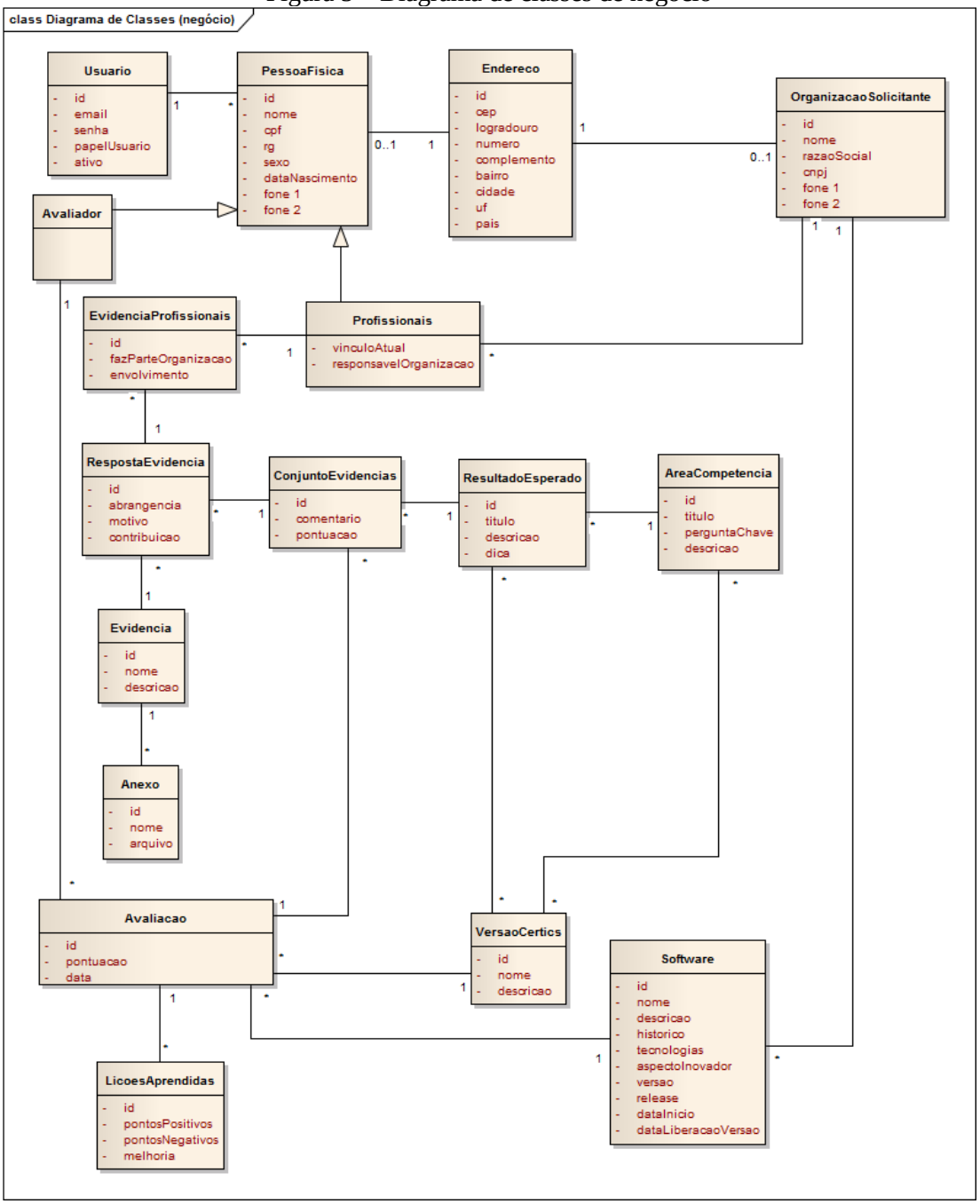
a) UC01 – Cadastrar usuário: permite cadastrar os usuários que vão ter acesso a ferramenta;
b) UC02 – Cadastrar área de competência: permite cadastrar as áreas de competências definidas na metologia CERTICS;
c) UC03 – Cadastrar resultado esperado: permite cadastrar os resultados esperados de uma área de competência definidas na metodologia CERTICS;
d) UC04 – Cadastrar avaliador: permite cadastrar o avaliador que será responsável por aplicar e avaliar as evidências;
e) UC05 – Cadastrar organização: permite cadastrar a organização que está solicitando a avaliação de seu software;
f) UC06 – Cadastrar profissionais: permite cadastrar os profissionais da organização que está solicitando a avaliação de seu software;
g) UC07 – Cadastrar softwares: permite cadastrar os softwares de uma organização que está solicitando a avaliação dos mesmos;
h) UC08 – Cadastrar evidências: permite cadastrar evidências e anexos que serão utilizadas para o vínculo de evidências em uma avaliação;
i) UC09 – Definir escalas de pontuação: permite a definição das escalas que serão utilizados pelo avaliador na avaliação das evidências registradas na ferramenta;
j) UC10 – Criar avaliação: permite criar uma avaliação de um software de uma organização solicitante com base em uma versão definida pelo avaliador;
k) UC11 – Vincular evidências: permite o profissional vincular as evidências cadastradas ao software que está sendo avaliado;
l) UC12 – Registrar avaliação: permite o avaliador registrar as observações das evidências do software vinculadas pelo profissional em uma avaliação;
m) UC13 – Cadastrar versão: permite cadastrar uma versão do modelo para definir o escopo de uma avaliação;
n) UC14 – Visualizar gráfico de atendimento das áreas de competência: permite o profissional emitir relatórios com informações de pendências da avaliação;

o) `UC15 – Visualizar resultado final da avaliação`: proporciona ao `avaliador` e ao `profissional` a geração de um relatório da avaliação detalhado com as evidências registradas.

3.2.2 Diagrama de classes

As entidades são responsáveis pela lógica de negócio. Esta camada tem como responsabilidade efetuar a persistência dos dados no banco de dados. Na figura 5 é apresentado o diagrama de classes da camada de negócio da ferramenta.

Figura 5 – Diagrama de classes de negócio



A seguir são descritas as principais classes exibidas na Figura 5:

a) `VersaoCertics`: armazena quais são as versões que serão utilizadas para as avaliações e quais as áreas de competências e resultados esperados que estarão vinculadas a versão definida pelo `administrador`;
b) `AreaCompetencia`: armazena as informações das áreas de competências definidas na metologia CERTICS;
c) `ResultadoEsperado`: armazena as informações dos resultados esperados vinculados a uma área de competência definido na metodologia CERTICS;
d) `PessoaFisica`: armazena as informações referentes a uma pessoa física registrada na ferramenta;
e) `Avaliador`: armazena a pessoa física registrada que possui o papel de `avaliador`;
f) `Profissionais`: armazena a pessoa física registrada que possui papel de `profissional` de uma organização solicitante;
g) `OrganizacaoSolicitante`: armazena as informações referentes à organização que está solicitando a avaliação de um software;
h) `Endereco`: armazena as informações referentes ao endereço das pessoas físicas ou organizações solicitantes registradas na ferramenta;
i) `Usuario`: armazena as informações que permite às pessoas físicas registradas acessar a ferramenta;
j) `Software`: armazena as informações do software de uma organização solicitante que passa por uma avaliação na ferramenta;
k) `Avaliacao`: armazena o software que está sendo avaliado, o avaliador que está avaliando as evidências vinculadas pelo `profissional` da organização solicitante e qual a pontuação final da avaliação;
l) `Evidencia`: armaneza as informações de uma evidência cadastrada na ferramenta que será utilizada para o vínculo de evidências em uma avaliação;
m) `Anexo`: armaneza os anexos referentes a uma evidência cadastrada na ferramenta;
n) `ConjuntoEvidencias`: armazena as observações do `avaliador` sobre as evidências vinculadas e define a pontuação de cada resultado esperado de uma área de competência;

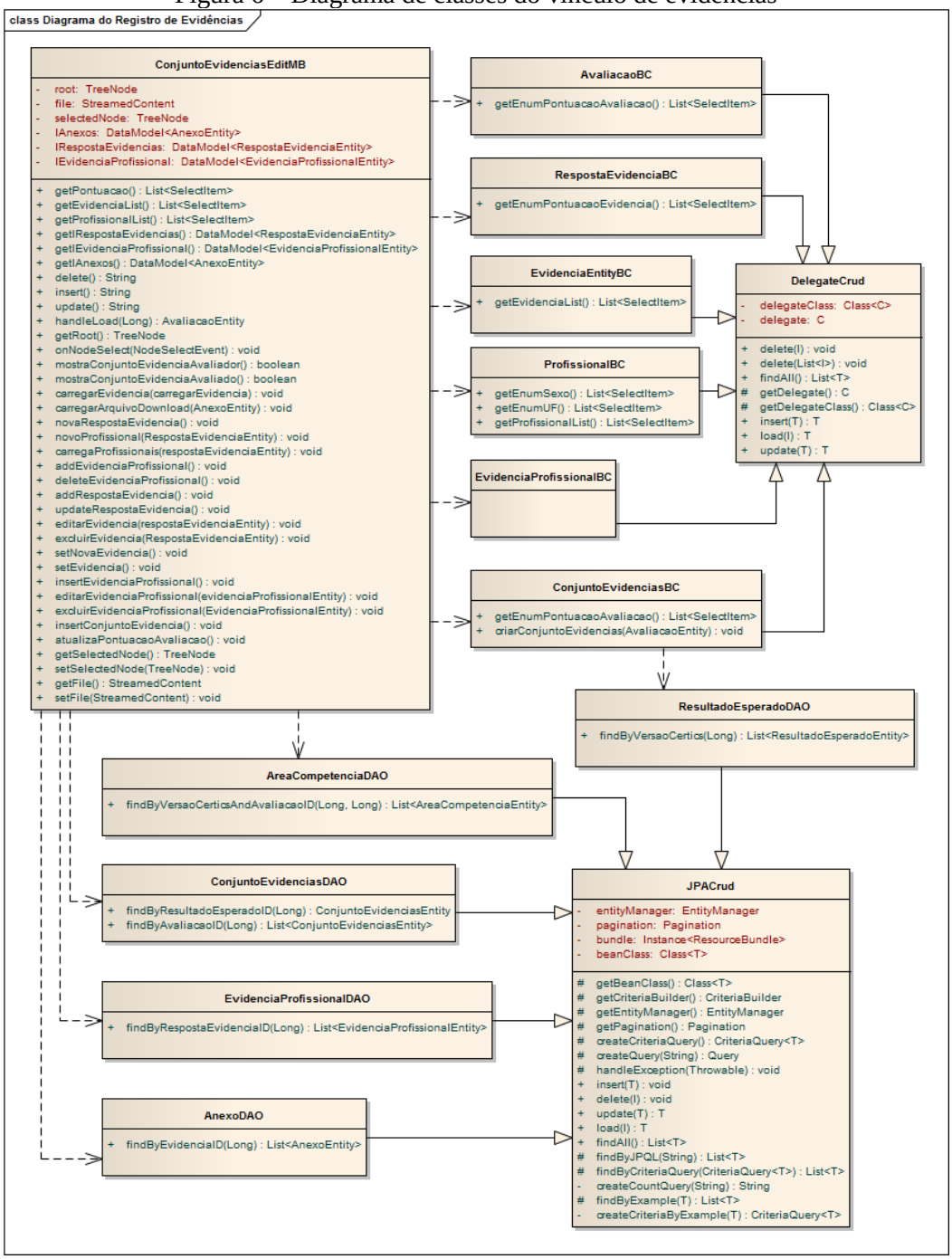
o) `RespostaEvidencia`: armazena as evidências vinculadas ao software pelo `profissional` da organização solicitante de um determinado resultado esperado;
p) `EvidenciaProfissionais`: armazena os profissionais vinculados às evidências de uma avaliação de um software da organização solicitante;
q) `LicoesAprendidas`: armazena as informações das lições aprendidas de uma avaliação.

Na figura 6 é apresentado o diagrama de classes referente à funcionalidade do vínculo de evidências ao software. Mostra-se a seguir as classes utilizadas:

a) `ConjuntoEvidenciasEditMB`: realiza os processos requisitados pelo usuário no vínculo de evidências. Essa classe realiza a comunicação com os outros objetos envolvidos no processo;
b) `AvaliacaoBC`: realiza o registro das informações referentes à avaliação;
c) `RespostaEvidenciaBC`: realiza o registro das informações referentes às respostas das evidências;
d) `EvidenciaEntityBC`: realiza o registro das informações referentes às evidências;
e) `ProfissionalBC`: realiza o registro das informações referentes aos profissionais;
f) `EvidenciaProfissionalBC`: realiza o registro das informações referentes às evidências vinculadas aos profissionais envolvidos;
g) `ConjuntoEvidenciasBC`: realiza o registro das informações dos resultados das avaliações;
h) `DelegateCrud`: realiza o registro e a seleção das informações dos objetos;
i) `ResultadoEsperadoDAO`: realiza a seleção das informações referentes aos resultados esperados da avaliação;
j) `AreaCompetenciaDAO`: realiza a seleção das informações referentes as áreas de competências da avaliação;
k) `ConjuntoEvidenciasDAO`: realiza a seleção das informações referentes aos resultados e informações da avaliação;
l) `EvidenciaProfissionalDAO`: realiza a seleção das informações das evidências dos profissionais envolvidos na avaliação;
m) `AnexoDAO`: realiza a seleção dos anexos vinculados às evidências;

n) `JPACrud`: realiza os processos de seleção de informações dos objetos.

Figura 6 – Diagrama de classes do vínculo de evidências

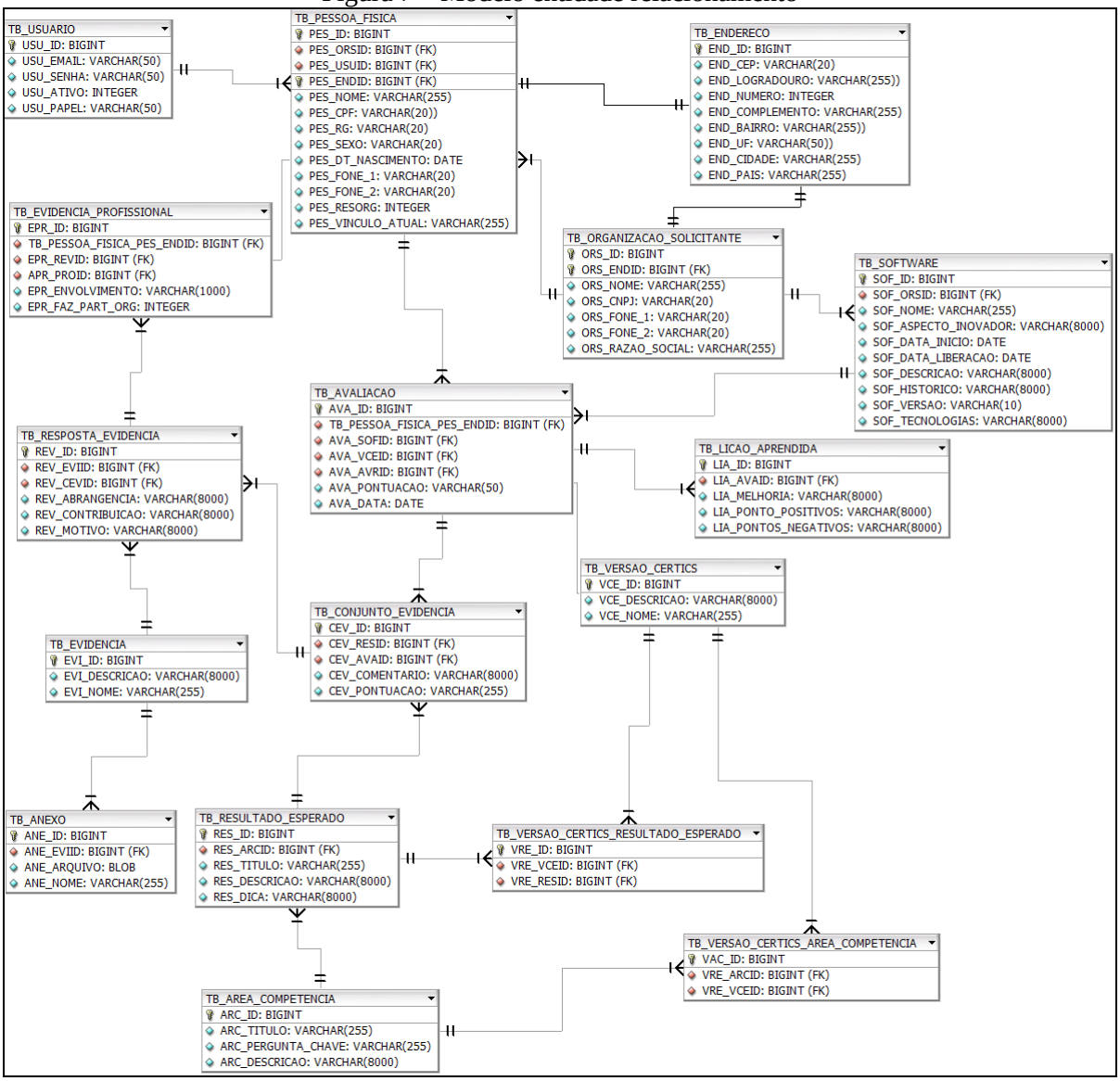


Na Figura 6 é identificado que a classe `ConjuntoEvidenciasEditMB` possui dependências de outras classes para que se realize o processo de recuperação e registro de informações. Também pode ser identificado que as classes herdam funcionalidades das classes `JPACrud` e `DelegateCrud`, que pertencem ao *framework Demoiselle* e realizam as operações de recuperar e registrar informações.

3.2.3 Diagrama de entidade relacionamento

Na figura 7 é mostrada a base de dados criada utilizando-se como base a camada de entidades da ferramenta. No diagrama foram especificadas as tabelas, colunas, chaves primárias e estrangeiras. No Apêndice B encontra-se o dicionário de dados.

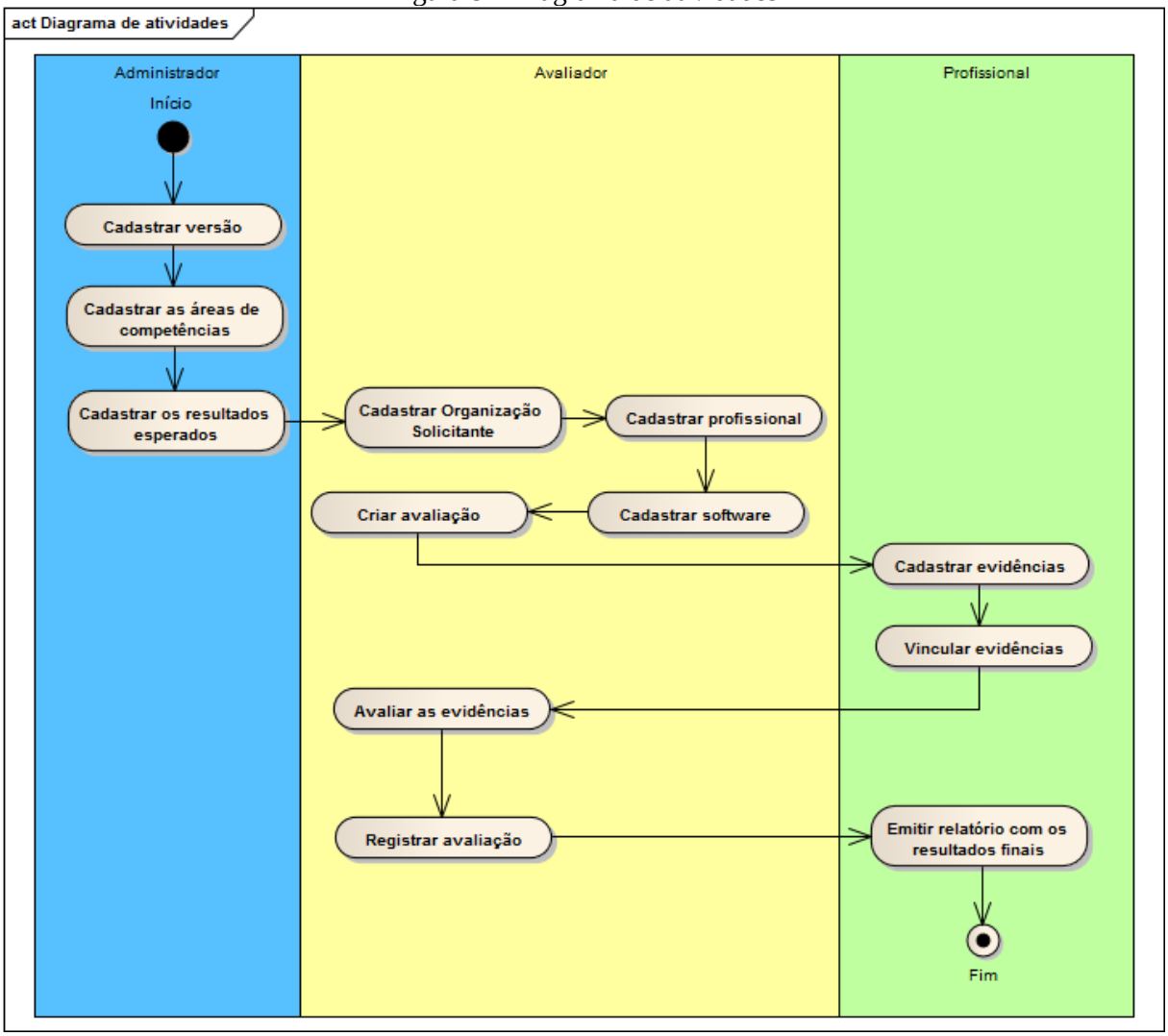
Figura 7 – Modelo entidade relacionamento



3.2.4 Diagrama de atividades

Na Figura 8 é apresentado o diagrama de atividades que irá exibir o fluxo da avaliação do software de uma organização solicitante no qual o avaliador é o responsável por criar a avaliação e avaliar as evidências vinculadas pelo profissional da organização.

Figura 8 – Diagrama de atividades



O processo de avaliação se inicia com o `administrador` cadastrando uma versão. Em seguida são definidas as áreas de competência e resultados esperados que possuam como objetivo serem utilizados nas avaliações dos softwares aplicadas pelos avaliadores.

Posteriormente, o `avaliador` deve cadastrar a organização solicitante da avaliação, o `profissional` responsável por vincular as evidências, o software que será avaliado e criar a avaliação do software definido.

Após a criação da avaliação, o `profissional` deve cadastrar e vincular as evidências referentes aos resultados esperados definidos pelo `avaliador`. O `avaliador` analisará as evidências vinculadas. Após finalizar o registro da avaliação ficará disponível para o `profissional` emitir um relatório com os resultados finais da avaliação.

3.3 IMPLEMENTAÇÃO

A seguir estão elencadas as técnicas e ferramentas utilizadas e a operacionalidade da implementação.

3.3.1 Técnicas e ferramentas utilizadas

Foi escolhida a linguagem JAVA na versão 7 para o desenvolvimento da ferramenta. Como ambiente de desenvolvimento foi utilizado o Eclipse 4.3.

As tecnologias utilizadas na implementação da ferramenta (Figura 9) encontram-se descritas a seguir:

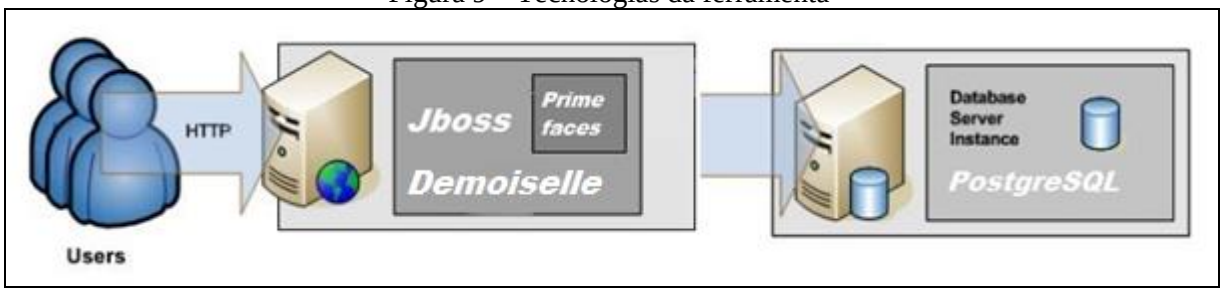
a) *Framework Demoiselle*: implementa o conceito de *framework* integrador. Seu objetivo é facilitar a construção de aplicações minimizando tempo dedicado à escolha e integração de *frameworks* especialistas, o que resulta no aumento da produtividade e garante a manutenibilidade dos sistemas. Disponibiliza mecanismos reusáveis voltados às funcionalidades mais comuns de uma aplicação (arquitetura, segurança, transação, mensagem, configuração, tratamento de exceções, etc);

b) JBoss AS 7.1.1.Final: O *JBoss Application Server* é um servidor de aplicação de código aberto desenvolvido pela JBoss para a plataforma *Java Enterprise Edition* (Java EE). Ele oferece toda a infraestrutura necessária para executar aplicações web desenvolvidas sobre essa plataforma. O JBoss AS 7 é completamente compatível com a especificação Java EE 6;

c) PrimeFaces: é uma biblioteca de componentes de código aberto para o JSF na versão 2.0 com mais de 100 componentes, permitindo criar interfaces ricas para aplicações web de forma simplificada e eficiente. Ele é considerado muito melhor do que outras bibliotecas de componentes JSF;

d) PostgreSQL: é um Sistema Gerenciador de Banco de Dados Objeto Relacional (SGBDOR), desenvolvido como projeto de código aberto. Hoje, o PostgreSQL é um dos SGBDs (Sistema Gerenciador de Bancos de Dados) de código aberto mais avançados. Para manipulação e consultas de dados diretamente no banco de dados foi utilizado o ambiente pgAdmin III.

Figura 9 – Tecnologias da ferramenta



3.3.2 Implementação da ferramenta

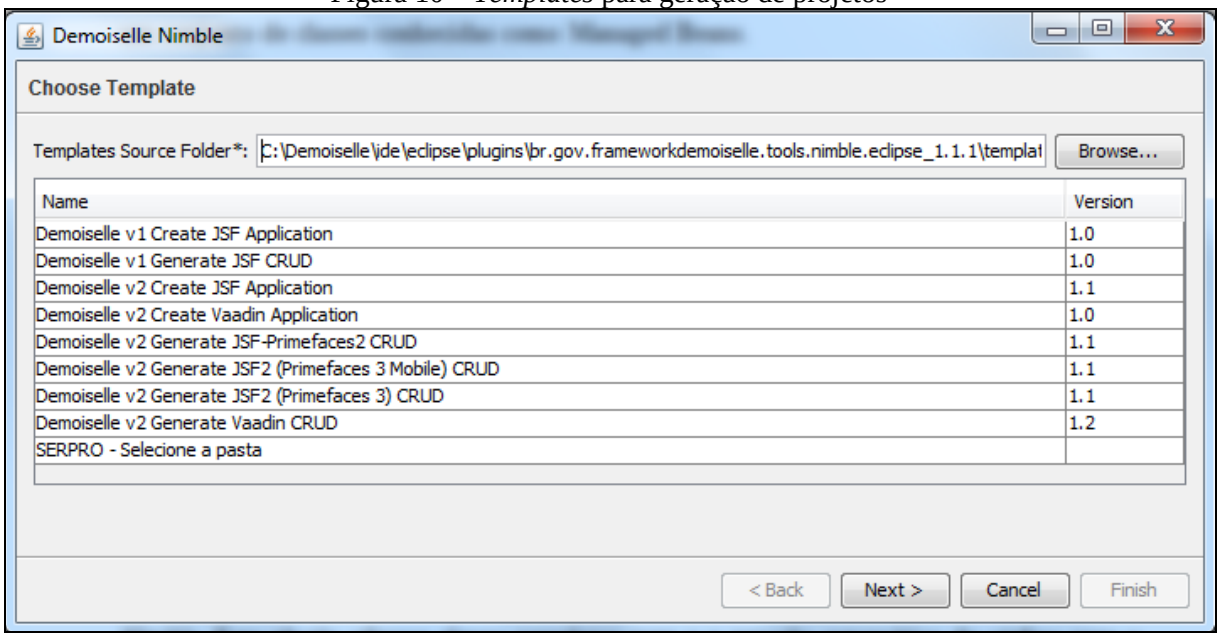
Para o desenvolvimento da ferramenta foi utilizada a especificação JSF juntamente com a biblioteca de componentes PrimeFaces. Esta biblioteca possui diversos componentes visuais que ajudam a criar páginas na web sem a utilização diretamente de JavaScript e HTML. Outra característica importante na arquitetura JSF é a separação entre as camadas de apresentação e de aplicação, sendo utilizado o modelo MVC. O JSF possui uma camada de visualização que, por padrão, se baseia nos Facelets e um conjunto de classes conhecidas como *Managed Beans*.

Uma das principais vantagens de utilizar esta biblioteca de componentes é que a mesma permite a criação de templates, ou seja, uma página padrão, sendo utilizada por todas as outras páginas do sistema, com isso as páginas do sistema herdam de sua estrutura uma aparência visual padronizada. O PrimeFaces também possibilita a utilização de temas para a aparência visual do sistema, sendo que no desenvolvimento foi utilizado o tema *south-street* para o *layout* das páginas. A disponibilidade de temas facilita deixar a ferramenta com um visual melhor e evita que se tenha que usar diretamente *Cascading Style Sheets* (CSS).

Para a construção da arquitetura da ferramenta seguindo o padrão MVC foi utilizado o *framework Demoiselle*, tendo ele como objetivo a completa aderência à especificação JAVA EE 6. O *Demoiselle* disponibiliza um ambiente com o Eclipse 4.3 pronto para desenvolvimento.

No Eclipse há um plugin para auxiliar na criação dos projetos, chamado *Demoiselle Nimble*. Este plugin oferece alguns *templates* para a geração automática de código para o *framework Demoiselle*. Na Figura 10 é possível visualizar os *templates* disponíveis.

Figura 10 – *Templates* para geração de projetos



Para gerar a ferramenta foi utilizado o *template Demoiselle V2 Create JSF Application*, que tem por finalidade criar uma aplicação completa e funcional baseada no *framework Demoiselle* com JSF e *Java Persistence Application Programming Interface* (JPA). Ainda sobre o *template*, ele gera os códigos para se iniciar um projeto com a arquitetura do *Demoiselle* utilizando os seus modulos *Core*, Extensões e Componentes.

O *Core* do *Demoiselle* contém aquelas funcionalidades comuns a todas as extensões e aplicações. O *Core* é simples, leve e formado majoritariamente por interfaces e poucas implementações. O *Core* é a base do *framework*. Sem ele, as extensões e a própria aplicação não funcionariam.

As extensões, como o próprio nome sugere, estendem o *Core* com funcionalidades extras e bem específicas a um domínio ou tecnologia. Neste contexto, caso a aplicação necessite de persistência com JPA, o *framework* fornecerá facilidades. Contudo, estas funcionalidades não estão no *Core*. Para este propósito existem as extensões como a *demoiselle-jpa*, por exemplo. Cabe destacar que as extensões não possuem vida própria, pois estão diretamente ligadas ao núcleo do *framework*, inclusive o ciclo de vida das extensões está totalmente acoplado ao do *Core*.

Já os componentes são artefatos separados e que, portanto, não são dependentes diretamente do *Core*. Aliás, os componentes podem até mesmo existir sem referenciar o *Core*.

O *Demoiselle* faz uso da solução proposta pelo *Apache Maven* para diversas fases do desenvolvimento de software. O artefato principal do *Maven* é o `pom.xml`, que é o arquivo *eXtensible Markup Language* (XML) que contém todas as informações necessárias para a

ferramenta gerenciar o projeto, entre as quais está o gerenciamento de dependências (bibliotecas), *build* do projeto, etc. Segue na Figura 11 o arquivo `pom.xml` da ferramenta.

Figura 11 – Arquivo `pom.xml`

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://maven.apache.org/POM/4.0.0"
3      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
4
5      <modelVersion>4.0.0</modelVersion>
6
7      <groupId>vinicius.ferneda.tcc</groupId>
8      <artifactId>certics</artifactId>
9      <version>0.0.1-SNAPSHOT</version>
10     <packaging>war</packaging>
11
12     <name></name>
13     <description></description>
14     <url></url>
15
16     <parent>
17         <groupId>br.gov.frameworkdemoiselle</groupId>
18         <artifactId>demoiselle-jsf-parent</artifactId>
19         <version>2.4.0</version>
20     </parent>
21
22     <dependencies>
23         <dependency>
24             <groupId>br.gov.frameworkdemoiselle</groupId>
25             <artifactId>demoiselle-jpa</artifactId>
26             <scope>compile</scope>
27         </dependency>
28
29         <dependency>
30             <groupId>br.gov.frameworkdemoiselle.component</groupId>
31             <artifactId>demoiselle-report</artifactId>
32             <version>2.2.0</version>
33             <scope>compile</scope>
34         </dependency>
35
36         <dependency>
37             <groupId>org.primefaces</groupId>
38             <artifactId>primefaces</artifactId>
39             <scope>compile</scope>
40         </dependency>
41 </project>
```

Como pode ser visto na Figura 11, na linha 18 foi utilizado o pacote *demoiselle-jsf-parent* que contém configurações úteis e necessárias para todas as aplicações que utilizarão a tecnologia JSF 2.0 para camada de apresentação. A extensão *demoiselle-jpa* será responsável por delegar o controle das transações para o `javax.persistence.EntityManager` da especificação JPA. O componente *demoiselle-report* (linha 31) será o responsável pela geração e exportação dos relatórios. E a bilbioteca PrimeFaces será responsável pelos componentes visuais da ferramenta.

O plugin *Demoiselle Nimble* também gera todos os pacotes para a criação das classes da ferramenta e as classes de configuração `web.xml`, `persistence.xml` e `beans.xml`. Nas Figuras 12, 13 e 14 são exibidos os arquivos com suas devidas configurações. Estes arquivos são responsáveis respectivamente pela delegação de requisições com o JSF, configuração dos

objetos que vão persistir as informações no banco de dados e delegação do interceptador das transações de segurança e exceções.

Figura 12 – Arquivo `web.xml`

```xml
<?xml version="1.0"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
    version="3.0">
    <filter>
        <filter-name>Character Encoding Filter</filter-name>
        <filter-class>vinicius.ferneda.tcc.certics.util.CharacterEncodingFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>Character Encoding Filter</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
    <filter>
        <filter-name>PrimeFaces FileUpload Filter</filter-name>
        <filter-class>org.primefaces.webapp.filter.FileUploadFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>PrimeFaces FileUpload Filter</filter-name>
        <servlet-name>Faces Servlet</servlet-name>
    </filter-mapping>
    <context-param>
        <param-name>primefaces.THEME</param-name>
        <param-value>south-street</param-value>
    </context-param>
    <servlet>
        <servlet-name>Faces Servlet</servlet-name>
        <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>Faces Servlet</servlet-name>
        <url-pattern>*.jsf</url-pattern>
    </servlet-mapping>
    <security-constraint>
        <display-name>Restrict raw XHTML Documents</display-name>
        <web-resource-collection>
            <web-resource-name>XHTML</web-resource-name>
            <url-pattern>*.xhtml</url-pattern>
        </web-resource-collection>
        <auth-constraint />
    </security-constraint>
</web-app>
```

Figura 13 – Arquivo `persistence.xml`

```xml
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.0"
    xmlns="http://java.sun.com/xml/ns/persistence"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
    http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">

    <persistence-unit name="certics-ds" transaction-type="RESOURCE_LOCAL">

        <non-jta-data-source>java:jboss/datasources/CerticsDS</non-jta-data-source>

        <class>vinicius.ferneda.tcc.certics.domain.AnexoEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.AreaCompetenciaEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.AvaliacaoEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.AvaliadorEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.ConjuntoEvidenciasEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.EnderecoEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.EvidenciaEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.EvidenciaProfissionalEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.LicaoAprendidaEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.OrganizacaoSolicitanteEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.ProfissionalEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.RespostaEvidenciaEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.ResultadoEsperadoEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.SoftwareEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.UsuarioEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.VersaoCerticsEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.VersaoCerticsAreaCompetenciaEntity</class>
        <class>vinicius.ferneda.tcc.certics.domain.VersaoCerticsResultadoEsperadoEntity</class>

        <properties>
            <property name="hibernate.dialect" value="org.hibernate.dialect.PostgreSQLDialect"/>
            <property name="hibernate.show_sql" value="true" />
            <property name="hibernate.hbm2ddl.auto" value="update"/>
            <property name="hibernate.connection.charSet" value="UTF-8" />
        </properties>
    </persistence-unit>

</persistence>
```

Figura 14 – Arquivo `beans.xml`

```xml
<beans xmlns="http://java.sun.com/xml/ns/javaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/beans_1_0.xsd">

    <interceptors>
        <class>br.gov.frameworkdemoiselle.transaction.TransactionalInterceptor</class>
        <class>br.gov.frameworkdemoiselle.security.RequiredPermissionInterceptor</class>
        <class>br.gov.frameworkdemoiselle.security.RequiredRoleInterceptor</class>
        <class>br.gov.frameworkdemoiselle.exception.ExceptionHandlerInterceptor</class>
    </interceptors>

</beans>
```

Após a geração do projeto e a criação das classes de domínio da ferramenta foi utilizado o template *Demoiselle v2 Generate JSF2 (PrimeFaces 3) Create, Read, Update and Delete* (CRUD) para criar todos os artefatos necessários para o CRUD de uma dada entidade. Na geração de código são gerados os arquivos *.xhtml* que são responsáveis pela parte visual da ferramenta e as camadas de visão, negócio e persistência que são anotadas respectivamente pelos estereótipos `@ViewController`, `@BusinessController` e *`@PersistenceController`*.

Para exemplificar, a arquitetura utilizada pelo *Demoiselle* será empregada à funcionalidade de vínculo de evidências de uma avaliação. Essa é a principal funcionalidade na ferramenta, tendo como objetivo vincular as evidências pelo profissional da organização, a avaliação das evidências por parte do avaliador e o cálculo da pontuação da avaliação.

Para organizar as informações foi utilizado o componente do Primefaces `p:layout` e seu derivado `p:layoutUnit`. O cabeçalho é formado pelas informações de nome do avaliador, nome da organização solicitante, nome do software e a pontuação da avaliação.

Para as informações das áreas de competência e resultados esperados da avaliação, foi criada uma estrutura de árvore com o componente `p:tree` dentro de outro `p:layoutUnit` para as informações serem mostradas abaixo do cabeçalho e a esquerda do *`p:layoutUnit`* principal. Na figura 15 é possível visualizar como foi criada a árvore de informações das áreas de competência e resultados esperados.

Figura 15 – Árvore de informações das áreas de competência e resultados esperados

```
1  <p:layoutUnit id="treeRegistroEvidencias" position="west"
2  resizable="true" size="20%">
3      <p:tree id="tree" value="#{conjuntoEvidenciasEditMB.root}" var="node"
4      cache="false" selectionMode="single"
5      selection="#{conjuntoEvidenciasEditMB.selectedNode}"
6      style="width:100%;height:100%">
7
8          <p:ajax event="select" listener="#{conjuntoEvidenciasEditMB.onNodeSelect}"
           update=":formRegistroEvidencias:panelPrincipalRegistroEvidencia,
          :formRegistroEvidencias:panelPrincipalRegistroEvidenciaAvaliador,
          :formRegistroEvidencias:panelPrincipalRegistroEvidenciaAvaliado,
          :formRegistroEvidencias:panelPrincipalComentarioAvaliador,
          :formRegistroEvidencias:dtEvidencias"/>
9
10         <p:treeNode id="treeNode">
11             <h:outputText id="lblNode" value="#{node.titulo}"/>
12         </p:treeNode>
13     </p:tree>
14 </p:layoutUnit>
```

Para as ações desta árvore foi necessário criar os métodos `getRoot` para a sua geração da árvore e `onNodeSelect` para ao selecionar um resultado esperado a ferramenta carregar as evidências. A seguir na Figura 16 é possível visualizar os métodos.

Figura 16 – Métodos responsáveis pelas ações da árvore

```
1  public TreeNode getRoot(){
2      List<AreaCompetenciaEntity> lAreaCompetencia = this.areaCompetenciaDAO.
       findByVersaoCerticsAndAvaliacaoID(this.getId(),
3      this.getBean().getVersaoCertics().getId());
4      this.root = new DefaultTreeNode("root", null);
5      for (AreaCompetenciaEntity areaCompetenciaEntity : lAreaCompetencia) {
6          TreeNode areaCompetencia = new DefaultTreeNode(new InformacoesArvore(
           areaCompetenciaEntity.getId(), areaCompetenciaEntity.getTitulo(),
           "areaCompetencia"), root);
7          for (ResultadoEsperadoEntity resultadoEsperadoEntity :
           areaCompetenciaEntity.getResultadosEsperados()) {
8              TreeNode resultadoEsperado = new DefaultTreeNode(new InformacoesArvore
               (resultadoEsperadoEntity.getId(), resultadoEsperadoEntity.getTitulo(),
                "resultadoEsperado"), areaCompetencia);
9          }
10     }
11     return this.root;
12 }
13
14 public void onNodeSelect(NodeSelectEvent event) {
15     if("resultadoEsperado".equals(((InformacoesArvore)event.getTreeNode().getData
       ()).getTipo())){
16         this.getBean().setConjuntoEvidenciasAux(this.conjuntoEvidenciasDAO.
           findByResultadoEsperadoID(((InformacoesArvore)event.getTreeNode().getData
           ()).getId()));
17     }
18 }
```

Estes métodos se encontram na classe `ConjuntoEvidenciasEditMB.java` que é anotada por `@ViewController`, sendo este o responsável pela comunição entre a parte do cliente e a parte do servidor da ferramenta.

No `p:layoutUnit` principal da tela estão as informações referentes às evidências da avaliação. O profissional possui as opções de vincular as evidências, inserir o motivo pelo qual ele não possui evidências do resultado esperado ou visualizar dicas de evidências. Para os registros dessas informações e a visualização da dica foi utilizado o componente `p:dialog`. Nas Figuras 17 e 18 é possível visualizar a organização dos componentes para as opções de registro de motivo e vínculo de evidências.

Figura 17 – Tela de registro de motivo

```
<p:dialog id="modalEvidenciaMotivo" header="#{messages['respostaEvidencia.dialog']}"
    widgetVar="dialogEvidenciaMotivo" width="600" height="200" resizable="false"
    closable="true">
    <h:panelGrid id="panelRegistroEvidenciaMotivo" styleClass="semBorda">
        <h:outputLabel value="#{messages['respostaEvidencia.label.motivo']}: "
        for="motivo" styleClass="text-input" />
        <p:inputTextarea id="motivo"
        value="#{conjuntoEvidenciasEditMB.bean.respostaEvidenciaAux.motivo}"
        title="#{messages['respostaEvidencia.alt.motivo']}" rows="5" cols="50"
        maxlength="1000" autoResize="false"/>
        <p:message for="motivo" />
    </h:panelGrid>
    <p:commandButton value="#{messages['button.save']}"
        actionListener="#{conjuntoEvidenciasEditMB.addRespostaEvidencia()}"
        update=":formRegistroEvidencias:dtEvidencias"
        oncomplete="dialogEvidencia.hide();"/>
</p:dialog>
```

Figura 18 – Tela de vínculo de evidências

```
<p:dialog id="modalEvidencia" header="#{messages['respostaEvidencia.dialog']}"
    widgetVar="dialogEvidencia" width="600" height="400" resizable="false"
    closable="true">
    <h:panelGrid id="panelRegistroEvidencia" styleClass="semBorda">
        <h:outputLabel value="#{messages['evidenciaEntity.label']}: "
        for="evidencia" styleClass="text-input" />
        <p:selectOneMenu id="evidencia"
        value="#{conjuntoEvidenciasEditMB.bean.respostaEvidenciaAux.evidencia}"
        effect="fade" converter="ConversorEvidencia">
            <f:selectItem itemLabel="Selecione" itemValue="" />
            <f:selectItems value="#{conjuntoEvidenciasEditMB.evidenciaList}"
            var="varEvidencia" itemLabel="#{varEvidencia.label}"
            itemValue="#{varEvidencia}" />
        </p:selectOneMenu>
        <p:message for="evidencia" />
        <h:outputLabel
        value="#{messages['respostaEvidencia.label.abrangencia']}: "
        for="abrangencia" styleClass="text-input" />
        <p:inputTextarea id="abrangencia"
        value="#{conjuntoEvidenciasEditMB.bean.respostaEvidenciaAux.abrangencia}"
        title="#{messages['respostaEvidencia.alt.abrangencia']}" rows="5"
        cols="50" maxlength="1000" autoResize="false"/>
        <p:message for="abrangencia" />
        <h:outputLabel
        value="#{messages['respostaEvidencia.label.contribuicao']}: "
        for="contribuicao" styleClass="text-input" />
        <p:inputTextarea id="contribuicao"
        value="#{conjuntoEvidenciasEditMB.bean.respostaEvidenciaAux.contribuicao}"
        title="#{messages['respostaEvidencia.alt.contribuicao']}" rows="5"
        cols="50" maxlength="1000" autoResize="false"/>
        <p:message for="contribuicao" />
    </h:panelGrid>
    <p:commandButton value="#{messages['button.save']}"
        actionListener="#{conjuntoEvidenciasEditMB.addRespostaEvidencia()}"
        update=":formRegistroEvidencias:dtEvidencias"
        oncomplete="dialogEvidencia.hide();"/>
</p:dialog>
```

As evidências vinculadas são disponibilizadas para visualização em uma tabela. Foi utilizado o componente `p:dataTable` para a criação da tabela, onde nela são exibidas as informações das evidências vinculadas, os anexos e para cada evidência será possível registrar os profissionais da organização solicitante que estão envolvidos. A tabela também será

visualizada pelo avaliador, porém ele não poderá alterá-la e nem inserir registros. Na Figura 19 é possível visualizar a organização das informações dentro do componente `p:dataTable`.

Figura 19 – Tabela com as informações das evidências da avaliação

```
<p:dataTable id="dtEvidencias" var="evi" value="#{conjuntoEvidenciasEditMB.lRespostaEvidencias}">
    <f:facet name="header">#{messages['evidenciaEntity.label']}</f:facet>
    <p:column headerText="#{messages['evidenciaEntity.label.nome']}">
        <p:commandLink id="clEvidencia"
            actionListener="#{conjuntoEvidenciasEditMB.carregarEvidencia(evi.evidencia)}"
            update=":formModalDtEvidencia:modalDtEvidencia" oncomplete="dialogDtEvidencia.show()">
            <h:outputText id="evidenciasNome" value="#{evi.evidencia.nome}"/>
        </p:commandLink>
    </p:column>
    <p:column headerText="#{messages['respostaEvidencia.label.abrangencia']}">
        <h:outputText id="evidenciasAbrangencia" value="#{evi.abrangencia}"/>
    </p:column>
    <p:column headerText="#{messages['respostaEvidencia.label.contribuicao']}">
        <h:outputText id="evidenciasContribuicao" value="#{evi.contribuicao}"/>
    </p:column>
    <p:column headerText="#{messages['respostaEvidencia.label.motivo']}">
        <h:outputText id="evidenciasMotivo" value="#{evi.motivo}"/>
    </p:column>
    <p:column headerText="#{messages['respostaEvidencia.label.profissionais']}">
        <p:commandButton value="#{messages['button.new.list']}"
            actionListener="#{conjuntoEvidenciasEditMB.novoProfissional(evi)}"
            update=":formModalProfissionais:panelProfissionais"
            oncomplete="dialogProfissionais.show()"
            rendered="#{securityContext.hasRole('AVALIADO') || securityContext.hasRole('ADM')}"/>
        <p:commandButton value="#{messages['button.list']}"
            actionListener="#{conjuntoEvidenciasEditMB.carregaProfissionais(evi)}"
            update=":formModalDtProfissionais:dtListEvidenciaProfissionais"
            oncomplete="dialogDtProfissionais.show()"
            rendered="#{securityContext.hasRole('AVALIADOR') || securityContext.hasRole('ADM')}"/>
    </p:column>
    <p:column headerText="#{messages['respostaEvidencia.label.acoes']}"
        rendered="#{securityContext.hasRole('AVALIADO') || securityContext.hasRole('ADM')}">
        <p:commandButton value="#{messages['button.edit']}"
            actionListener="#{conjuntoEvidenciasEditMB.editarEvidencia(evi)}"
            update=":formModalEditarEvidencia:panelEditarRegistroEvidencia"
            oncomplete="dialogEditarEvidencia.show()"/>
        <p:commandButton value="#{messages['button.delete']}"
            actionListener="#{conjuntoEvidenciasEditMB.excluirEvidencia(evi)}"
            update=":formRegistroEvidencias:dtEvidencias"/>
    </p:column>
</p:dataTable>
```

Será permitido para o avaliador o registro da pontuação e o comentário das evidências vinculadas. Na Figura 20 é possível visualizar os componentes responsáveis pelo registro de pontuação e comentário.

Figura 20 – Registro de pontuação e comentário pelo avaliador

```
1  <h:panelGrid  id="panelPrincipalRegistroEvidenciaAvaliador" columns="2"
2  rendered="#{conjuntoEvidenciasEditMB.mostraConjuntoEvidenciaAvaliador()}">
3      <h:outputLabel value="#{messages['conjuntoEvidencias.label.pontuacao']}: "
       styleClass="text-input" />
4      <p:selectOneMenu id="pontuacao" effect="fade"
5      value="#{conjuntoEvidenciasEditMB.bean.conjuntoEvidenciasAux.pontuacao}">
6          <f:selectItems value="#{conjuntoEvidenciasEditMB.pontuacao}" />
7      </p:selectOneMenu>
8
9      <h:outputLabel value="#{messages['conjuntoEvidencias.label.comentario']}: "
       for="conjuntoEvidenciasComentario" styleClass="text-input" />
10     <p:inputTextarea id="conjuntoEvidenciasComentario"
11     value="#{conjuntoEvidenciasEditMB.bean.conjuntoEvidenciasAux.comentario}"
       title="#{messages['conjuntoEvidencias.alt.comentario']}" rows="10" cols="80"
       maxlength="8000" autoResize="false"/>
12     <p:message for="conjuntoEvidenciasComentario" />
13
14     <p:commandButton value="#{messages['button.save']}"
15     actionListener="#{conjuntoEvidenciasEditMB.insertConjuntoEvidencia()}"/>
16 </h:panelGrid>
```

No componente `p:commandButton` é realizada a chamada do método `insertConjuntoEvidencia` da classe `ConjuntoEvidenciasEditMB.java`. Esse método será responsável por inserir as informações de pontuação e comentário.

Além do método citado anteriormente existe também o método `atualizaPontuacaoAvaliacao`, sendo que nesses dois métodos podem ser notadas as chamadas dos métodos `update` e `findByAvaliacaoID` das classes `ConjuntoEvidenciasBC.java` e `ConjuntoEvidenciasDAO.java`. A classe `ConjuntoEvidenciasBC.java` é anotada por `@BusinessController` que é responsável pela organização da camada de negócios da ferramenta e a classe `ConjuntoEvidenciasDAO.java` que é anotada por `@PersistenceController` que é responsável pela organização da camada de persistência.

A ação dos métodos anteriormente descritos tem por finalidade registrar o comentário sobre as evidências e gerar a pontuação dos resultados esperados da avaliação. Após o registro da pontuação do resultado esperado é realizada a ação de calcular a pontuação geral da avaliação. Este cálculo acontece sempre após o registro de pontuação do resultado esperado e vai gerando a pontuação da avaliação até que sejam registradas todas as pontuações dos resultados esperados, tendo consequentemente um resultado final da avaliação. Na Figura 21 é possível visualizar os métodos `insertConjuntoEvidencia` e `atualizaPontuacaoAvaliacao`.

Figura 21 – Métodos `insertConjuntoEvidencia` e `atualizaPontuacaoAvaliacao`

```
1  public void insertConjuntoEvidencia(){
2      this.conjuntoEvidenciasBC.update(this.getBean().getConjuntoEvidenciasAux());
3      atualizaPontuacaoAvaliacao();
4  }
5
6  private void atualizaPontuacaoAvaliacao() {
7      int qtdRespostas = 0, completamenteAtendido = 0, largamenteAtendido = 0,
       parcialmenteAtendido = 0, naoAtendido = 0, naoRespondida = 0;
8      for (ConjuntoEvidenciasEntity conjuntoEvidenciasEntity :
       conjuntoEvidenciasDAO.findByAvaliacaoID(getId())) {
9          qtdRespostas++;
10         if(conjuntoEvidenciasEntity.getPontuacao() != null){
11             switch (conjuntoEvidenciasEntity.getPontuacao()) {
12             case F:
13                 completamenteAtendido++;
14                 break;
15             case L:
16                 largamenteAtendido++;
17                 break;
18             case P: parcialmenteAtendido++;
19                 break;
20             case N: naoAtendido++;
21                 break;
22             }
23         }else{
24             naoRespondida++;
25         }
26     }
27
28     if(naoRespondida == 0 && (parcialmenteAtendido > 0 || naoAtendido > 0)){
29         getBean().setPontuacao(EnumPontuacaoAvaliacao.REPROVADA);
30     }else if(naoRespondida == 0 && qtdRespostas == (completamenteAtendido+
       largamenteAtendido)){
31         getBean().setPontuacao(EnumPontuacaoAvaliacao.APROVADA);
32     }else{
33         getBean().setPontuacao(EnumPontuacaoAvaliacao.PENDENTE);
34     }
35     update();
36 }
```

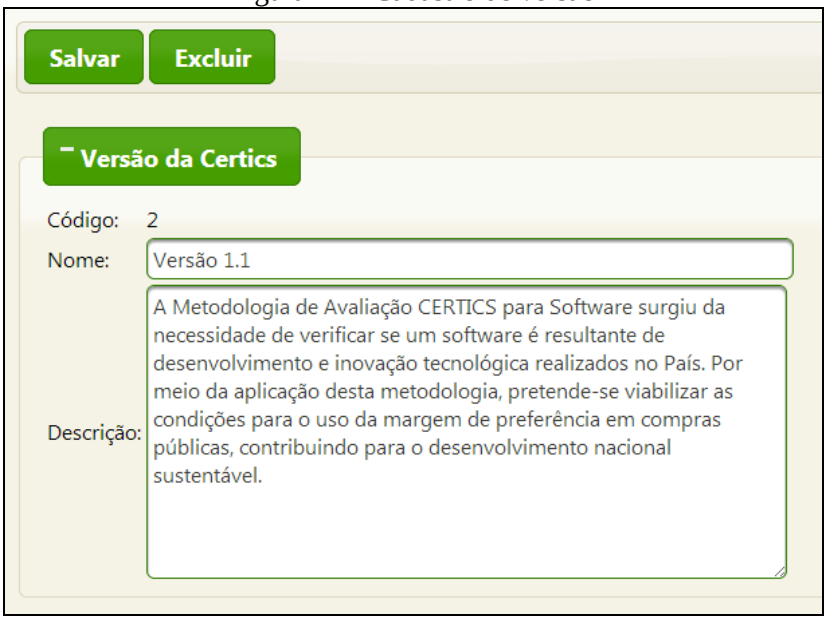
Durante o processo de avaliação o avaliador e o profissional poderão emitir relatórios para verificar quais os resultados esperados que não foram preenchidos e para visualizar as evidências vinculadas. Para isso foi criada a classe `ExportarRelatorio.java`, ela será reponsável por emitir os relatórios. Nessa classe foi criado o método `exportarRelatorioPdf` que será o responsável por executar a exportação do relatório utilizando a biblioteca JasperReports.

3.3.3 Operacionalidade da implementação

Esta seção apresenta a operacionalidade da ferramenta desenvolvida. Para realizar a demonstração da ferramenta foi simulada uma avaliação de um software fictício de determinada organização solicitante fictícia para que se pudesse demostrar os resultados que a ferramenta disponibiliza.

Para iniciar a utilização da ferramenta o administrador deve criar uma versão da CERTICS. Esta versão será utilizada para vincular as áreas de competência e os resultados esperados em uma avaliação. Na Figura 22 podem ser observadas as informações necessárias para se cadastrar uma versão.

Figura 22 – Cadastro de versão



Em seguida o administrador deve cadastrar as áreas de competência e os resultados esperados das áreas de competência registradas. Para se realizar o cadastro dessas duas informações recomenda-se utilizar como base o que está definido na metodologia da CERTICS. Para os dois cadastros devem ser vinculadas as versões que corresponderão em uma avaliação. Cada item pode ter mais de uma versão vinculada. As Figuras 23 e 24 que demonstram os cadastros das áreas de competência e resultados esperados.

Figura 23 – Cadastro de área de competência

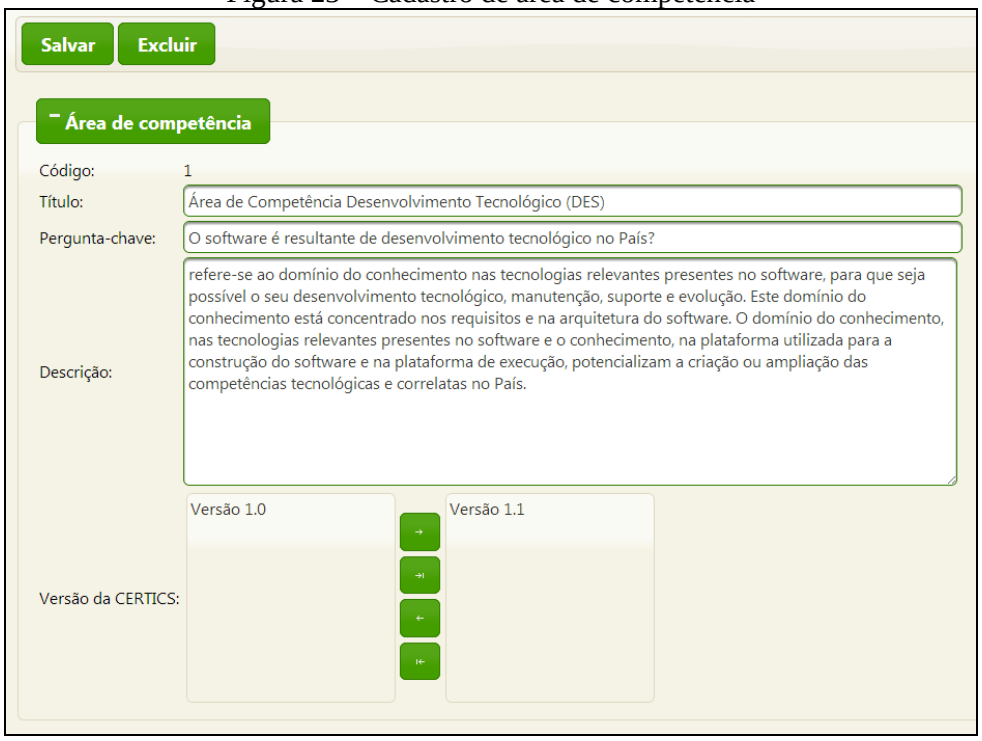
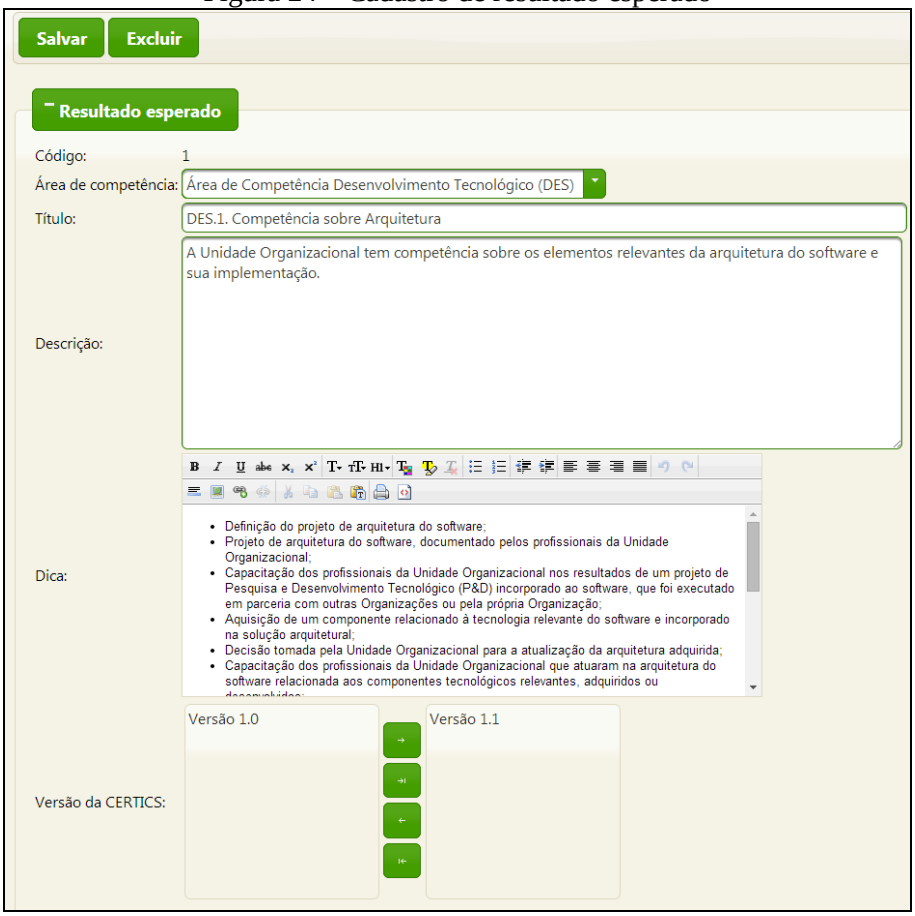


Figura 24 – Cadastro de resultado esperado



Com a versão, áreas de competências e resultados esperados definidos, o avaliador poderá criar uma avaliação. Para isso ele deve cadastrar uma organização solicitante, um

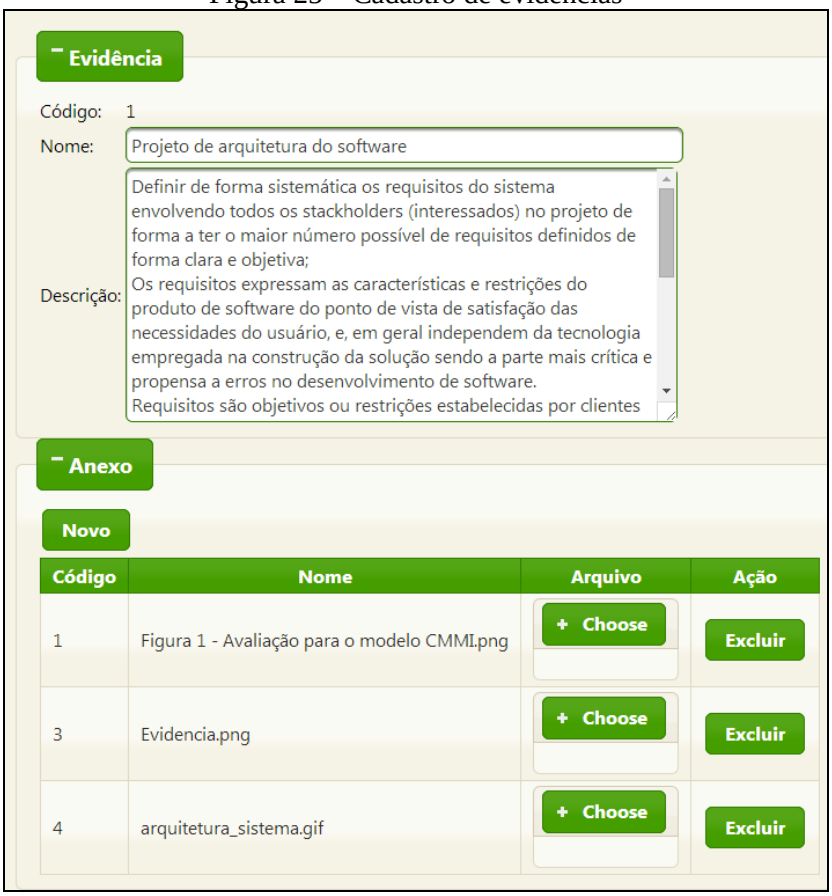
profissional responsável pela organização e um software dessa organização e, por fim, registrar uma avaliação. Para registrar uma avaliação ele deve escolher qual software será avaliado, a versão e a data de início da avaliação. Após o avaliador executar a ação de cadastrar a avaliação, a ferramenta vai montar a estrutura da avaliação com as áreas de competência e resultados esperados da versão informada pelo avaliador.

Após o término do processo de cadastro da avaliação o avaliador deve comunicar o profissional responsável pela organização solicitante de que está disponível a avaliação para ele vincular as evidências.

O profissional então vai cadastrar os profissionais que fazem parte da organização e que estão envolvidos com o software avaliado. Em seguida ele deve começar a registrar as evidências que serão vinculadas à avaliação.

No cadastro das evidências o profissional deve descrever detalhamente o que é a evidência e anexar arquivos que comprovem que a evidência atende aos critérios do resultado esperado. A Figura 25 mostra como está estruturada a tela de cadastro de evidências.

Figura 25 – Cadastro de evidências



Com as evidências cadastradas o profissional deve iniciar o vínculo delas na avaliação do software que está em aberto. Na tela de vínculo de evidências, o profissional vai visualizar

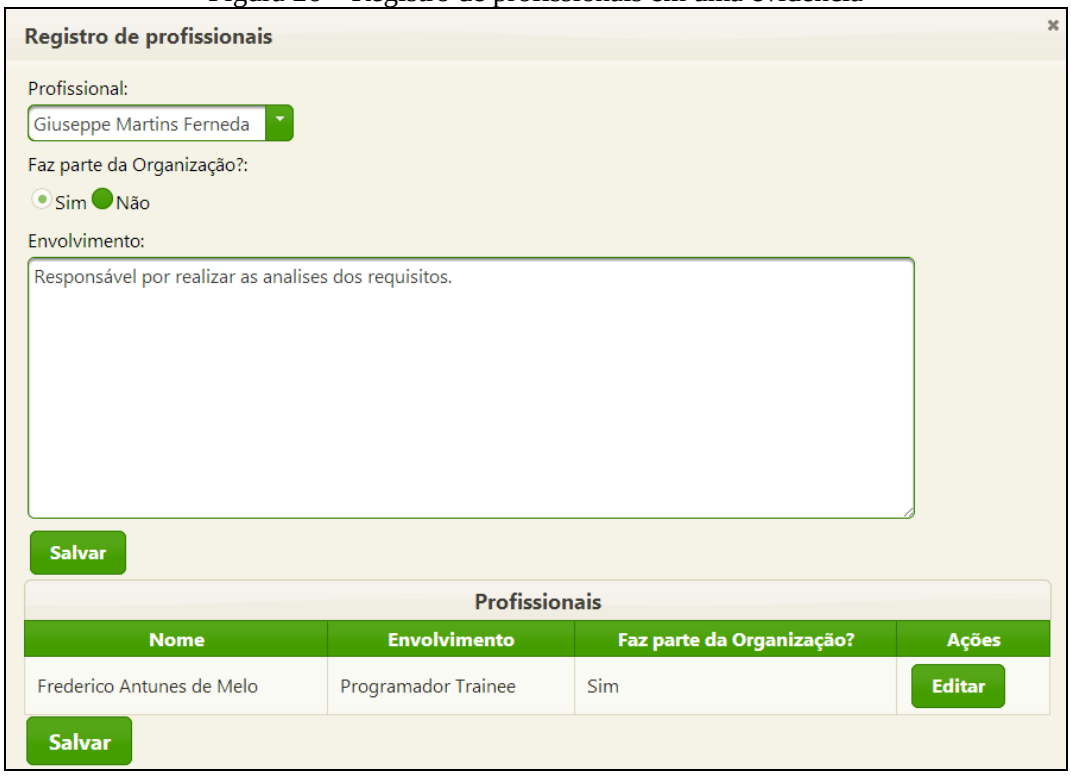
um cabeçalho com as informações da avaliação como: quem é o avaliador, o nome da organização solicitante, o nome do software e a pontuação da avaliação.

Abaixo do cabeçalho ele vai poder visualizar uma árvore com as informações das áreas de competência e seus respectivos resultados esperados. O profissional deve selecionar uma área de competência para que então ele possa selecionar um resultado esperado e realizar o vínculo das evidências para o item selecionado.

Após a seleção de um resultado esperado será disponibilizado à direita da árvore a tabela de evidências daquele resultado esperado. O profissional terá a sua disposição três botões com as seguintes opções: `Sim`, `Não` e `Não sei, preciso de ajuda`. Se ele escolher a opção `Sim`, então ele deve escolher qual a evidência que vai ser vinculada, descrever a abrangência e a contribuição dessa evidência. Se ele escolher a opção `Não`, deverá registrar o motivo pelo qual o software não possui evidências desse resultado esperado. E por último ele pode escolher a opção de `Não sei, preciso de ajuda`, então será aberta uma janela com dicas sobre o que ele pode vincular como evidência para atender aquele resultado esperado.

Com o vínculo das evidências a tabela está preenchida e então o avaliador pode registrar os profissionais nas evidências vinculadas. Para registrá-los deve ser escolhida a opção de novos profissionais na evidência correspondente e uma janela será aberta para que ele possa registrá-los. Na Figura 26 é mostrada a janela de registro de profissionais.

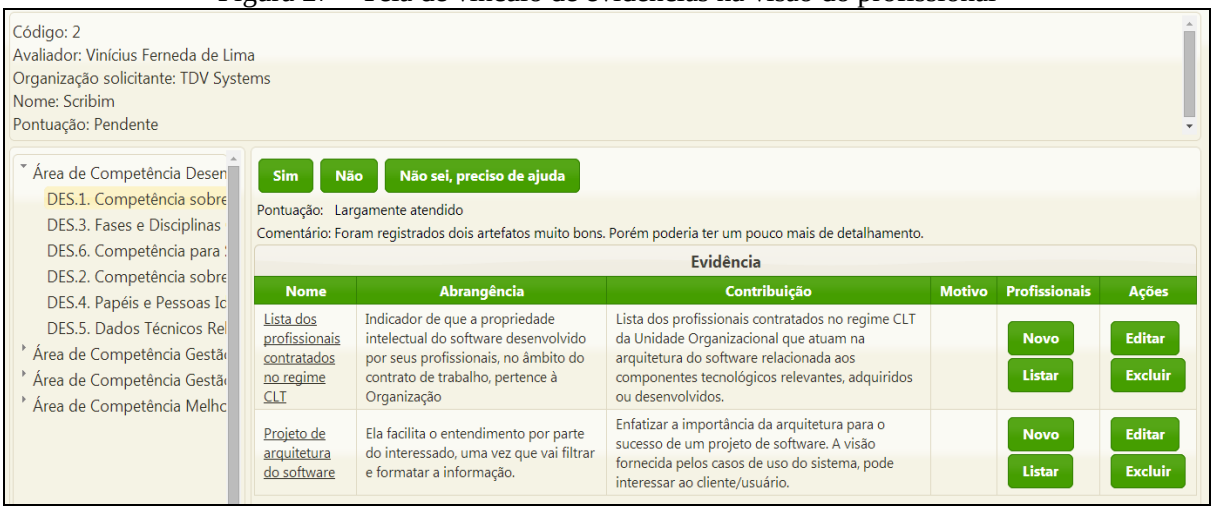
Figura 26 – Registro de profissionais em uma evidência



Deve ser selecionado o profissional a ser registrado na evidência, identificar se ele ainda faz parte da organização e descrever qual foi o envolvimento que ele possuía com aquele software. Após o registro das informações o profissional é salvo em uma tabela logo abaixo das informações e em seguida o profissional responsável que está vinculando às evidências deve salvar todos os profissionais que estão listados na tabela.

Após todos os vínculos de evidências terem sido realizados, o profissional deve notificar o avaliador do software para que ele possa realizar a avaliação das evidências vinculadas. Na Figura 27 é mostrada a tela de vínculo de evidências na visão do profissional.

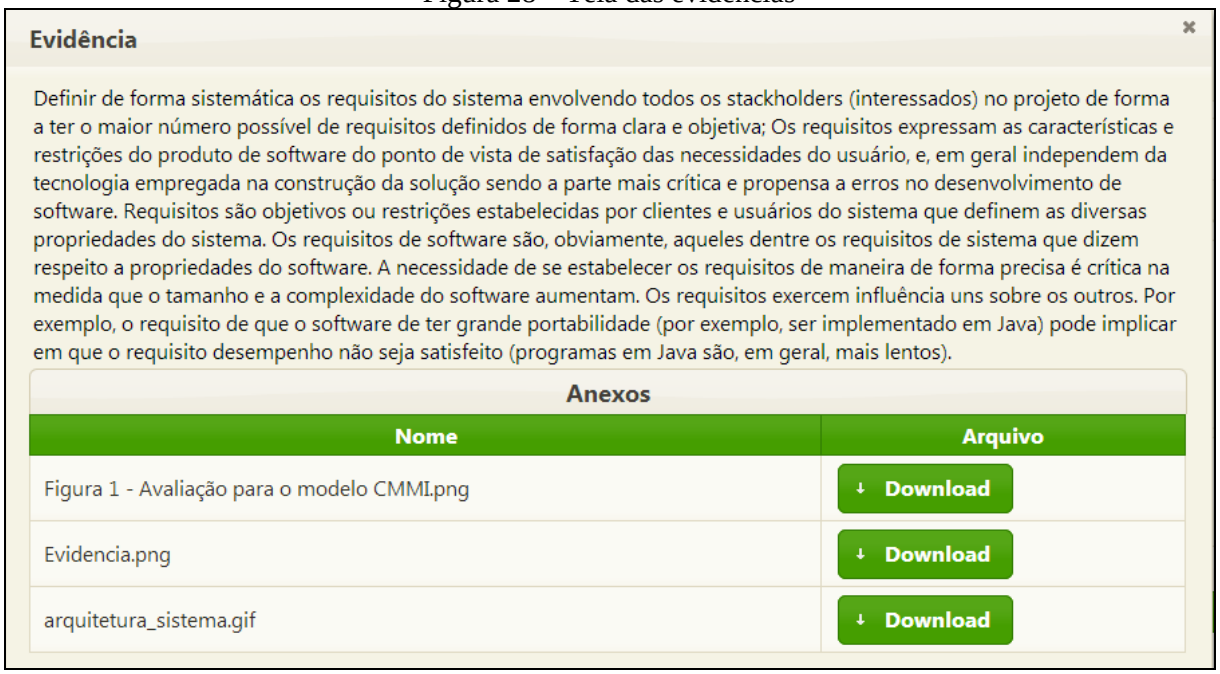
Figura 27 – Tela de vínculo de evidências na visão do profissional



Com o término do vínculo das evidências por parte do profissional da organização solicitante o avaliador deve iniciar a sua avaliação das evidências. O avaliador conseguirá visualizar todas as evidências vinculadas dos resultados esperados.

Ao selecionar um resultado esperado é disponibilizada para ele a tabela das evidências, podendo inclusive visualizar as descrições das evidências. Ele ainda conta com a opção de visualizar os anexos e os profissionais vinculados às evidências. Na Figura 28 é apresentada a tela com a descrição da evidência e os anexos disponíveis para *download* do avaliador.

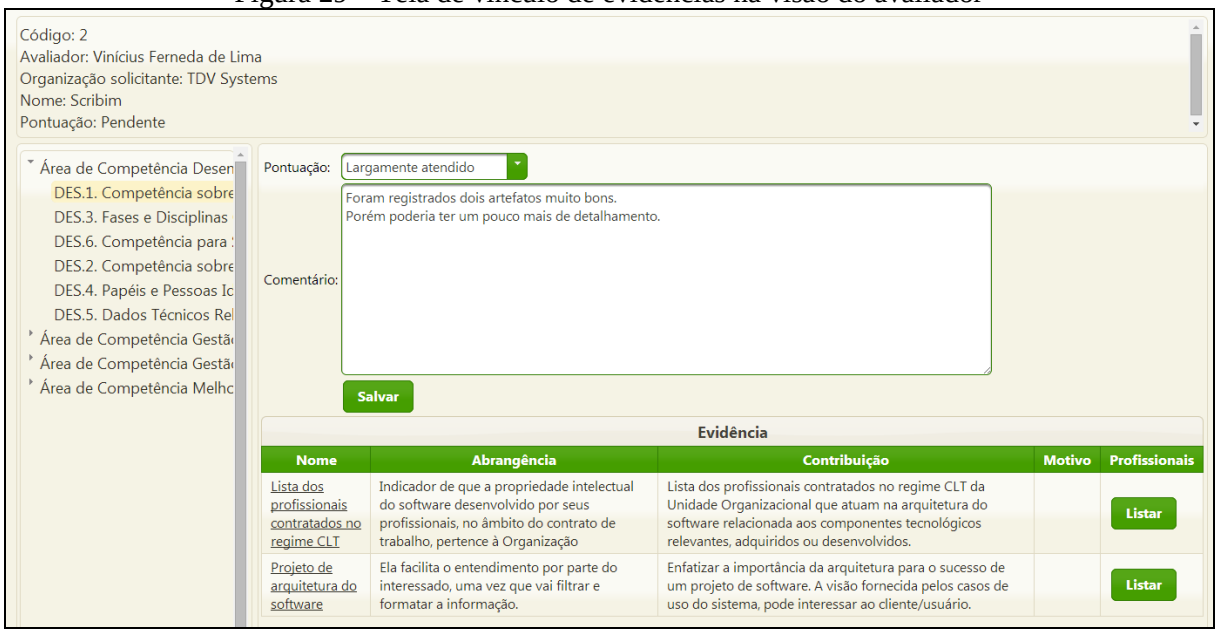
Figura 28 – Tela das evidências



Depois de avaliar todas as evidências vinculadas ao resultado esperado o avaliador deve definir uma pontuação e se achar necessário, realizar um comentário sobre as evidências vinculadas. Para cada resultado esperado avaliado a ferramenta realiza o cálculo da pontuação da avaliação. Após a avaliação de todos os resultados esperados ela disponibiliza o resultado final da avaliação.

Caso tenha algum resultado esperado que não foi aprovado por não ter as evidências corretas, o avaliador deve notificar o profissional que deve corrigir os itens pendentes para que se possa realizar novamente a avaliação destes itens e finalizar a avaliação. Na Figura 29 é possível visualizar a tela de vínculo de evidências na visão do avaliador.

Figura 29 – Tela de vínculo de evidências na visão do avaliador



Na tela de seleção de avaliações para o vínculo de evidências estarão disponíveis dois relatórios. Um relatório é para verificar quais são os itens pendentes e quantas evidências já foram vinculadas. Outro relatório exibirá detalhadamente as informações vinculadas nos resultados esperados das avaliações. Nas Figuras 30 e 31 podem ser visualizados os relatórios exportados pela ferramenta.

Figura 30 – Gráfico de atendimento das áreas de competência

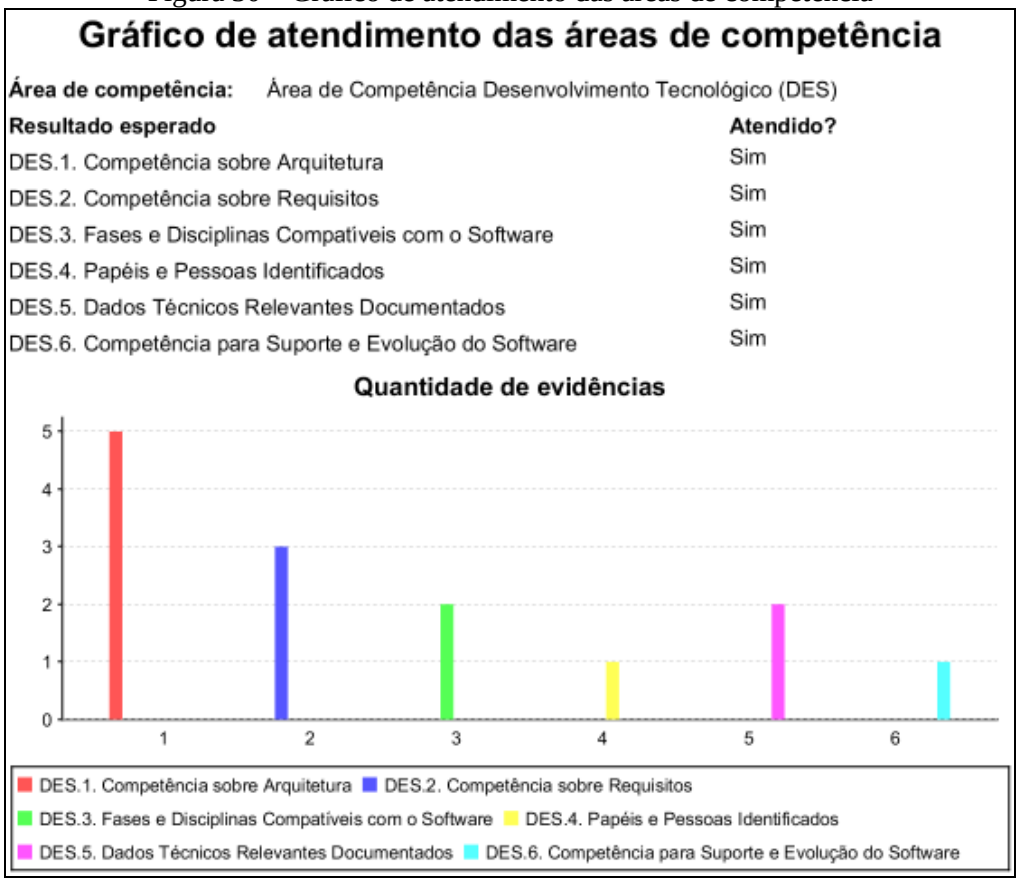


Figura 31 – Relatório de evidências

Relatório de evidências

Avaliador:	Vinicius Ferneda de Lima	**Versão:**	Versão 1.1
Organização Solicitante:	TDV Systems	**Data:**	03/06/2014
Software:	Scribim	**Pontuação:**	Pendente
Área de competência:	Área de Competência Desenvolvimento Tecnológico (DES)		
Resultado esperado:	DES.1. Competência sobre Arquitetura		
Pontuação:	Largamente atendido		
Comentário:	Foram registrados dois artefatos muito bons.		

Evidência	Descrição	Abrangência	Contribuição	Motivo
Projeto de arquitetura do software	Definir de forma sistemática os requisitos do sistema envolvendo todos os stackholders (interessados) no projeto de forma a ter o maior número possível de requisitos definidos de forma clara e objetiva; Os requisitos expressam as características e restrições do produto de software do ponto de vista de satisfação das necessidades do usuário, e, em geral independem da tecnologia empregada na construção da solução sendo a parte mais crítica e propensa a erros no desenvolvimento de software. Requisitos são objetivos ou restrições estabelecidas por clientes e usuários do sistema que definem as diversas propriedades do sistema.	Ela facilita o entendimento por parte do interessado, uma vez que vai filtrar e formatar a informação.	Enfatizar a importância da arquitetura para o sucesso de um projeto de software. A visão fornecida pelos casos de uso do sistema, pode interessar ao cliente/usuário.	

Anexo

Evidencia.png

arquitetura_sistema.gif

Profissional	Envolvimento	Vínculo	Faz parte da Organização
Frederico Antunes de Melo	Programador Trainee	Empregado	Sim

Após todo o processo de avaliação do software estar concluído o profissional da organização solicitante pode realizar um *feedback* da avaliação através do registro de lições aprendidas. Nesta funcionalidade o profissional pode informar quais foram os pontos positivos, negativos e o que ainda precisa ser melhorado no software para que ele possa atender melhor a metodologia CERTICS.

3.4 RESULTADOS E DISCUSSÃO

A ferramenta criada se mostrou flexível, pois permite a criação de versões da metodologia CERTICS. Isto é importante, pois é provável a evolução da própria metodologia.

Para o avaliador criar uma avaliação, ele deve cadastrar a organização solicitante, o software e definir qual vai ser o profissional responsável pela organização. Após esses passos ele deve escolher a versão que deseja aplicar à avaliação, realizar o registro da avaliação e notificar a organização solicitante da aplicação da avaliação.

O profissional vai poder acessar a ferramenta através da web para então registrar as evidências do software na avaliação. Para isso ele deve cadastrar os profissionais e as evidências que serão vinculados à avaliação. No tocante às evidências, ele pode cadastrar anexos para que fique mais fácil de compreender e facilite a avaliação do avaliador. Após esse processo ele deve somente vincular as evidências e registrar os profissionais aos devidos resultados esperados da avaliação e esperar pela avaliação do avaliador.

O avaliador poderá então visualizar todas as evidências vinculadas, seus anexos e os profissionais vinculados em uma tela simples e funcional o que torna a sua avaliação muito mais prática. Caso queira realizar a avaliação sem precisar acessar a web, ele pode gerar um relatório da avaliação com todas as evidências vinculadas aos resultados esperados.

Concluída a avaliação ele deve atribuir uma pontuação a cada resultado esperado e então a ferramenta realiza o cálculo da pontuação final da avaliação. O avaliador pode comunicar à organização solicitante que, a avaliação do software foi realizada e os resultados e comentários estarão disponíveis para visualização na ferramenta.

Com relação aos trabalhos correlatos, os mesmos apresentam soluções para a avaliação de softwares ou organizações através de ferramentas em ambientes *desktops* ou web utilizando como referência às metodologias baseadas na área de qualidade de software. Estas ferramentas foram estudadas para verificar o desenvolvimento da ferramenta de avaliação. A ferramenta proposta por Albuquerque Júnior, Santos e Furtado (2008) tem por objetivo realizar o processo de avaliação com base na metologia do modelo MPS.BR. O trabalho de Borges (2006) teve como objetivo auxiliar no processo de avaliação de produtos de software utilizando como estrutura de avaliação a norma NBR ISO/IEC 14598-5 e o método de avaliação da qualidade de produto de software MEDE-PROS. A ferramenta CERTICSys proposta pelo CTI Renato Archer (2013d) é a única das ferramentas estudadas que segue o modelo de avaliação proposta pela CERTICS. O Quadro 4 apresenta as principais diferenças entre os trabalhos correlatos e o trabalho desenvolvido.

Quadro 4 – Comparativo entre os trabalhos

FUNCIONALIDADES	**TCC**	**Albuquerque Júnior**	**Borges**	**CERTICSys**
Metodologia CERTICS	X			X
Ambiente web	X		X	X
Evidências com anexo	X	X	X	X
Relatórios gerenciais	X	X	X	X
Portal aberto para pessoas físicas	X	X	X	
Tutor automatizado que auxilia na identificação das evidências				X

4 CONCLUSÕES

Após a análise da metodologia CERTICS, verificou-se que a quantidade de regras que a avaliação possui, justifica a criação de uma ferramenta para o auxílio dos avaliadores neste processo, diminuindo assim o tempo e os custos da avaliação. Sem este tipo de ferramenta além da avaliação ser mais demorada, pode gerar muitas vezes uma decisão incorreta e com isso não gerar o resultado que a organização pretendia.

Este trabalho apresentou o desenvolvimento da ferramenta que teve como principal objetivo o auxílio do processo de avaliação de um software utilizando a metodologia CERTICS, proporcionando uma maior facilidade ao avaliador para que as evidências sejam devidamente cadastradas e avaliadas corretamente dentro das regras apresentadas pela metodologia. Com a ferramenta evita-se que o avaliador tenha que fazer uso de outros softwares como editores de texto e planilhas de cálculo. A ferramenta tende a diminuir o tempo gasto pelo avaliador, porém ele não será o único beneficiado, pois a organização avaliada também ganhará com uma melhor visão das inconformidades e ajustes que ela precisa realizar para que seu software seja adequado à metodologia CERTICS.

A utilização do *framework Demoiselle* foi um grande facilitador no desenvolvimento da ferramenta. Ele demonstrou ser fácil de configurar, com uma grande diversidade de materiais e intuitivo para o desenvolvimento de projetos. A utilização do *framework* Primefaces também foi um grande facilitador na parte visual da ferramenta. É disponibilizado um ambiente em que se pode estudar quais os melhores componentes que serão usados no desenvolvimento das telas. Feito para trabalhar com JSF, se torna totalmente compatível com o *framework Demoiselle*. Na parte de geração de relatórios a biblioteca JasperResports facilitou a configuração e criação dos relatórios.

Em relação aos trabalhos correlatos estudados, pode-se afirmar que a ferramenta proposta diferencia-se em alguns aspectos das descritas anteriormente. Todas as ferramentas apresentam soluções de avaliação com linguagens de programação e modos de acessos diferentes da ferramenta proposta neste trabalho. O principal aspecto a ser apresentado é que a ferramenta utilizará a metodologia CERTICS para a realização da avaliação do software, sendo que das três ferramentas estudadas para a realização deste trabalho somente a ferramenta CERTICSys proposta pelo CTI Renato Archer (2013d) utiliza a metodologia CERTICS, sendo que o principal aspecto que diferencia a ferramenta CERTICSys para este trabalho é a oportunidade que ele proporciona ao professor de qualidade de software um ensino mais dinâmico da metologia.

4.1 EXTENSÕES

Como sugestões de extensões para a continuidade do presente trabalho têm-se:

a) construir um recurso que gere as dicas dos resultados esperados com base nas evidências já cadastradas na ferramenta;
b) criar um forúm que possibilite o avaliador e a organização solicitante se comunicar dentro da ferramenta;
c) permitir que o avaliador crie *templates* de avaliação para o registro de evidências;
d) disponibilizar para o profissional avaliado *templates* com modelos que ele possa seguir para o cadastro das evidências;
e) disponibilizar um mecanismo que realize uma avaliação prévia automaticamente das evidências vinculadas;
f) criar um mecanismo de controle para impedir que o avaliador e o profissional façam alterações ao mesmo tempo sobre a mesma avaliação;
g) criar um mecanismo que contabilize quantas vezes o avaliador já reprovou uma avaliação;
h) possibilitar criar uma versão a partir de outra já existente.

REFERÊNCIAS

ALBUQUERQUE JÚNIOR, Carlos A. C. de; SANTOS, João A. F. F.; FURTADO, Julio C. C. **WISE:** uma abordagem de ferramenta de software para o auxílio no modelo de avaliação do MPS.BR. 2008. 65 f. Trabalho de conclusão de curso (Bacharelado em Ciência da Computação) - Centro de Ciências Exatas e de Tecnologia, Universidade da Amazônia, Belém.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR ISO/IEC 15504-2:2003:** tecnologia da informação - avaliação de processo parte 2: realização de uma avaliação. Rio de Janeiro, 2008.

BORGES, Jonathan M. **Ambiente web de suporte ao processo de avaliação da qualidade de produtos de software**. 2006. 134 f. Trabalho de conclusão de curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

CTI RENATO ARCHER. **Modelo de referência para avaliação da CERTICS**: documento de detalhamento: documento de definição, versão 1.1. Campinas, 2013a.

_____. **Metodologia de avaliação da CERTICS para software**, versão 1.1. Campinas, 2013b.

_____. **Método de avaliação da CERTICS**: documento de detalhamento, versão 1.1. Campinas, 2013c.

_____. **Manual de uso CERTICSys**, versão 1.0. Campinas, 2013d.

KOSCIANSKI, André; SOARES, Michel dos S. **Qualidade de software:** aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software. São Paulo: Novatec, 2006.

APÊNDICE A – Descrição dos principais casos de uso

Neste apêndice são apresentados em detalhes os dois principais casos de uso presentes no diagrama da Figura 4, sendo: UC10 - Criar avaliação e UC11 - Vincular evidências.

O caso de uso UC10, descrito em detalhes no Quadro 5, representa a criação de uma avaliação de um software de uma organização solicitante.

Quadro 5 – Descrição do UC10

UC10 - Criar avaliação	
Descrição	Permite criar uma avaliação de um software de uma organização solicitante com base em uma versão definida pelo avaliador
Pré condição	O usuário logado deve possuir o papel de avaliador.
Cenário principal	1) O avaliador escolhe o software; 2) O avaliador escolhe a versão; 3) O avaliador escolhe a data de início da avaliação; 4) O avaliador solicita o registro da avaliação; 5) A ferramenta busca as áreas de competência e os resultados esperados da versão informada pelo avaliador; 6) A ferramenta registra a avaliação com as informações recuperadas no passo anterior; 7) A ferramenta retorna mensagem para o usuário informando que a avaliação foi registrada com sucesso.
Exceção 01	No passo 3 se o avaliador escolher uma data anterior a data atual é gerada uma mensagem informando que a data deve ser igual ou posterior a data atual.
Exceção 02	No passo 5 se não existir áreas de competência ou resultados esperados da versão informada pelo avaliador, a ferramenta retorna uma mensagem para o avaliador informar outra versão.
Pós condição	Foi criada uma avaliação para o software informado e a avaliação se torna disponível para a organização solicitante vincular as evidências.

O caso de uso UC11, descrito em detalhes no Quadro 6, representa o processo de vínculo das evidências em uma avaliação.

Quadro 6 – Descrição do UC11

UC11 – Vincular evidências	
Descrição	Permite ao profissional de uma organização solicitante vincular as evidências do software que está sendo avaliado
Pré condição	O usuário logado deve possuir o papel de profissional e pertencer à organização solicitante da avaliação do software.
Cenário principal	1) O profissional deve selecionar qual o software que ele deseja vincular as evidências; 2) A ferramenta apresenta os resultados esperados; 3) O profissional deve vincular as evidências para todos os resultados esperados das áreas de competências vinculadas à avaliação; 4) A ferramenta registra a evidência ao resultado esperado; 5) A ferramenta lista as evidências vinculadas; 6) O profissional deve registrar os profissionais da organização solicitante envolvidos na evidência; 7) A ferramenta registra o profissional a evidência; 8) A ferramenta lista os profissionais envolvidos na evidência.
Exceção	No passo 2 caso o profissional não preencher as informações solicitadas, a ferramenta apresenta mensagem de erro.
Exceção	No passo 5 caso o profissional não preencher as informações solicitadas, a ferramenta apresenta mensagem de erro.

O caso de uso UC12, descrito em detalhes no Quadro 7, representa o processo de cálculo da pontuação da avaliação e registro de observações em uma avaliação.

Quadro 7 – Descrição do UC12

UC12 – Registrar avaliação	
Descrição	Permite ao avaliador avaliar as evidências vinculadas na avaliação, registrar uma pontuação e observações.
Pré condição	O usuário logado deve possuir o papel de avaliador e ser o responsável pela avaliação do software.
Cenário principal	1) O avaliador deve selecionar qual o software que ele deseja avaliar as evidências; 2) A ferramenta apresenta os resultados esperados; 3) O avaliador deve avaliar e atribuir uma pontuação para cada resultado esperado; 4) A ferramenta realiza o cálculo da pontuação da avaliação do software.
Pós condição	É disponibilizado um relatório com os resultados finais da avaliação do software.

APÊNDICE B – Dicionário de dados do MER

Neste apêndice é apresentado o dicionário de dados referente ao modelo entidade relacionamento mostrado no Quadro 8.

Quadro 8 – Dicionário de dados

Tabela: TB_ANEXO				
KEY	**CAMPO**	**DESCRIÇÃO**	**TIPO**	**TAM.**
PK	ANE_ID	Identificador	BIGINT	
FK	ANE_EVIID	Chave de ligação com a evidência	BIGINT	
	ANE_ARQUIVO	Arquivo	BLOB	
	ANE_NOME	Nome	VARCHAR	255
Tabela: TB_AREA_COMPETENCIA				
PK	ARC_ID	Identificador	BIGINT	
	ARC_TITULO	Título	VARCHAR	255
	ARC_PERGUNTA_CHAVE	Pergunta-chave	VARCHAR	255
	ARC_DESCRICAO	Descrição	VARCHAR	8000
Tabela: TB_AVALIACAO				
PK	AVA_ID	Identificador da avaliação	BIGINT	
FK	AVA_SOFID	Chave de ligação com o *software*	BIGINT	
FK	AVA_VCEID	Chave de ligação com a versão	BIGINT	
FK	AVA_AVRID	Chave de ligação com o avaliador	BIGINT	
	AVA_PONTUACAO	Pontuação	VARCHAR	50
	AVA_DATA	Data de início	DATE	
Tabela: TB_CONJUNTO_EVIDENCIA				
PK	CEV_ID	Identificador do conjunto de evidências	BIGINT	
FK	CEV_RESID	Chave de ligação com o resultado esperado	BIGINT	
FK	CEV_AVAID	Chave de ligação com a avaliação	BIGINT	
	CEV_COMENTARIO	Comentário do artefato	VARCHAR	8000
	CEV_PONTUACAO	Pontuação do artefato	VARCHAR	255
Tabela: TB_ENDERECO				
PK	END_ID	Identificador	BIGINT	
	END_CEP	CEP	VARCHAR	20
	END_LOGRADOURO	Logradouro (Rua, Avenida, etc.)	VARCHAR	255
	END_NUMERO	Número	INTEGER	
	END_COMPLEMENTO	Complemento	VARCHAR	255
	END_BAIRRO	Bairro	VARCHAR	255
	END_UF	Estado	VARCHAR	50
	END_CIDADE	Cidade	VARCHAR	255
	END_PAIS	País	VARCHAR	255
Tabela: TB_EVIDENCIA				
PK	EVI_ID	Identificador	BIGINT	
	EVI_DESCRICAO	Descrição	VARCHAR	8000
	EVI_NOME	Nome	VARCHAR	255
Tabela: TB_EVIDENCIA_PROFISSIONAL				
PK	EPR_ID	Identificador	BIGINT	
FK	EPR_REVID	Chave de ligação com a evidência	BIGINT	
FK	EPR_PROID	Chave de ligação com o	BIGINT	

		profissional		
	EPR_ENVOLVIMENTO	Envolvimento	VARCHAR	1000
	EPR_FAZ_PART_ORG	Faz parte da organização	INTEGER	
Tabela: TB_LICAO_APRENDIDA				
PK	LIA_ID	Identificador	BIGINT	
FK	LIA_AVAID	Chave de ligação com a avaliação	BIGINT	
	LIA_MELHORIA	Melhorias	VARCHAR	8000
	LIA_PONTO_POSITIVOS	Pontos positivos	VARCHAR	8000
	LIA_PONTOS_NEGATIVOS	Pontos negativos	VARCHAR	8000
Tabela: TB_ORGANIZACAO_SOLICITANTE				
PK	ORS_ID	Identificador	BIGINT	
FK	ORS_ENDID	Chave de ligação com o endereço	BIGINT	
	ORS_NOME	Nome	VARCHAR	255
	ORS_CNPJ	CNPJ	VARCHAR	20
	ORS_FONE_1	Telefone principal	VARCHAR	20
	ORS_FONE_2	Telefone alternativo	VARCHAR	20
Tabela: TB_PESSOA_FISICA				
PK	PES_ID	Identificador	BIGINT	
FK	PES_ORSID	Chave de ligação com a organização solicitante	BIGINT	
FK	PES_USUID	Chave de ligação com o usuário	BIGINT	
FK	PES_ENDID	Chave de ligação com o endereço	BIGINT	
	PES_NOME	Nome	VARCHAR	255
	PES_CPF	CPF	VARCHAR	20
	PES_RG	RG	VARCHAR	20
	PES_SEXO	Sexo	VARCHAR	20
	PES_DT_NASCIMENTO	Data de nascimento	DATE	
	PES_FONE_1	Telefone principal	VARCHAR	20
	PES_FONE_2	Telefone alternativo	VARCHAR	20
	PES_RESORG	Responsável pela organização	INTEGER	
	PES_VINCULO_ATUAL	Vínculo atual	VARCHAR	255
Tabela: TB_RESPOSTA_EVIDENCIA				
PK	REV_ID	Identificador	BIGINT	
FK	REV_EVIID	Chave de ligação com a evidência	BIGINT	
FK	REV_CEVID	Chave de ligação com o artefato	BIGINT	
	REV_ABRANGENCIA	Abrangência	VARCHAR	8000
	REV_CONTRIBUICAO	Contribuição	VARCHAR	8000
	REV_MOTIVO	Motivo	VARCHAR	8000
Tabela: TB_RESULTADO_ESPERADO				
PK	RES_ID	Identificador	BIGINT	
FK	RES_ARCID	Chave de ligação com a área de competência	BIGINT	
	RES_TITULO	Título	VARCHAR	255
	RES_DESCRICAO	Descrição	VARCHAR	8000
	RES_DICA	Dica	VARCHAR	8000
Tabela: TB_SOFTWARE				
PK	SOF_ID	Identificador	BIGINT	
FK	SOF_ORSID	Chave de ligação com a organização solicitante	BIGINT	
	SOF_NOME	Nome	VARCHAR	255
	SOF_ASPECTO_INOVADOR	Aspecto inovador	VARCHAR	8000
	SOF_DATA_INICIO	Data de início	DATE	
	SOF_DATA_LIBERACAO	Data de liberação	DATE	

	SOF_DESCRICAO	Descrição	VARCHAR	8000
	SOF_HISTORICO	Histórico	VARCHAR	8000
	SOF_VERSAO	Versão	VARCHAR	10
	SOF_TECNOLOGIAS	Tecnologias	VARCHAR	8000
Tabela: TB_USUARIO				
PK	USU_ID	Identificador	BIGINT	
	USU_EMAIL	Email	VARCHAR	50
	USU_SENHA	Senha	VARCHAR	50
	USU_ATIVO	Ativo	INTEGER	
	USU_PAPEL	Papel	VARCHAR	50
Tabela: TB_VERSAO_CERTICS				
PK	VCE_ID	Identificador	BIGINT	
	VCE_DESCRICAO	Descrição	VARCHAR	8000
	VCE_NOME	Nome	VARCHAR	255
Tabela: TB_VERSAO_CERTICS_AREA_COMPETENCIA				
PK	VAC_ID	Identificador	BIGINT	
FK	VRE_ARCID	Chave de ligação com a área de competência	BIGINT	
FK	VRE_VCEID	Chave de ligação com a versão	BIGINT	
Tabela: TB_VERSAO_CERTICS_RESULTADO_ESPERADO				
PK	VRE_ID	Identificador	BIGINT	
FK	VRE_VCEID	Chave de ligação com a versão	BIGINT	
FK	VRE_RESID	Chave de ligação com o resultado esperado	BIGINT	