Integração baseada em Serviços de Aplicações Web Legadas

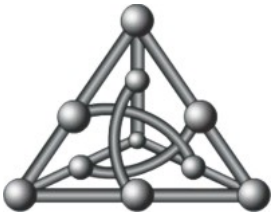
Maxwell Sampaio dos Santos

Dissertação de Mestrado

Orientação:Prof. Dr. Marcelo Augusto Santos Turine

Área de Concentração: Engenharia de Software

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, Curso de Pós-Graduação em Ciência da Computação, Faculdade de Computação da Fundação Universidade Federal de Mato Grosso do Sul.



Faculdade de Computação
Universidade Federal de Mato Grosso do Sul
Novembro/2011

A minha amada e abençoada família, especialmente
a minha mãe, Francisca Bernardino Sampaio,
e à memória de meu pai, Luiz Barbosa dos Santos.

Agradecimentos

À Deus por me dar forças, me guiar e me iluminar em todos os momentos da minha vida.

À minha amada família por seu apoio e presença, tanto física quanto espiritual, em todos os momentos. Especialmente, a minha mãe Francisca e ao meu falecido pai Luiz, que sempre foram o exemplo a ser seguido, e a minha irmã Daiane pelas lições de vida de todos os dias.

Aos meus familiares que me apoiaram e sempre acreditaram em mim.

Ao meu orientador professor Marcelo Turine, pela dedicação e confiança durante realização deste trabalho, sem as quais ele não seria realidade.

Aos meus amigos Camilo, Márcio, Lucas, Ronaldo, Kishi, companheiros de tantos projetos e atividades.

Aos meus primos e amigos, sempre presentes desde a minha infância e nos momentos em que mais precisei.

A todos os integrantes e ex-membros do projeto Web-PIDE tanto da UFMS quanto da UFSCar pela dedicação e apoio na realização deste trabalho e pelas amizades que fiz.

Ao INEP pelo fomento do projeto Web-PIDE e pela disponibilização dos dados gerados por meio dos diversos exames de avaliação educacional aplicados desde 1995.

Aos amigos do curso de Ciência da Computação com os quais criei fortes laços de amizade.

A todos os membros do Laboratório de Engenharia de Software (LEDES) pelas amizades que lá fiz.

A todos os professores e funcionários da Faculdade de Computação (FACOM) da Universidade Federal de Mato Grosso do Sul (UFMS), que me ajudaram sempre que precisei.

À CAPES e ao INEP pelo apoio financeiro para realização do mestrado.

Resumo

A constante evolução tecnológica é um fator que contribui para a fragmentação de sistemas em qualquer organização, resultando na falta de integração entre seus sistemas existentes, geralmente denominados sistemas legados, pois são importantes sistemas para a organização e normalmente foram desenvolvidos e mantidos por diferentes equipes de desenvolvimento. Em especial, na área educacional é comum encontrar aplicações legadas em algumas instituições de ensino que poderiam ser utilizadas de forma integrada a outras aplicações com o propósito de formar uma solução computacional mais robusta para um determinado problema. Neste contexto, a integração de sistemas legados é importante para a evolução destes sistemas pois aumenta a acessibilidade deles não permitindo que eles se tornem tão obsoletos com o passar do tempo. Na literatura existem alguns trabalhos que propõem desde abordagens para integração até ambientes computacionais de apoio à integração destes sistemas legados. Entretanto, poucos são os trabalhos que definem uma abordagem detalhada para integração de sistemas legados do tipo *web*, sendo este o foco desta pesquisa. Com isso, o objetivo deste trabalho é definir um processo para integração de aplicações web legadas, intitulado PIBSAWL, e um ambiente integrador de aplicações, intitulado AIA. O PIBSAWL foi definido com base no processo de desenvolvimento orientado a serviço bottom-up visando especificar um processo genérico de apoio à integração de aplicações legadas. Já o AIA foi construído utilizando a combinação das tecnologias Web Service e LDAP para compor um ambiente genérico que pode ser utilizado para (1) integrar aplicações *web* legadas de acesso restrito, permitindo que um usuário logado no AIA obtenha acesso autenticado em uma aplicação integrada a ele e (2) registrar aplicações web legadas de acesso público e aplicações *desktop* legadas, permitindo, respectivamente, o acesso à aplicação *web* por meio de um *browser* e o *download* do instalador da aplicação *desktop*. Por fim, dois estudos de caso são apresentados para avaliar o processo PIBSAWL e o ambiente AIA propostos neste trabalho.

Palavras-chave: processo, integração, aplicações web, web service, ldap, bottom-up.

Abstract

The constant evolution of technology is one factor that contributes to the fragmentation of systems in any organization, resulting in the lack of integration between yours systems, generally known as legacy systems, beacause they are important systems for organization and they were usually developed and maintained by different development teams. In particular, in the educational area is common to find legacy applications in some educational institutions that could be used in an integrated way to other applications with the purpose to create a more robust solution for a specific computational problem. In this context, the integration of legacy systems is fundamental to the evolution of these systems because it increases the accessibility of them not allowing them to become as obsolete with the passage of time. In the literature there are some works wich propose since approaches for integration until computing environments to support the integration of these systems. However, there are few works that define a detailed approach for integration of legacy systems like web and this is the focus of this research. Thus, the purpose of this research is to define a process for integration of legacy web applications, entitled PIBSAWL, and an environment for integration of applications, entitled AIA. The PIBSAWL was defined based on the service-oriented development process bottom-up which is also presented in this research. In relation the AIA, it was built using a combination of LDAP and Web Service technologies to compose a generic environment that can be used (1) to integrate legacy web applications with restricted access, allowing that a user logged in AIA gets authenticated access in an integrated application in the AIA and (2) to register legacy web applications of public access and legacy desktop applications, allowing, access via a browser int the web application and the download of desktop application's intaller, respectively. Finally, using the Web-PIDE website like environment for integration of applications, two case studies are presented to evaluate the PIBSAWL process and AIA environment, proposed in this research.

Palavras-chave: process, integration, web applications, web service, ldap, bottom-up.

Conteúdo

Lista de Siglas

AJAX *Asynchronous Javascript and XML*

CAPES Coordenação de Aperfeiçoamento de Pessoal de Nível Superior

CEB Censo Escolar da Educação Básica

CMS *Content Management Systems*

CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico

DEAR *Data Extractor ASCII to Relational*

DNS *Domain Name System*

DW *Data Warehouse*

EAD Ensino à Distância

ENC Exame Nacional de Cursos

ENEM Exame Nacional do Ensino Médio

ENADE Exame Nacional de Desempenho de Estudantes

HTML *Hypertext Transfer Language*

HTTP *Hypertext Transfer Procotol*

HTTPS *Hypertext Transfer Procotol Secure*

INEP Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira

JSON *JavaScript Object Notation*

LDAP *Lightweight Directory Access Protocol*

LIDE Linguagem para Integração de Dados Educacionais

MEC Ministério da Educação

OLAP *On-Line Analytical Processing*

PDF *Portable Document Format*

PHP *Hypertext Preprocessor*

REST *Representational State Transfer*

SAEB Sistema Nacional de Avaliação da Educação Básica

SAFE *Software Engineering Available for Everyone*

SAS *Statistical Analysis Software*

SGBDRO Sistema de Gerenciamento de Bancos de Dados Objeto-Relacionais

SIGFAP Sistema de Informação e Gestão de Projetos das Fundações de Amparo à Pesquisa

SOA *Service-Oriented Architecture*

SOAP *Simple Object Access Protocol*

SQL *Structured Query Language*

SSL *Secure Socket Layer*

TLS *Transport Layer Security*

UDDI *Universal Description Discovery and Integration*

UFMS Universidade Federal de Mato Grosso do Sul

UFSCar Universidade Federal de São Carlos

UML *Unified Modeling Language*

XML *Extensible Markup Language*

Web-PIDE Plataforma Aberta para Integração e Avaliação de Dados Educacionais na Web

WS *Web Service*

WSDL *Web Service Description Language*

WSAS *Web Services Based Authentication System*

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Contextualização

Sistemas legados são sistemas antigos que possuem uma função crítica em relação ao processo funcional de uma organização. Importantes regras corporativas podem estar inseridas neles e podem não estar documentadas em nenhum outro lugar [17]. Desta forma, uma simples substituição do sistema pode ter consequências imprevisíveis.

De acordo com Cummins [17], manter sistemas legados envolve um grande dispêndio de dinheiro e tempo, pois, (1) não há um estilo de programação consistente em todo o sistema já que partes dele, geralmente, foram desenvolvidas por diferentes equipes; (2) uma parte ou todo o sistema pode ter sido implementado em uma linguagem obsoleta o que dificulta encontrar pessoal capacitado para manter o sistema; (3) frequentemente, a documentação é inadequada e desatualizada. Em alguns casos, a única documentação é o código-fonte. Em outros, tem-se apenas o executável do sistema; e (4) por falta de acessibilidade e manutenibilidade, estes sistemas acabam se tornando inutilizáveis pois com o avanço da tecnologia e o surgimento de outros sistemas concorrentes não é possível manter estes sistemas atrativos no cenário atual.

A constante evolução tecnológica é um fator que contribui para a fragmentação de sistemas em qualquer organização, fazendo com que as mesmas se deparem com falta de integração entre seus sistemas legados [17]. Em especial, na área educacional é comum encontrar aplicações legadas em diversas instituições de ensino que poderiam ser utilizadas de forma integrada a outras aplicações com o propósito de formar uma solução computacional mais robusta para um determinado problema. Como exemplo, uma instituição de ensino poderia realizar uma integração entre seus sistemas legados tanto *web* quanto *desktop* e fornecer, por exemplo, serviços *web* de acesso único e exclusivo aos seus alunos, professores e funcionários, sem que cada um deles tenha que realizar novas requisições de acesso a cada um dos sistemas legados.

Neste contexto, a integração de sistemas legados é fundamental para a evolução destes sistemas e na literatura já existem alguns trabalhos relacionados, como por exemplo, (1) Sarmento [61], em seu trabalho, propõe a integração de um ambiente virtual de aprendi-

zagem com três aplicações educacionais móveis com o intuito de ampliar os mecanismos de notificação e comunicação aos usuários presentes nos atuais ambientes web de aprendizagem; (2) Cunha *et al* [18] propõem uma arquitetura orientada a serviço para integração dos sistemas de informação do Centro Federal de Educação Tecnológica de Alagoas (CEFET-AL), com o intuito de obter uma operação mais eficaz dos processos de negócio da instituição e diminuir a inconsistência e replicação de dados; (3) Alkouz e El-Seoud [1] propõem um sistema de autenticação baseado em web services para plataforma de ensino à distância utilizando tecnologias Web Services e LDAP para fornecer um modo de acesso centralizado e único às plataformas de EaD; e (4) Zhang *et al* [82] propõem uma estratégia caixa-preta para migração de sistemas legados baseados em interface gráfica com a utilização de web services tornando tais sistemas acessíveis em uma rede distribuída através de *Virtual Network Computing* (VNC).

Entretanto, poucos foram os trabalhos encontrados que descrevem todo um processo voltado para a integração de aplicações legadas [82] [29] e isto é uma das motivações da presente proposta, ressaltando apenas a integração de aplicações web legadas. Outro motivador é o projeto Web-PIDE que produziu alguns sistemas para resolver problemas específicos, com isto surgiu a necessidade de integrá-los para aumentar ainda mais a acessibilidade e utilidade deles.

Com isso, o objetivo do presente trabalho é definir um processo genérico para a integração baseada em serviços de aplicações web legadas denominado PIBSAWL. Para apoiar o uso deste processo, foi construído um ambiente integrador de aplicações, intitulado AIA, que integra o controle de acesso às aplicações web legadas que forem integradas utilizando o processo proposto neste trabalho. Assim, um usuário autenticado no AIA realiza acesso autenticado nas aplicações web legadas sem necessidade de informar novamente seus dados de acesso. Para a construção do AIA foi utilizado a combinação das tecnologias Web Service e LDAP para fornecer tal funcionalidade de acesso.

Dois estudos de caso são apresentados para validar a proposta como um processo genérico, um com a integraçãdo da ferramenta Pentaho/Web-PIDE ao Portal Web-PIDE e outro com a integração do SIGFAP ao Portal Web-PIDE. Ambos estudos de caso utilizaram a portal Web-PIDE como AIA, que recebeu um evolução em sua arquitetura para integrar os dois sistemas legados dos estudos de caso.

1.2 Motivação e Justificativa

O INEP estruturou o Sistema Nacional de Avaliação e Informação (SINAIS), dessa forma, vários censos e pesquisas são realizados e bases de dados produzidas. No entanto, estes dados que dão subsídios ao levantamento de informações educacionais estão distribuídos em diferentes bases de dados, de maneira não uniformizada e de difícil reuso e integração como, por exemplo, os sistemas de avaliação institucional das universidades brasileiras. Este fato dificulta substancialmente a tomada de decisão na gestão pública.

Com isso, a CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) [11] e o INEP iniciaram em 2007 o Programa Observatório da Educação [50], uma

iniciativa para fomentar o desenvolvimento de estudos e pesquisas em educação, com a finalidade de estimular a produção acadêmica e a formação de recursos pós-graduados, em nível de mestrado e doutorado, em áreas voltadas à pesquisa da educação, por meio de financiamento específico, para consolidar e ampliar o pensamento crítico estratégico para o desenvolvimento sustentável do País.

Neste contexto, em janeiro de 2007 foi aprovado pelo INEP/CAPES, o projeto de pesquisa intitulado "Web-PIDE - Uma Plataforma aberta para Integração e avaliação de Dados Educacionais na Web" [72] da UFMS (Universidade Federal de Mato Grosso do Sul) em parceria com a UFSCar (Universidade Federal de São Carlos) como parte do Programa Observatório de Educação. O projeto tem como objetivo produzir um sistema computacional para integrar e disponibilizar os dados educacionais do INEP, facilitando a consulta aos dados das avaliações educacionais (SAEB, ENEM, ENADE e outras) gerando hipóteses a serem investigadas.

Entretanto, o grande volume de dados destas avaliações precisam de padronização, o que motivou a geração de *Data Warehouses* (DW) para algumas bases do INEP (SAEB e CEB) padronizando o grande volume de dados destas bases. Em seguida, a equipe do projeto iniciou o desenvolvimento de diversas aplicações Web de avaliação educacional para atuarem sobre os DW. Porém, nem todas as aplicações foram desenvolvidas e integradas na plataforma Web-PIDE, dificultando a consulta aos dados do INEP e tomadas de decisão. Dessa forma, as aplicações que foram desenvolvidas precisam ser integradas na plataforma Web-PIDE e disponibilizadas como serviços.

Foi constatado também que o INEP e algumas instituições de ensino, como a UFMS por exemplo, possuem diversos sistemas de informação, considerados sistemas legados, e nem todos eles encontram-se integrados. Com a criação de um processo para integração destes sistemas e um ambiente integrador de aplicações seria possível aumentar a acessibilidade destes sistemas tornando-os ainda mais difundidos na comunidade geral, que muitas vezes nem sabe que tais sistemas existem.

Outro fator motivador deste trabalho é que foram encontrados na literatura poucos trabalhos [82] [29] que definem uma abordagem detalhada apoiada por um ambiente computacional integrador para realizar a integração de aplicações legadas. Já outros trabalhos [61] [1] [5] [18], apenas definem um ambiente computacional de apoio à integração, mas sem especificarem um processo, que utilize tal ambiente, para a integração de novas aplicações legadas. Com isto, novas aplicações que forem integradas a estes ambientes não possuem o apoio de um processo facilitador da integração, tornando esta tarefa de integração um pouco mais complexa.

1.3 Objetivos

O objetivo geral do presente trabalho é definir um processo para integração baseada em serviços de aplicações web legadas a um ambiente integrador de aplicações, que também é proposto neste trabalho.

Para tanto, os seguintes objetivos específicos são necessários:

- Realizar levantamento bibliográfico quanto à (1) abordagens para integração de aplicações legadas e (2) ambientes computacionais que apoiem a integração de aplicações legadas;
- Analisar os processos de desenvolvimento de serviços *Top-down*, *Bottom-up* e *Meet-in-the-middle*, mais utilizados na literatura, para embasar o processo de integração proposto;
- Definir o processo para integração baseada em serviços de aplicações web legadas à um ambiente integrador de aplicações a partir da adaptação dos modelos avaliados anteriormente;
- Definir, especificar e implementar um ambiente integrador de aplicações usando uma arquitetura orientada a serviço para apoiar a integração de aplicações legadas e a execução do processo definido anteriomente; e
- Validar o processo proposto utilizando como estudo de caso uma aplicação desenvolvida pelos membros do projeto Web-PIDE e uma aplicação desenvolvida pelo grupo de Engenharia de Software do LEDES/UFMS.

1.4 Organização do Texto

O presente texto está dividido em seis capítulos e um apêndice. No capítulo atual foram apresentados o contexto do problema a ser investigado, as motivações e os objetivos do presente trabalho.

O Capítulo 2 compreende os três principais processos de desenvolvimento de software baseado em serviços, *Top-down*, *Bottom-up* e *Meet-in-the-middle*. Em seguida, no Capítulo 3 é apresentado a proposta de um processo para integração baseada em serviços de aplicações web legadas (PIBSAWL), além da construção de um ambiente integrador de aplicações (AIA) para apioar o uso desse processo.

No Capítulo 4 são apresentados dois estudos de caso de integração de aplicações legadas utilizando o processo PIBSAWL e o AIA: Pentaho/Web-PIDE e SIGFAP, onde são indicados quais passos das atividades do processo PIBSAWL foram executados e como foram executados para realizar a integração destas aplicações.

No Capítulo 5 são apresentados alguns trabalhos relacionados, sob duas perpectivas, (1) abordagens para integração de aplicações legadas e (2) ambientes computacionais de apoio à integração de aplicações legadas.

Por fim, no Capítulo 6 são apresentadas as contribuições e limitações deste trabalho, além de possíveis trabalhos futuros e no Apêndice A é apresentado uma visão geral do projeto Web-PIDE juntamente com a avaliação e definição de qual ferramenta desenvolvida pela equipe do projeto Web-PIDE foi escolhida e utilizada como estudo de caso do presente trabalho.

Capítulo 2

Desenvolvimento Orientado a Serviços

2.1 Considerações Iniciais

Como um dos focos principais do trabalho é a integração baseada em serviços de aplicações web legadas, neste capítulo é apresentada a definição e exemplificação do que é um serviço. Em seguida, são fornecidas definições importantes sobre o paradigma orientado a serviço, o que é um Web Service e os componentes que o compõe. Por fim, são apresentadas três abordagens de processo de desenvolvimento de serviços: *top-down*, *bottom-up* e *meet-in-the-middle*. Foram escolhidas estas abordagens por serem as mais utilizadas pelas instituições públicas e privadas no que diz respeito ao processo de migração para o desenvolvimento orientado a serviço.

2.2 Orientação a Serviço

Para Vissers e Logrippo [77] e Quartel et al [59] uma definição genérica de um serviço que faz sentido tanto nas empresas quanto no domínio de Tecnologia da Informação (TI), é "um serviço é um ambiente observável de um sistema (provedor de serviço) em termos de interações que podem ocorrer nas interfaces entre o sistema (solicitante de serviço), o ambiente e as intercomunicações entre essas interações." No caso, o termo sistema, usado anteriormente, refere-se tanto a aplicações como unidades organizacionais.

Stojanovic e Dahanayake [67] reforçam ainda que o conceito de serviço é o resultado da separação de ambientes internos e externos de um sistema. Portanto, um serviço precisa ser independente e ter um objetivo claro na perspectiva de seus ambientes. O ambiente externo, em outras palavras, representa o que é requisitado para realizar esse serviço. Para os consumidores de um serviço, o ambiente interno de um sistema ou organização é geralmente irrelevante: eles estão somente interessados na funcionalidade e qualidade do que será fornecido. Por exemplo, quando um cliente contrata o serviço de internet ADSL

de uma companhia telefônica, ele não está interessado em como funciona a estrutura física e lógica do serviço oferecido, nem se este serviço necessita de outros serviços para serem entregues, mas sim na qualidade do serviço final prestado (banda larga disponível, quedas e travamentos da conexão ADSL, suporte técnico, etc).

Segundo Engels et al [23] não existe um acordo na literatura do que é um serviço. Reussner e Hasselbring [60] definem um serviço como: "Implementação de uma lógica de negócio que pode ser acessada por meio de interfaces padronizadas." Enquanto Muller et al [48] definem um serviço como: "Um canal de comunicação onde suas funcionalidades podem ser acessadas por meio de sistemas de mensagens padronizados, sendo ele independente do contexto ou estado de outros serviços".

Desta forma, entende-se que um serviço é uma funcionalidade de um sistema que pode ser acessada por qualquer outro sistema, livre de plataforma, por meio de interfaces previamente padronizadas desde que exista uma canal de comunicação fortemente estabelecido entre eles e que não depende de outros sistemas (externos) ou funcionalidades (internas). Um exemplo de serviço seria o cálculo do frete de um produto em um determinado sistema cliente. Neste caso, o sistema cliente precisa informar ao sistema servidor apenas o CEP do destinatário e o peso (em Kilogramas, por exemplo) do produto a ser fretado, outras opções como: o tipo do frete (SEDEX, PAC, etc) ou se a embalagem precisa ou não precisa de cuidados especiais não são obrigatórias mas poderiam ser informadas. Com isso, o sistema servidor responde ao sistema cliente o valor do frete para cada um dos tipos de frete que ele tem disponível de acordo com o CEP informado.

De acordo com Bubeck, no contexto computacional, um serviço oferece funcionalidade reusável que é contratualmente definida em uma descrição de serviço. Sendo a descrição de serviço uma combinação de informação comportamental, semântica e sintática. Na orientação a serviço, a montagem da aplicação é baseada somente em descrições de serviço; os provedores de serviço são descobertos e integrados dentro da aplicação, durante, após ou, em geral antes, da execução da aplicação. Como resultado, a orientação a serviço foca em como os serviços são descritos de forma que suportem a pesquisa dinâmica em tempo real de serviços apropriados [10].

Uma suposição importante na orientação a serviço é que os serviços podem estar dinamicamente disponíveis, ou seja, sua disponibilidade pode variar continuamente. Dessa forma, um serviço pode estar disponível ou não sem que o sistema como um todo, fique indisponível. Isto favorece a manutenibilidade do sistema. Por exemplo, quando uma aplicação requisita, por meio de um Web Service, demais informações relacionadas a um CEP no site dos CORREIOS, ela está propícia a não obter acesso ao serviço ou não obter os resultados naquele momento, pois trata-se de um serviço e este pode estar temporariamente indisponível (ele próprio ou outros serviços que ele depende). Com isso, a aplicação precisaria realizar uma nova requisição posteriormente.

Outro exemplo que ocorre rotineiramente são os acessos à sites na Web através de um browser. Muitas vezes o usuário solicita acesso à uma página Web entretanto não obtem os resultados necessários por conta da página (ou serviço) estar temporariamente indisponível (por questões de manutenção ou problemas técnicos). Com isto, o usuário realiza novas requisições até que a página retorne os resultados satisfatórios.

Conforme ilustra a Figura 2.1, para apoiar a pesquisa dinâmica de serviços, a orientação a serviço é baseada em um padrão de interação proposto por Stojanovic e Dahanayake [67], que envolve três diferentes atores:

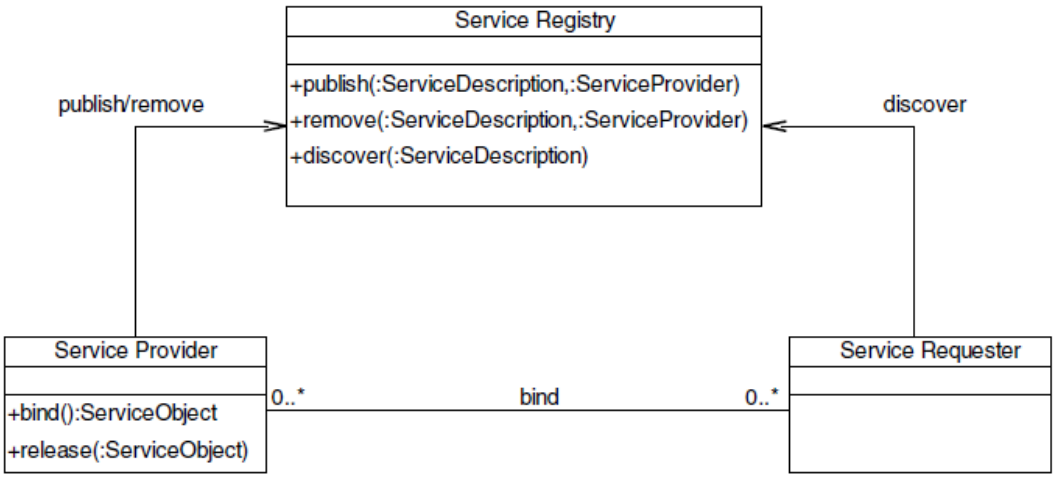


Figura 2.1: Atores do padrão de interação orientado a serviço [67].

- **Provedor de serviço**: responsável por fornecer ou remover objetos de serviço que implementam a funcionalidade do serviço por meio de um registrador de serviço.
- **Solicitante de serviço**: é o cliente de um serviço particular, sendo que ele solicita determinado serviço a um provedor de serviço diretamente ou indiretamente, por meio de um registrador de serviço.
- **Registrador de serviço**: é o intermediário entre os solicitantes e os provedores de serviço.

2.3 Padrão de Interação Orientado a Serviço

De acordo com Stojanovic e Dahanayake [67], o padrão básico da interação que caracteriza a orientação a serviço é retratada no diagrama de sequência da Figura 2.2. Este diagrama ilustra um provedor de serviço (objeto da classe `ServiceProvider`) que publica uma descrição de serviço (por meio do método `publish`) em um registrador de serviço (objeto da classe `ServiceRegistry`). Para descobrir serviços, um solicitante de serviço (objeto da classe `ServiceRequester`) pergunta (por meio do método `discover`) ao registrador de serviço baseado em um conjunto de critérios relacionados a descrição do serviço. Se os serviços que cumprem os critérios foram previamente publicados nos provedores de serviço, o registrador de serviço seleciona (por meio do método `filterProviders`) os provedores de serviço baseado nos critérios de descrição do serviço que ele recebeu e retorna as referências dos provedores ao solicitante do serviço. Neste caso, onde multiplas referências de provedores são retornadas, o solicitante de serviço pode selecionar (por meio

do método `filterResults`) um provedor de serviço específico para conectar-se (por meio do método `bind`) a ele.

Quando o solicitante de serviço conecta ao provedor (por meio do método `create`), um objeto de serviço é criado (objeto da classe `ServiceObject`) e retornado pelo provedor. Neste momento, o solicitante interage (por meio do método `interactWithService`) com o objeto de serviço retornado pelo provedor. Finalmente, quando o solicitante de serviço encerra a interação com o serviço, o objeto de serviço é liberado (por meio do método `release`) implicitamente ou explicitamente. Além disso, um provedor de serviço pode ainda remover (por meio do método `remove`) uma descrição de serviço em um registrador de serviço.

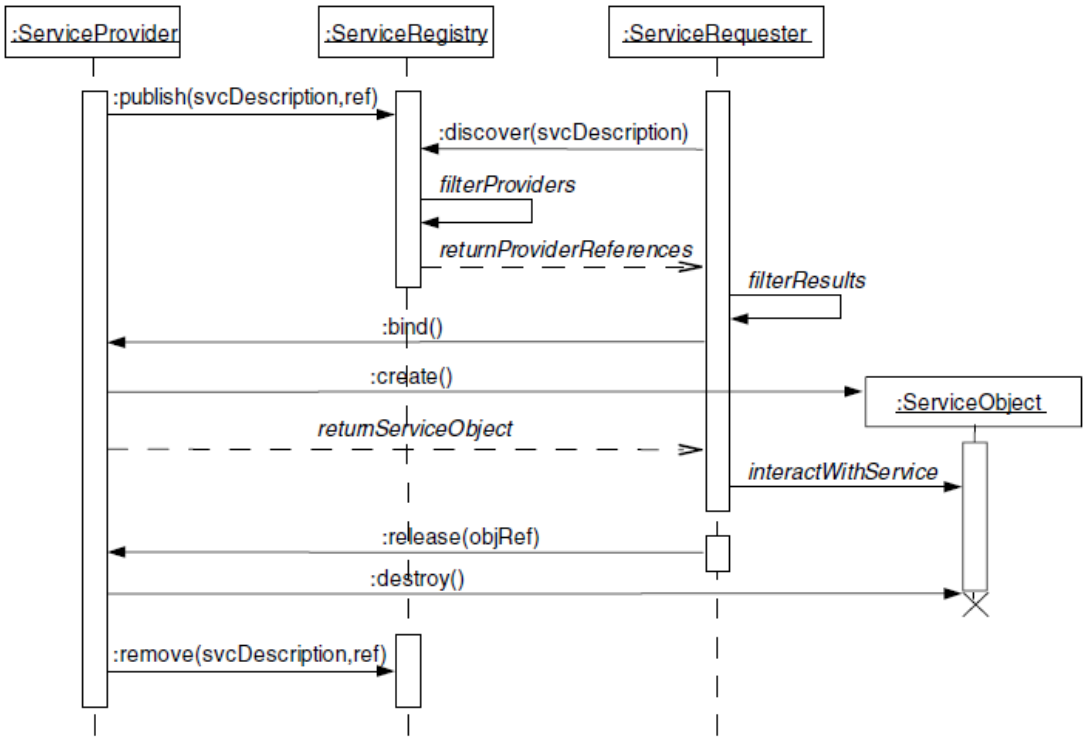


Figura 2.2: Diagrama de sequência do padrão de interação orientado a serviço [67].

Um registrador de serviço contém um conjunto de descrições de serviço junto com as referências para os provedores de serviço. Além disso, fornece mecanismos para a publicação, remoção e pesquisa de serviços. Este elemento não precisa necessariamente estar envolvido no processo de interação orientado a serviço, já que um solicitante de serviço pode conectar-se diretamente a um provedor de serviço, sem a necessidade de consultar um registrador de serviço, desde que ele conheça corretamente a descrição do serviço.

Descrição de serviço

A descrição de serviço combina informação sintática, semântica e comportamental do serviço. A parte sintática da descrição de serviço é tipicamente incorporada como uma

interface de serviço, que é um conjunto de operações que fornece a funcionalidade do serviço. Uma interface de serviço define um contrato sintático e também fornece um nível limitado de semântica dos nomes de suas operações; por exemplo, um método chamado print() em uma interface de serviço de impressão.

Objeto de Serviço

O objeto de serviço implementa a interface de serviço e é retornado por um provedor de serviço no momento em que um solicitante de serviço conecta ao provedor. Objetos de serviço são criados e liberados de acordo com um conjunto de políticas. Em orientação a serviço, os solicitantes de serviço não tem conhecimento sobre as políticas seguidas pelo provedor de serviço quando cria objetos de serviço durante a conexão. Existem diferentes políticas para criação de objeto de serviço:

- ***Shared object***: o provedor de serviço cria um simples objeto que é retornado para todos os solicitantes de serviço quando eles conectam ao provedor; o objeto é, consequentemente, compartilhado por todos os solicitantes.
- ***Object pool***: o provedor de serviço cria uma certa quantidade de objetos de serviço. Um objeto de serviço em uma posição diferente da fila de objetos é retornado para cada requisição que se conecta ao provedor de serviço de acordo com a disponibilidade. Uma vez que um objeto é liberado, ele é retornado para a fila para reuso posterior. Nesta situação, objetos de serviço são compartilhados pelos solicitantes, mas não concorrentemente, desde que dois solicitantes conectados ao provedor nunca obtenham uma referência do mesmo objeto simultaneamente. Esta política é útil, por exemplo, quando recursos, tais como memória, são limitados ou quando objetos de serviço representam um recurso físico que é limitado em quantidade, tais como uma porta de comunicação.
- ***One object per requester***: um objeto de serviço é criado para cada solicitante. Se o mesmo solicitante se conecta ao provedor de serviço várias vezes, o solicitante sempre obtém o mesmo objeto de serviço. Esta política é útil, por exemplo, quando o solicitante de serviço interage com um objeto de serviço remoto por meio de várias chamadas de método mas não mantém uma conexão contínua com o objeto de serviço, como por exemplo se a comunicação é feita por meio de um protocolo sem conexão, como o HTTP.
- ***One object per binding***: um objeto de serviço diferente é criado cada vez que um solicitante de serviço se conecta ao provedor de serviço. Como esta abordagem exige muito processador e memória não é tão interessante utilizá-la pois pode ocorrer um ataque de negação de serviço caso ela não seja bem utilizada.

Stojanovic e Dahanayake [67], a escolha da política de criação de objeto é importante quando provedores de serviço são *stateful*. Um serviço *stateful* é capaz de manter o estado de persistência a partir de várias chamadas de método pelo mesmo cliente. As políticas de criação fila de objetos e um objeto por solicitante são adequadas para serviços *stateful*. A

política de criação de objeto compartilhado não é uma boa política para serviços *stateful*, a menos que a intenção seja explicitamente compartilhar o estado com todos os solicitantes. A política de criação de um objeto por conexão pode ser usada por serviços *stateful*, mas isso requer que o solicitante de serviço esteja informado da situação e somente conecte ao provedor de serviço uma vez e mantenha o objeto de serviço retornado em todas as interações com o serviço. Em contraste, todas as políticas de criação são adequadas para serviços *stateless*. Um serviço *stateless* é aquele que não mantêm um estado de persistência a partir de várias chamadas de método pelos clientes do serviço.

Liberação de Serviço

Ao fim da interação entre o solicitante e o serviço, o solicitante deve liberar o objeto de serviço. Esta etapa é necessária desde que o provedor de serviço necessite saber se o objeto de serviço não está mais sendo usado para então destrui-lo ou entregá-lo a outro solicitante. Duas políticas de liberação podem existir:

- **Explícita**: quando a liberação é explícita, o solicitante explicitamente invoca algum método que informa ao provedor de serviço que sua interação com o serviço foi encerrada.
- **Implícita**: quando a liberação é implícita, o fim do uso do objeto é determinado por meio da chamada de métodos implícitos, tais como, *garbage collection* ou *lease expiration*. O conceito de *leasing* permite que um provedor de serviço libere automaticamente um objeto de serviço de um solicitante antes mesmo do solicitante renovar a posse do objeto. Esta política pode ser usada, por exemplo, em conjunto com uma política de fila de objetos para garantir que após um certo tempo os objetos de serviço sejam liberados e retornados à fila.

Composição de Serviço

Composição de serviço representa o uso de um conjunto de serviços para completar uma tarefa particular. Composição de serviço é frequentemente considerada responsabilidade dos solicitantes de serviço. Na prática, a composição de serviço é a incorporação de diferentes serviços dentro de uma aplicação para executar alguma função global, onde a aplicação representa o papel de solicitante de serviço. Tal aplicação obtém o fluxo de dados e de controle que coordena a invocação do serviço e transfere dados entre os diferentes serviços. A aplicação coordenadora pode ser escrita em uma linguagem de programação padronizada, entretanto, existe uma tendência forte para favorecer o uso de processos executáveis para escrever tais aplicações. Um exempo disto é BPEL4WS [4], que é usada no contexto de orquestração de Web services [54].

Uma composição de serviço é escrita em termos das interfaces de serviço e é considerada abstrata até o tempo de execução, quando os provedores de serviço são descobertos e conectados. Composições de serviço devem tratar assuntos relacionados ao descobrimento de serviço e trocas constantes de serviço. Com respeito ao descobrimento de serviço, estes

assuntos incluem disponibilidade do serviço, filtragem do lado do solicitante e ausência de conhecimento no que diz respeito às políticas de criação de objetos de serviço. Com respeito as trocas constantes de serviço, é possivel que um provedor de serviço particular se torne indisponível enquanto uma aplicação coordenadora estiver executando. Este problema é específico em Web services onde mecanismos de transação permitem desfazer as ações no caso de falhas de invocação de serviço.

Segundo Peltz [55], os termos orchestração e coreografia, no contexto de orientação a serviço, descrevem dois aspectos da criação de processos de negócio compostos de Web Services. Os dois termos se sobrepõem um pouco. Orquestração refere-se a processos de negócio executáveis que podem interagir tanto com Web Services internos quanto externos. As interações ocorrem a nível de mensagem. Elas incluem lógica de negócio, executam tarefas ordenadas e podem transpor aplicações e organizações para definir um modelo de processo multi-etapas, transacional e duradouro.

Peltz afirma que a orquestração representa sempre o controle do ponto de vista de uma das partes. Isto difere da coreografia que é mais colaborativa e permite que cada parte envolvida descreva sua participação na interação. Coreografia segue as sequências de mensagens entre várias partes e fontes, ao invés de um processo específico de negócio que uma simples parte executa.

Execução de Serviço

Na orientação a serviço, um ambiente de execução fornece dois mecanismos principais para que provedores e solicitantes de serviço suportem o padrão de interação orientado a serviço. O primeiro mecanismo é o acesso ao registrador de serviço, que inclui três principais operações:

- ***Publish***: usado pelos provedores de serviço para adicionar uma descrição de serviço, juntamente com uma referência ao provedor de serviço, para o registrador de serviço.
- ***Remove***: usado pelos provedores de serviço para remover uma descrição de serviço previamente publicada. Em certas tecnologias orientadas a serviço, a remoção de um provedor de serviço de um registrador de serviço requer que os objetos de serviço que o provedor criou sejam liberados pelos solicitantes.
- ***Discover***: usado pelos solicitantes de serviço para obter referências aos provedores de serviço em um registrador de serviço. Para obter um serviço, o solicitante de serviço envia critérios para o registrador que são usados para selecionar o conjunto de provedores de serviço (filtragem do lado do registrador); sendo que somente provedores de serviço que cumpram os critérios são retornados. A seleção final de um provedor de serviço específico é deixado para o solicitante, que pode precisar escolher um simples provedor de serviço quando múltiplos provedores de serviço cumprirem os critérios fornecidos (filtragem do lado do solicitante).

De acordo com Stojanovic e Dahanayake [67], para permitir que os provedores de serviço estejam cientes das mudanças nos serviços, o ambiente de execução fornece um

segundo mecanismo que são notificações para sinalizar as mudanças de serviço. Por meio de notificações, os solicitantes de serviço podem saber sobre mudanças na disponibilidade do serviço para serem capazes de incorporarem novos serviços que se tornaram disponíveis ou para encerrarem o uso de serviços que se tornaram indisponíveis. A notificação de serviços preocupa-se com os seguintes eventos:

- **Serviço publicado**: um evento que ocorre quando um serviço é publicado no registrador.
- **Serviço removido**: um evento que ocorre quando um serviço é removido do registrador.
- **Serviço modificado**: um evento que ocorre quando um serviço que está registrado é modificado sem ser removido do registrador. Uma modificação pode ocorrer, por exemplo, quando os atributos que caracterizam o serviço são modificados.

Quando o ambiente de execução não fornece mecanismos de notificação, os solicitantes de serviço podem nomear um registrador periodicamente para saber quando que os serviços são publicados ou removidos.

2.4 Web Service

Existem na literatura diversas tecnologias para implementar arquiteturas orientadas a serviço, tais como: CORBA Traders [68], JavaBeans Context [35], Jini [6] e OSGi framework [53]. Entretanto, foi considerado apenas o uso de Web Service por ser mais simples e atender as necessidades de integração propostas neste trabalho, então optou-se por compreender melhor como funciona tal tecnologia. Em trabalhos futuros pode ser considerado o uso das outras tecnologias devidamente adaptadas para o contexto deste trabalho enriquecendo ainda mais a presente proposta.

De acordo com Andrade e Fiadeiro [3] e Curbera et al [19], Web Services emergiram da necessidade de interação entre aplicações residentes dentro de diferentes instituições. Diversidade essa não somente considerada no nível de linguagem de implementação mas também no nível dos modelos de interação, protocolos de comunicação e qualidade do serviço.

A descrição de um Web Service é realizada em uma linguagem chamada *Web Service Description Language* (WSDL) [80]. Esta linguagem, baseada em XML [81], suporta a descrição de interfaces de serviço, tipos de dados, protocolos de transporte e comunicação de serviços e localização de serviços.

O registro de serviço, chamado de *Universal Description Discovery and Integration* (UDDI) [74], permite a publicação de descritores de serviço, chamados de tipos de serviço, juntamente com os prestadores de serviço. UDDI é um registro de serviço distribuído em que a informação é replicada em diferentes sites, e como consequência, um provedor

de serviço precisa somente publicar seus serviços em um simples registro. Aqui pode-se imaginar, por exemplo, o serviço de rede DNS (*Domain Name System*) em que um site para ser incluído na Internet recebe um endereço IP associado ao seu nome. Dessa forma qualquer usuário na Internet pode acessar o site informando apenas o nome, pois o serviço DNS fica responsável pela conversão do nome no endereço IP (*Internet Protocol*) cadastrado em seu banco de dados.

A seguir são apresentadas as características orientadas a serviço dos Web services.

- **Descrição de Serviço**: em Web services, de acordo com a Figura 2.3, uma descrição de serviço contém as seguintes informações:
  - *definitions*: nome do serviço e *namespace*;
  - *types*: definições de tipos complexos de dados;
  - *message*: descrição de uma mensagem (solicitação ou resposta). Esta descrição contém o nome da mensagem e um número de partes que descrevem o valor de retorno e/ou os valores dos parâmetros;
  - *portType*: descrição do método que une várias mensagens, por exemplo uma solicitação e uma resposta;
  - *binding*: descrição do protocolo de transmissão da mensagem; e
  - *service*: localização do serviço (como uma URI [76]).

- **Publicação de Serviço**: os principais métodos que são fornecidos pela UDDI para publicar informação são:
  - *save_service*: publicar um tipo de serviço; e
  - *save_business*: publicar um provedor de serviço.

  Os principais métodos que permitem que a informação seja removida do registro são:
  - *delete_service*: remove um tipo de serviço; e
  - *delete_business*: remove um provedor de serviço.

- **Pesquisa de Serviço**: UDDI fornece diferentes métodos para pesquisa de serviço, como por exemplo:
  - *find_service*: retorna informações sobre os serviços fornecidos por uma organização; e
  - *find_business*: retorna informações sobre um ou mais provedores de serviço.

  Os métodos de pesquisa permitem buscas utilizando expressões regulares. Estes métodos retornam chaves que podem ser usadas depois para obter informação mais detalhada e os valores podem ser retornados em um modo ordenado seguindo diferentes critérios (por exemplo, ordem alfabética, data de registro e disponibilidade de certificação).

a) Service Description

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="HelloService"
   targetNamespace=
   "http://www.foo.com/wsdl/PrinterService.wsdl" xmlns=...>
   <message name="PrintRequest">
      <part name="data" type="xsd:string"/>
   </message>
   <message name="PrintResponse">
      <part name="jobID" type="xsd:string"/>
   </message>
   <portType name="Print_PortType">
      <operation name="print">
         <input message="tns:PrintRequest"/>
         <output message="tns:PrintResponse"/>
      </operation>
   </portType>
   <binding name="Print_Binding"
            type="tns:Print_PortType">
     ...
   </binding>
   <service name="Print_Service">
      <documentation>
        WSDL pour service impression
      </documentation>
      <port binding="tns:Print_Binding" name="Print_Port">
         <soap:address
           location="http://www.printhost:8080/printsvca"/>
      </port>
   </service>
</definitions>
```

b) Publication

```
save_service (example not given for lack of space)
```

c) Discovery and binding

```
find_service (example not given for lack of space)
```

Figura 2.3: Exemplo de descrição de um Web Service [67].

- **Políticas para criação de objetos do serviço**: em Web Services tanto o método *Shared object* como o método *one object per requester* podem ser usados por um provedor de serviço.
- **Notificações do Serviço**: UDDI pode notificar clientes sobre mudanças no registro com relação a tipos de serviços, sendo que as notificações incluem adição, remoção e modificação.
- **Liberação de objetos do serviço**: Desde que os Web Services suportem diferen-

tes protocolos de comunicação tanto políticas de liberação implícita como explícita podem ser implementadas pelos provedores de serviço.

2.5 Processos de Desenvolvimento Orientado a Serviços

Segundo Erl [24], o ciclo de vida de entrega de software é composto por uma série de etapas, que precisam ser completadas em ordem para construir serviços para uma determinada solução orientada a serviço. O ciclo de entrega de um projeto de software baseado em SOA (*Service-Oriented Architecture*) é ilustrado na Figura 2.4.

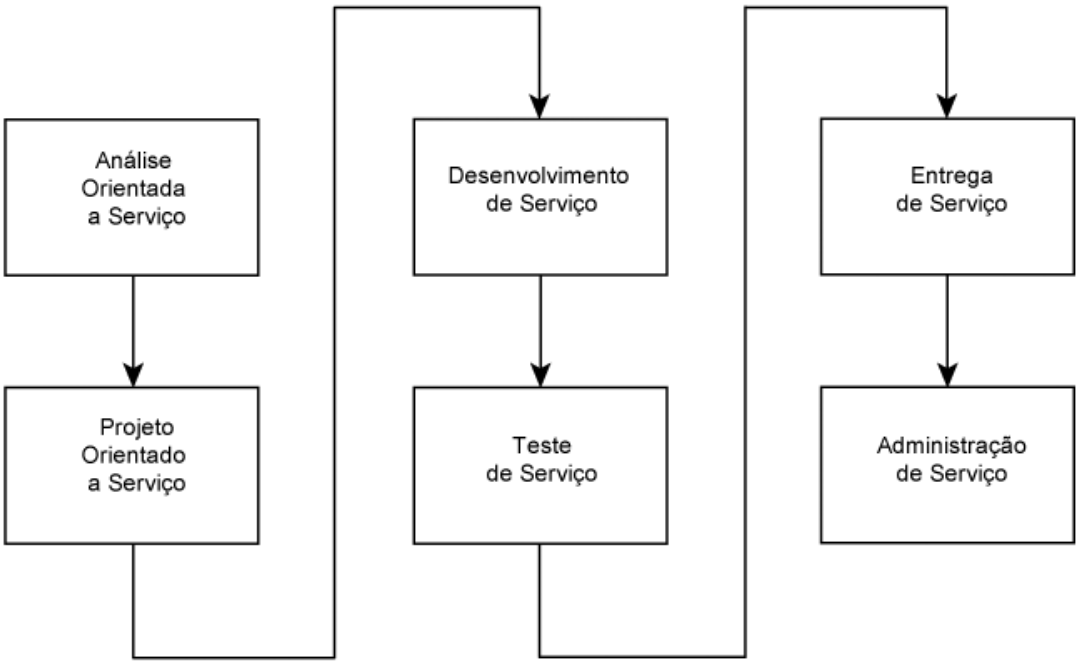


Figura 2.4: Etapas comuns do ciclo de vida de entrega de um projeto de software baseado em SOA. Traduzida de Erl [24]

Conforme visto na Figura 2.4, a primeira etapa é chamada "análise orientada a serviço". Nela o domínio e o processo de negócio da organização, assim como os serviços, são identificados e modelados. Na fase, intitulada "projeto orientado a serviço", o plano de trabalho, bem como as camadas de serviço e as interfaces de serviço são preparadas e projetadas. Em seguida, no estágio de "desenvolvimento de serviço", efetivamente os serviços identificados são implementados usando linguagens de programação e locais físicos de trabalho. Na etapa seguinte, intitulada "teste de serviço", a qualidade dos serviços é testada antes deles realmente serem disponibilizados em um ambiente de produção. Depois na etapa de "entrega de serviço", as partes geradas e interfaces definidas são carregadas, interconectadas e disponibilizadas em um ambiente de produção. Finalmente, na etapa de "administração de serviço", o monitoramento de serviços, o gerenciamento de mensagens, riscos e crises são executados.

Os estágios do ciclo de vida SOA identificados anteriormente representam um simples

caminho sequencial para a construção de serviços. Para que os serviços gerados tenham qualidade é necessário aplicar um processo de desenvolvimento de serviços a este ciclo de vida para organizar as tarefas a serem executadas em cada uma das etapas, sendo que um processo pode:

- auxiliar na escolha de que tipos de camadas de serviço serão disponilizadas;
- coordenar a entrega dos serviços de aplicação, de negócio e de processo; e
- apoiar a transição em direção a padronização SOA enquanto ajuda na especificação total e imediata dos requisitos do projeto;

Erl [24] destaca, ainda, que o sucesso de SOA dentro de uma organização é, geralmente, dependente da medida em que ela é padronizada. Entretanto, o sucesso de um projeto de entrega de uma solução orientada a serviço, geralmente é avaliado pelo escopo em que a solução satifaz os requisitos esperados com relação ao orçamento e tempo de execução previstos.

Para enfrentar este problema é necessário o uso de uma estratégia de desenvolvimento de serviços. Esta estratégia pode ser baseada em prioridades da organização para estabelecer o equilíbrio correto entre a entrega dos objetivos de migração a longo prazo com o cumprimento dos requisitos a curto prazo. Existem diversas estratégias para auxiliar neste problema, mas as três principais são descritas a seguir.

2.5.1 Estratégia *Top-Down*

De acordo com Erl, esta estratégia é uma abordagem que enfatiza muito a etapa de análise de software, que não requer somente processos de negócio para se tornar orientada a serviço, ela também estimula a criação (ou realinhamento) de um modelo de negócio que abrange toda a organização. Este processo é exclusivamente ligado ou derivado de uma lógica de negócio já existente na organização.

A abordagem *top-down* contém tipicamente alguns ou todos os estágios ilustrados na Figura 2.4, conforme pode ser observado na Figura 2.5. É importante observar que este processo assume que os requisitos de negócio já foram coletados e definidos.

Etapa 1: Defina ontologia relevante para a organização

Parte do que uma ontologia estabele é uma classificação de um ou mais conjuntos de informações processadas por uma organização. Isto resulta em um vocabulário comum, bem como a definição de como esses conjuntos de informações se relacionam ente si. Grandes organizações com várias áreas de negócio podem ter diversas ontologias, cada uma governando uma divisão específica de negócios. É esperado que todas essas ontologias especializadas se alinhem para compor a ontologia da organização.

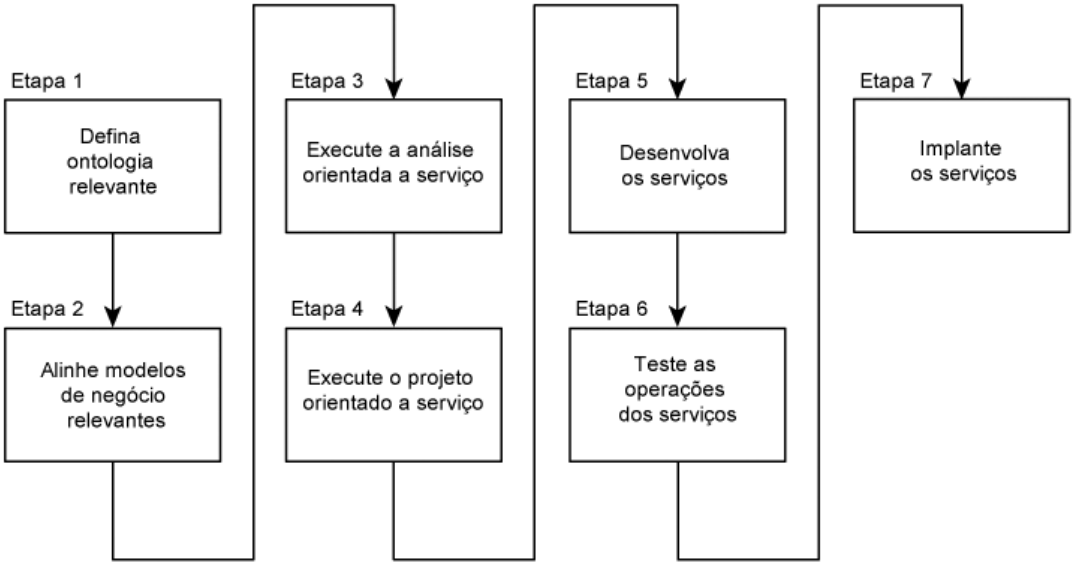


Figura 2.5: Etapas comuns do processo *top-down*. Adaptada de Erl [24].

Se um vocabulário de negócio semelhante ainda não existe para algum dos conjuntos de informação, uma solução é exigida para prosseguir, portanto esta etapa exige que ele seja definido. Uma coleta de informações de alta relevância e análise de negócios em alto nível podem então ser exigida.

Etapa 2: Alinhe modelos de negócio relevantes (incluindo modelos de entidade) com a ontologia nova ou revisada

Após definição da ontologia da organização, os modelos de negócio existentes podem precisar de ajustes (ou serem criados) para representar apropriadamente o vocabulário fornecido pelos termos do modelo de negócio. Modelos de entidade em particular são importantes, pois eles podem ser usados mais tarde como base para os serviços de negócio centrado em entidade.

As etapas 1 e 2, especificadas anteriormente, são pré-requisitos para a fase de análise orientada a serviço, descrita a seguir.

Etapa 3: Execute a análise orientada a serviço

Esta etapa é responsável por determinar quais serviços serão construídos e qual lógica será encapsulada por cada serviço. Para alcançar esses objetivos, nesta etapa é necessário realizar as seguintes atividades:

1. Definir um conjunto preliminar de candidatos a operação de serviço;
2. Agrupar os candidatos a operação de serviço dentro de contextos lógicos. Esses contextos representam os serviços candidatos;

3. Definir preliminarmente os limites do serviço para que eles não sobreponham algum serviço existente ou planejado;
4. Identificar lógica encapsulada com potencial reuso;
5. Garantir que o contexto da lógica encapsulada é apropriada para o uso intencionado; e
6. Definir preliminarmente alguma idéia de modelos de composição de serviço.

Etapa 4: Execute o projeto orientado a serviço

O projeto orientado a serviço é o processo pelo qual projetos de serviços fisícos e concretos são derivados de candidatos a serviços lógicos e então são montados dentro de composições abstratas que implementam um processo de negócio.

Esta etapa é responsável por determinar: como que as definições de interfaces de serviços fisícos serão derivadas dos serviços candidatos modelados durante a fase de análise orientada a serviço; quais características SOA serão utilizadas; e quais padrões industriais e extensões serão exigidos pela SOA para implementar os projetos de serviços planejados e as características SOA definidas.

Para alcançar os objetivos mencionados anteriormente, nesta etapa é necessário realizar as seguintes atividades:

1. Determinar o núcleo de extensões arquiteturais;
2. Definir os limites da arquitetura;
3. Identificar os padrões de projeto exigidos;
4. Definir projetos abstratos de interface de serviço;
5. Identificar potenciais composições de serviço;
6. Avaliar o suporte aos princípios de orientação a serviço; e
7. Explorar o suporte às características de SOA contemporâneas.

Etapa 5: Desenvolva os serviços

Nesta etapa os serviços são desenvolvidos, utilizando linguagens de programação, de acordo com suas respectivas especificações de projeto e descrições de serviço definidas na etapa 4.

Etapa 6: Teste os serviços e todas as operações dos serviços

Nesta etapa de teste é exigido que todas as operações de serviço passem necessariamente pelo controle de garantia de qualidade. Isso tipicamente excede uma quantidade de testes requerida para a lógica de automação ser implementada porque os serviços reusáveis precisarão ser submetidos a testes fora do escopo da solução.

Etapa 7: Implante os serviços

Nesta etapa finalmente os serviços são implantados em um ambiente de produção. Além disso, é importante considerar o potencial reuso futuro dos serviços e para facilitar as várias requisições de serviço, os serviços altamente reusáveis podem exigir um poder de processamento extra e podem ter requisitos de acessibilidade e segurança que precisam ser cumpridos.

2.5.2 Estratégia *Bottom-Up*

De acordo com Erl, esta estratégia encoraja essencialmente a criação de serviços como forma de cumprir os requisitos centrados na aplicação. Web services são construídos basicamente conforme as necessidades e modelados para encapsular lógica de aplicação para melhor satisfazer os requisitos imediatos da solução. A integração é o motivador primário para projetos *bottom-up*, onde a necessidade em levar vantagem sobre *Frameworks* de código-aberto de comunicação SOAP podem ser encontrados simplesmente anexando serviços como integradores para sistemas legados.

Uma abordagem *bottom-up* típica segue um processo similar ao ilustrado na Figura 2.6. É importante observar que este processo assume que os requisitos de negócio já foram coletados e definidos.

Etapa 1: Modele os serviços de aplicação exigidos

Esta etapa resulta na definição de requisitos de aplicação que podem ser cumpridos por meio do uso de Web services. Requisitos típicos incluem a necessidade em estabelecer canais de integração ponto-a-ponto entre sistemas legados e soluções B2B. Outros requisitos comuns surgem do desejo em substituir tecnologia de comunicação remota tradicional por *Framework* de comunicações de mensagem SOAP.

No caso de soluções que empregam a estratégia *bottom-up* para a entrega de soluções altamente centradas em serviço, os serviços de aplicação também serão modelados para incluir lógica e regras de negócio específica. Neste caso, é provável que duas camadas de serviço de aplicação surgirão consistindo de serviços híbridos (serviços que dependem de uma API externa, por exemplo, Google Maps, para realizarem suas funcionalidades) e utilitários (serviços que auxiliam na execução de outros serviços). Estes serviços classificados como reusáveis podem atuar como aplicações finais genéricas para efeitos de integração.

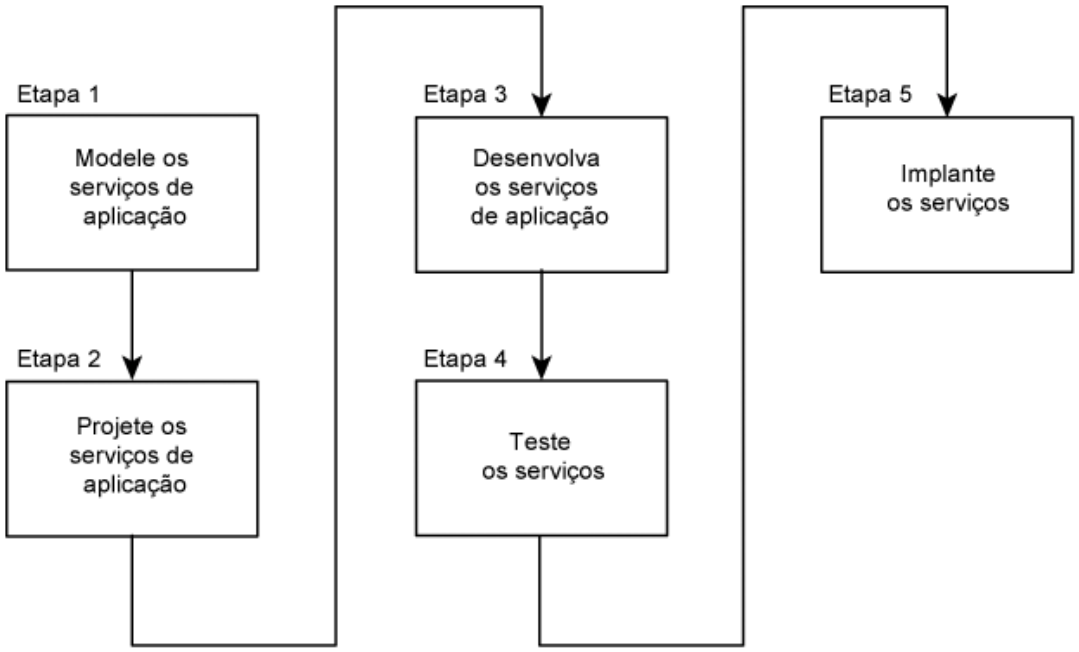


Figura 2.6: Etapas comuns do processo *bottom-up*. Adaptada de Erl [24]

Etapa 2: Projete os serviços de aplicação exigidos

Alguns dos serviços de aplicação modelados na Etapa 1 podem ser entregues por meio do aluguel ou aquisição de serviços de envelope ou quem sabe por meio da criação de serviços de *proxy* gerados automaticamente. Estes serviços podem fornecer pouca oportunidade para projeto adicional. Serviços de aplicação customizáveis, terão de passar por um processo de projeto em que os padrões de projeto serão aplicados para avaliar o nível de consistência. Neste caso existem processos de projeto específicos para serviços de aplicação reusáveis.

Etapa 3: Desenvolva os serviços de aplicação exigidos

Os serviços de aplicação são desenvolvidos de acordo com suas respectivas descrições de serviço e especificações adequadas de projeto.

Etapa 4: Teste os serviços

Os serviços, além do ambiente de solução associado e a estrutura lógica que foi construída, são testados para garantir que o processamento dos requisitos pode ser satisfeito. Testes de desempenho e confiabilidade frequentemente são usados para ajustar o processamento de parâmetros de sistemas legados expostos como serviços encapsuladores. Teste de segurança é também uma importante parte desta fase.

Etapa 5: Implante os serviços

A solução e seus serviços de aplicação são implantados em produção. Considerações de implementação para serviços de aplicação frequentemente incluem requisitos de segurança e desempenho.

2.5.3 Estratégia *Meet-in-the-Middle*

Segundo Erl, o desafio desta abordagem é encontrar um equilíbrio aceitável entre incorporar princípios de projeto orientado a serviço dentro de ambientes de análise de negócios sem ter que esperar pela integração de tecnologias Web Services dentro de ambientes técnicos. Para muitas organizações é importante visualizar essas duas abordagens ao extremo e encontrar um meio termo adequado.

Isto é possível pois esta estratégia define um novo processo que permite que análises no nível de negócio ocorram concorrentemente com o desenvolvimento e projeto de serviços. A estratégia *meet-in-the-middle*, mais conhecida como estratégia ágil, é mais complexa do que as duas apresentadas anteriormente porque ela precisa cumprir dois conjuntos opostos de requisitos, requisitos de negócio e requisitos de aplicação.

As etapas da estratégia ágil, ilustradas na Figura 2.7 apresentam como é possível alcançar os respectivos objetivos das estratégias *top-down* e *bottom-up* em uma única estratégia.

Etapa 1: Inicie a análise *top-down*, focando primeiro nas partes chaves da ontologia e entidades de negócio relacionadas

A análise padrão *top-down* inicia com um foco mais limitado. As partes dos modelos de negócio diretamente relacionados a lógica de negócio sendo automatizada recebem prioridade máxima.

Etapa 2: Quando a análise *top-down* for suficientemente progredida, execute a análise orientada a serviço

Enquanto a Etapa 1 ainda está em andamento, esta etapa inicia a fase da análise orientada a serviço. Dependendo da magnitude das análises exigidas para completar a Etapa 1, é aconselhável dar um bom avanço nesta etapa, pois quanto mais progredir nesta etapa mais projetos de serviço serão beneficiados depois.

Após a análise *top-down* ter progredido suficientemente, modele os serviços de negócio para melhor representar o modelo de negócio com qualquer um dos resultados da análise que estão disponíveis. Isto é a decisão principal neste processo. Neste momento pode-se exigir um julgamento rígido para determinar se a análise *top-down*, enfim, é suficientemente madura para proceder com a criação dos modelos de serviço de negócio ou não.

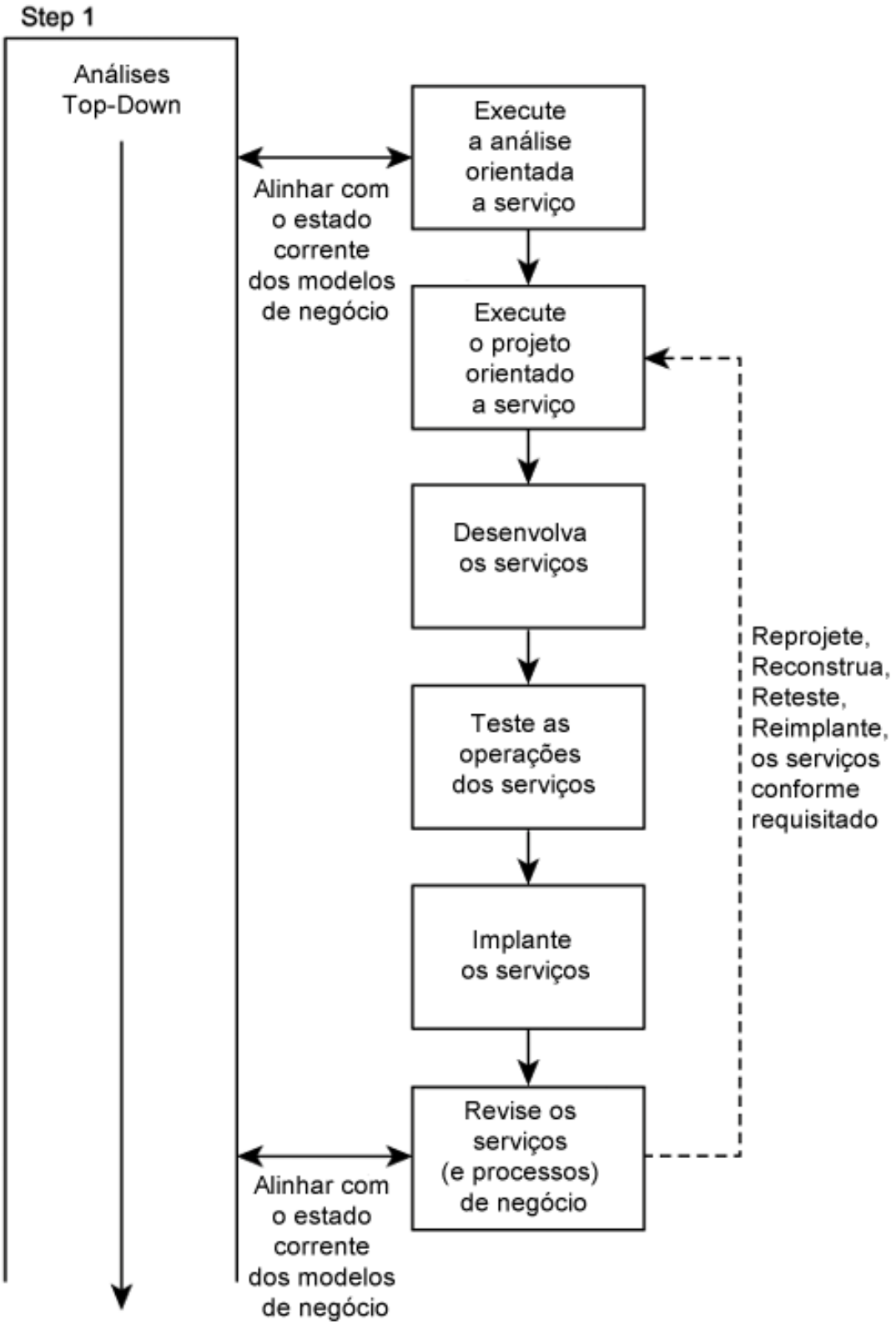


Figura 2.7: Um processo simples da estratégia ágil. Adaptado de Erl [24]

Esta consideração deve então ser avaliada contra a importância e urgência dos requisitos de projeto pendentes.

Etapa 3: Execute o projeto orientado a serviço

As camadas de serviço selecionadas são definidas, e serviços individuais são projetados como parte do processo de projeto orientado a serviço.

Etapas 4, 5, e 6: Desenvolva, teste e implante os serviços

Desenvolva os serviços e submeta-os aos testes padrões e a rotina de implantação.

Etapa 7: Enquanto a análise *top-down* continuar progredindo, revise os serviços de negócio periodicamente

Execute avaliações periódicas de todos os serviços de negócio para comparar seus projetos com o estado corrente dos modelos de negócio. Faça um anotação das discrepâncias e agende um reprojeto para aqueles serviços fora dos padrões. Isso tipicamente vai exigir uma prorrogação de um serviço já existente para que seja feita uma melhor análise das capacidades de serviço exigidas. Quando reprojetado, um serviço precisará novamente passar pelas etapas de desenvolvimento, teste e implantação.

Para preservar a integridade produzida por esta abordagem, o conceito de contratos de serviço imutáveis precisam ser estritamente reforçados. Após um contrato estar publicado, não pode ser alterado. A não ser que as revisões de serviço resultem em extensões que impõem restrições em um contrato existente (como adição de novas operações na descrição WSDL), a etapa 7 deste processo provavelmente resultará na necessidade de publicar novas versões de contrato e requisitos para um sistema de gerenciamento de versão.

2.5.4 Análise Comparativa dos Processos de Desenvolvimento de Serviços

A Tabela 2.1 apresenta as vantagens e desvantagens, propostas por Erl [24], no emprego das estratégias de desenvolvimento orientado a serviço apresentadas anteriormente, em projetos de *software* orientado a serviço.

Baseado nas vantagens e desvantagens das estratégias de desenvolvimento de serviços apresentadas na Tabela 2.1, a estratégia que melhor se adaptada ao processo de integração proposto no Capítulo 4 é a *bottom-up*, por ser uma estratégia voltada para a adoção da tecnologia Web Service em aplicações legadas, como é o caso do problema tratado neste trabalho de mestrado. Outro fator que contribuiu para escolha da estragégia *bottom-up* é que ela permite a criação de protótipos de serviços da aplicação o que agiliza o processo

Tabela 2.1: Vantagens e desvantagens do emprego das estratégias *Top-Down*, *Bottom-Up* e *Meet-in-the-middle* em projetos de *software* orientado a serviço, segundo Erl [24].

Estratégia	**Vantagens**	**Desvantagens**
Top-Down	Geralmente resulta em uma arquitetura de serviço de alta qualidade já que o projeto e os parâmetros lógicos ao redor de cada serviço são analizados a fundo, maximizando o potencial de reusabilidade e oportunidade para composições simplificadas.	Os obstáculos para seguir esta abordagem estão associados com tempo e custo financeiro. As organizações são exigidas a investir significativamente em análise de projetos que podem tomar muito tempo, sem obter retornos imediatos.
Bottom-Up	É utilizado pela maioria das organizações já que simplesmente adicionam Web services aos seus ambientes de aplicação existentes para alavancar o uso da tecnologia Web services. A arquitetura entre os Web services e a própria aplicação permanece inalterada e os princípios de orientação a serviço são então raramente considerados.	Esta estratégia realmente não é considerada uma abordagem válida para obter SOA moderna. Embora esta estratégia permita a criação eficiente de Web services exigidos pelas aplicações, implementar uma SOA apropriada tardiamente pode resultar em muitos reajustes ou até mesmo na introdução de novas camadas de serviço padronizadas posicionadas sobre o topo dos serviços não padronizados produzidos por esta abordagem.
Meet-in-the-middle	Emprega o que tem de melhor nas outras duas estratégias combinando em uma única abordagem para concretizar a aplicação de SOA. Descobre requisitos imediatamente sem comprometer a integridade do modelo de negócio de uma organização e as qualidades orientadas a serviço da arquitetura.	O sucesso desta estratégia está associado com a entrega de cada serviço logo se os serviços não forem entregues brevemente o emprego desta estratégia poderá fracassar. Ela impõe tarefas de gerenciamento que são exigidas para garantir que serviços existentes são realmente mantidos em conformidade com modelos de negócio revisados, sendo que mesmo assim os serviços ainda correm o risco de ficarem incompatíveis com a troca constante do modelo de negócio da organização.

de integração com outra aplicação, já que os testes de integração podem ser realizados com a troca de mensagens XML entre as duas aplicações.

Como as aplicações legadas serão integradas e fornecerão apenas um serviço web que será responsável pela autenticação remota na aplicação, a estratégia *bottom-up* apresentada anteriormente será adaptada para fornecer as atividades necessárias para a definição do processo de integração proposto neste trabalho.

O processo de integração que será apresentado no Capítulo 3 foi definido como uma variação da estratégia *Bottom-Up* junto com mecanismos da estratégia *Meet-in-the-Middle*. O desenvolvimento ou evolução de um software orientado a serviço, na estratégia *Bottom-Up*, é sequencial, já no processo de integração proposto permite o reprojeto, reimplementação, reteste e reimplatação da aplicação durante a integração dela a um ambiente integrador de aplicações também proposto neste trabalho (veja mais informações na Seção 3.3 Capítulo 3). O processo proposto também considera que as aplicações legadas de acesso restrito serão adaptadas para uma arquitetura orientada a serviço e disponibilizarão apenas um único serviço responsável pelo processo de autenticação e acesso à aplicação.

2.6 Considerações Finais

Conforme visto neste capítulo, foram apresentados: o paradigma orientado a serviço, a tecnologia *Web Service* e três estratégias para o desenvolvimento de *software* baseado em serviços. A estratégia *bottom-up* foi a escolhida dentre as três apresentadas e servirá de base, bem como a adoção da tecnologia *Web Service*, para a definição do processo para integração baseada em serviços de aplicações web legadas e a construção do ambiente integrador de aplicações que serão apresentados no Capítulo 3.

Capítulo 3

Integração baseada em Serviços de Aplicações Web Legadas

3.1 Considerações Iniciais

Neste capítulo é apresentada a proposta do presente trabalho. A proposta inclui a definição de um processo para integração baseada em serviços de aplicações web legadas, intitulado PIBSAWL. Em seguida, é apresentado o ambiente integrador de aplicações, intitulado AIA, que apoia a execução do PIBSAWL realizando a integração de aplicações legadas.

O PIBSAWL será descrito, logo a seguir, utilizando alguns elementos fundamentais do arcabouço do RUP [38]: atividades, passos, papéis, artefatos de entrada e de saída, guia de gabaritos para artefatos de saída e guia de ferramentas de apoio computacional.

3.2 PIBSAWL: Processo para Integração baseada em Serviços de Aplicações Web Legadas

O PIBSAWL foi especificado, seguindo a notação BPMN[1], como uma série de atividades para serem executadas durante a integração de uma aplicação web legada ao ambiente integrador. A Figura 3.1 ilustra as dez atividades que devem ser executadas de forma iterativa e incremental com o intuito de alcançar o objetivo final que é a integração da aplicação web legada ao AIA. A seguir são descritas cada uma das atividades do PIBSAWL.

[1] *Business Process Modeling Notation*, ou Notação de Modelagem de Processos de Negócio é uma notação da metodologia de gerenciamento de processos de negócio e trata-se de uma série de ícones padronizados para o desenho de processos, o que facilita o entendimento do usuário.

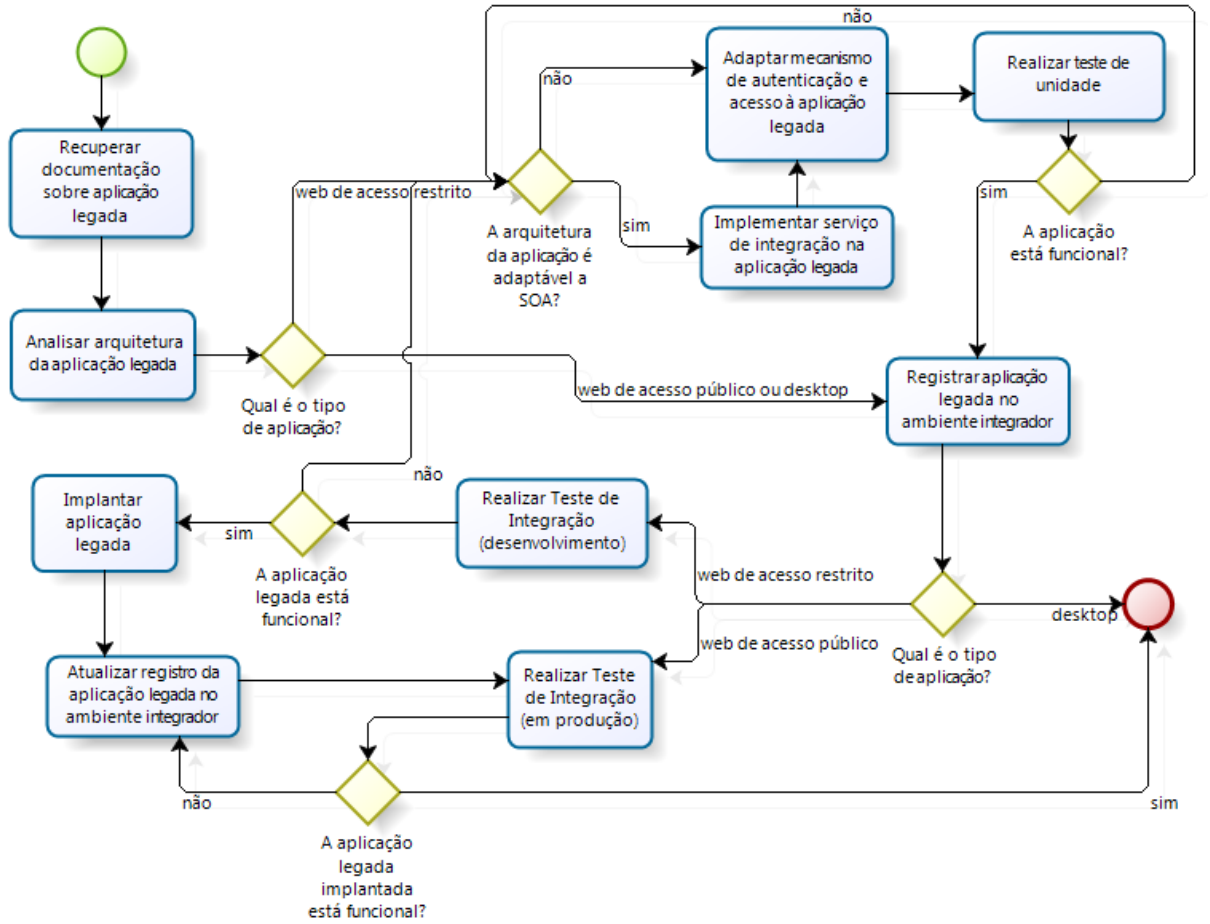


Figura 3.1: Modelagem do PIBSAWL utilizando o BPMN.

Atividade 1: Recuperar documentação sobre a aplicação legada

Objetivo: Recuperar documentação técnica (como foi desenvolvida a aplicação) e manuais de uso (como deve ser utilizada a aplicação) sobre a aplicação a ser integrada ou efetuar uma engenharia reversa caso não exista tais documentações. Todo tipo de documentação sobre a aplicação é importante mas procure recuperar e analisar somente aquilo que for necessário para uma compreensão geral do uso, processo de desenvolvimento e estrutura lógica da aplicação.

Papéis: Analista de Sistema

Artefatos de Entrada: Código-fonte da aplicação, Documento da Arquitetura da Aplicação, Manual de Uso e Instalação da Aplicação.

Artefatos de Saída: Todos os artefatos produzidos ou recuperados pela atividade.

Passos:

1. Consultar equipe de desenvolvimento ou entidade detentora da aplicação legada sobre permissão de acesso ao código-fonte dela.

2. Caso seja concedido acesso ao código-fonte, verificar as licenças de uso e modificações da aplicação. Caso contrário, a aplicação não poderá ser integrada. Ela será apenas registrada no ambiente integrador. Prossiga para a atividade 6.

3. Caso a licença de uso da aplicação legada permita realizar alterações nela, então prossiga para o próximo passo. Caso contrário, a aplicação não poderá ser integrada. Ela será apenas registrada no ambiente integrador. Prossiga para a atividade 6.

4. Recuperar ou criar um documento que especifique a arquitetura da aplicação legada.

5. Recuperar ou criar um manual de uso e instalação da aplicação legada.

6. Prosseguir para a atividade 2.

Ferramentas: Ferramenta CASE[2] que apóia modelagem em UML (por exemplo, Rational Rose[3] e ArgoUML[4]).

Atividade 2: Analisar a arquitetura da aplicação legada

Objetivo: Analisar a arquitetura da aplicação legada para determinar se a aplicação legada pode ser adaptada para uma arquitetura orientada a serviço ou não.

Papéis: Arquiteto de Software

Artefatos de Entrada: Código-fonte da aplicação e Documento da Arquitetura da Aplicação.

Artefatos de Saída: Documento da Arquitetura da Aplicação e Documento com Especificação de Requisitos de Integração da Aplicação.

Passos:

1. Se a plataforma da aplicação legada for do tipo *Desktop* ela não poderá ser integrada. Neste caso, ela será apenas registrada no ambiente integrador, prossiga para a atividade 6. Caso contrário, a plataforma da aplicação legada é do tipo *Web* então, prossiga para o próximo passo.

2. Analisar a arquitetura da aplicação legada, com base nos artefatos de entrada, para determinar se a aplicação legada pode ser adaptada para uma arquitetura orientada a serviço. Neste caso deve-se levar em consideração também se a linguagem de programação na qual foi desenvolvida a aplicação tem suporte às tecnologias Web Service e LDAP.

[2] *Computer-Aided Software Engineering*, ou Engenharia de Software Apoiada por Computador, é uma classificação que abrange todas as ferramentas baseadas em computadores que auxiliam atividades de engenharia de software, desde análise de requisitos e modelagem até programação e testes.

[3] www.ibm.com/software/awdtools/developer/rose/

[4] http://argouml.tigris.org/

3. Definir a nova arquitetura da aplicação legada. Se a aplicação legada não comportar uma arquitetura orientada a serviço pode-se adaptar a arquitetura da própria aplicação para autenticar seus usuários em uma base LDAP ao invés de sua própria base de dados.

4. Atualizar, caso a aplicação seja do tipo *web* de acesso restrito, o primeiro artefato de saída desta atividade com a nova arquitetura da aplicação: orientada a serviço autenticando usuários na base LDAP ou apenas autenticando usuários na base LDAP.

5. Elaborar o segundo artefato de saída desta atividade com a especificação de requisitos quanto a integração da aplicação legada com base no gabarito definido na Tabela 3.1.

6. Se a aplicação é adaptável à SOA (uma aplicação é adaptável a SOA quando as tecnologias e métodos utilizados para construção da aplicação permitem aplicar uma arquitetura orientada a serviço) então, prossiga para a atividade 3. Caso contrário, prossiga para a atividade 4.

Gabarito: Na Tabela 3.1 é apresentado um guia de gabarito do artefato "Documento com Especificação de Requisitos de Integração da Aplicação" que deve ser produzido nesta atividade para servir de apoio na implementação dos mecanismos de autenticação e acesso (via WS ou via LDAP) a serem desenvolvidos na aplicação legada.

Atividade 3: Implementar serviço de integração na aplicação legada

Objetivo: Implementar um web service na aplicação legada capaz de autenticar usuários autenticados e provenientes do ambiente integrador na aplicação legada utilizando tecnologia Web Service.

Papéis: Desenvolvedor Web

Artefatos de Entrada: Última versão do Código-fonte da Aplicação, Documento com Especificação de Requisitos de Integração da Aplicação e Última versão do Documento da Arquitetura da Aplicação.

Artefatos de Saída: Nova versão do Código-fonte da Aplicação.

Passos:

1. Identificar na aplicação legada quais métodos (caso a arquitetura seja orientada a objetos), serviços (caso a arquitetura seja orientada a serviço) ou funções (caso a arquitetura seja procedural) são responsáveis pela autenticação de usuários na aplicação legada.

2. Utilizando o artefato "Documento com Especificação de Requisitos de Integração da Aplicação", criar um web service na aplicação legada replicando os itens identificados anteriormente e inserindo-os de forma lógica no web service criado. Com isso a aplicação legada continuará em funcionamento já que o mecanismo de autenticação

Tabela 3.1: Documento com Especificação de Requisitos de Integração da Aplicação.

Id.	**Descrição**
	Aplicação Web Adaptável a SOA
REQ_1	A aplicação deve prover um Web service (WS) que receba duas informações (*login* do usuário (*string* com até 64 caracteres) e *hash* (*string* com até 512 caracteres)). Com base nestas informações o WS deve realizar a autenticação do usuário recuperando a *hash* randômica da base LDAP e comparando com as informações enviadas. Por fim, o WS deve retornar o seguinte padrão de mensagem, um vetor contendo os seguintes campos, "STATUS" (valor 1 indicando que a autenticação ocorreu com sucesso ou 0 caso contrário), "MESSAGE" (texto informando motivo do erro no processo de autenticação) e "HASH" (*hash* randômica e temporária gerada pela própria aplicação que dará acesso autenticado à aplicação em período de tempo limitado).
REQ_2	O mecanismo de acesso à aplicação deverá ser adaptado para receber, por GET, o envio de um login e uma hash correspondentes, respectivamente, ao login do usuário na base LDAP e hash retornada pela WS da aplicação legada. Com isto, a aplicação deverá comparar essas duas informações com informações registradas na base LDAP associadas ao login do usuário. Caso estejam corretas, a aplicação deverá liberar o acesso do usuário à aplicação, caso contrário, deverá impedir o acesso.
	Aplicação Web não Adaptável a SOA
REQ_2	O mecanismo padrão de autenticação e acesso à aplicação legada deverá ser replicado e adaptado para autenticar usuários utilizando informações da base LDAP e não mais da própria base da aplicação. Com isso o mecanismo padrão de autenticação e acesso da aplicação legada continuará autenticando os usuários já existentes na base da aplicação e o novo mecanismo criado passará a autenticar os usuários oriundos do ambiente integrador.
	Qualquer Aplicação Web de Acesso Restrito
REQ_3	Usuários autenticados na aplicação com sucesso por meio da base LDAP e WS mas que não possuam registro na aplicação podem ser cadastrados no banco de dados da própria aplicação utilizando informações do usuário presentes na base LDAP. Isso possibilita o correto uso da aplicação, já que algumas aplicações exigem que os usuários estejam cadastrados em sua própria base de dados e não somente na base LDAP.
REQ_4	A aplicação não tem permissão para modificar a base LDAP. Ela pode apenas visualizar e recuperar informações de usuários baseado em um *login* e senha de acesso à base LDAP, fornecidos via contato por email ou telefone, disponibilizados no ambiente integrador.
	Demais Aplicações
REQ_5	No caso de aplicações do tipo *desktop* ou *web* de acesso público, a aplicação não precisará ser adaptada, ela será apenas registrada no ambiente integrador. Aplicações do tipo *desktop* não poderão ser acessadas através do ambiente integrador, já aplicações do tipo *web* de acesso público, sim.

da aplicação será apenas replicado para realizar a autenticação através de um web service. O web service deverá receber como parâmetros de entrada as seguintes informações: *login* (login do usuário) e *hash* (hash randômica e temporária simbolizando a senha do usuário) e retornar um vetor com os seguintes campos: *STATUS* (valor 1 indicando que a autenticação ocorreu com sucesso ou 0 caso contrário), *MESSAGE* (texto informando motivo do erro no processo de autenticação) e *HASH* (nova hash randômica e temporária que permitirá acesso autenticado à aplicação).

3. Prosseguir para a atividade 4.

Atividade 4: Adaptar mecanismo de autenticação e acesso à aplicação legada

Objetivo: Adaptar o mecanismo de autenticação e acesso à aplicação para permitir que apenas usuários autenticados e provenientes do ambiente integrador possam concretizar um acesso restrito à aplicação legada, utilizando para isso a tecnologia LDAP.

Papéis: Desenvolvedor Web

Artefatos de Entrada: Última versão do Código-fonte da Aplicação, Última versão do Documento da Arquitetura da Aplicação, Última versão do Documento com Especificação de Requisitos de Integração da Aplicação.

Artefatos de Saída: Nova versão do Código-fonte da Aplicação e Documento de Casos de Teste de Unidade.

Passos:

1. Recuperar do artefato "Última versão do Documento da Arquitetura da Aplicação" as informações de acesso à base LDAP, tais como: host, porta, login e senha. Além da estrutura hieráquica da base LDAP. Caso não estejam disponibilizadas neste artefato, solicitar para a equipe de desenvolvimento do ambiente integrador tais informações através de email ou telefone disponibilizados na página de suporte do ambiente integrador.

2. Caso a aplicação legada seja adaptável à SOA, então altere o web service criado na atividade anterior para realizar a autenticação de usuários utilizando as informações de acesso à base LDAP, recuperadas no passo anterior, e a estrutura hieráquica da base LDAP. Caso contrário, prossiga para o passo 4.

3. Criar mecanismo de acesso à aplicação legada permitindo receber um *login* e *hash*, retornados pelo próprio web service da aplicação, através de mensagens GET, do protocolo HTTP. Em seguida, comparar informações recebidas com dados armazenados na base LDAP e permitir acesso ou não à aplicação legada. Caso o usuário ainda não exista no banco de dados da aplicação, crie o usuário utilizando informações da base LDAP e atribua as permissões necessárias a ele. Por fim, prossiga para o passo 6.

4. Caso a aplicação legada não seja adaptável à SOA, então replique o mecanismo de autenticação dela e adapte-o para realizar a autenticação dos usuários utilizando as

informações de acesso à base LDAP, recuperadas no passo 1, e a estrutura hieráquica da base LDAP.

5. Criar mecanismo de acesso à aplicação legada permitindo receber um *login* e *senha* através de mensagens GET ou POST, do protocolo HTTP. Em seguida, comparar informações recebidas com dados armazenados na base LDAP e permitir acesso ou não à aplicação legada. Caso o usuário ainda não exista no banco de dados da aplicação, crie o usuário utilizando informações da base LDAP e atribua as permissões necessárias a ele.

6. Elaborar o Documento de Casos de Teste de Unidade especificando valores de entrada e valores de retorno para os processos de autenticação e acesso à aplicação legada.

7. Prosseguir para a atividade 5.

Tabela 3.2: Documentação dos casos de teste de unidade da aplicação legada.

Id.	**Entrada**	**Saída Esperada**
Autenticação via web service da aplicação		
CT_1	Login exist. no LDAP e *hash* gravada no LDAP	STATUS='1', MESSAGE='', HASH='$hash_x$'
CT_2	Login exist. no LDAP e uma *hash* qualquer	STATUS='0', MESSAGE='erro de autenticação', HASH=''
CT_3	Login não exist. no LDAP e *hash* gravada no LDAP	STATUS='0', MESSAGE='erro de autenticação', HASH=''
CT_4	Login não exist. no LDAP e uma *hash* qualquer	STATUS='0', MESSAGE='erro de autenticação', HASH=''
Acesso via mecanismo de autenticação web service		
CT_5	Login existente no LDAP e $hash_x$	Permitir acesso à aplicação
CT_6	Login não exist. no LDAP e $hash_x$	Não permitir acesso à aplicação
CT_7	Login exist. no LDAP e Hash qualquer	Não permitir acesso à aplicação
CT_8	Login não exist. no LDAP e Hash qualquer	Não permitir acesso à aplicação
Acesso via mecanismo de autenticação LDAP		
CT_9	Login e senha existente no LDAP	Permitir acesso à aplicação
CT_{10}	Login exist. e senha não exist. no LDAP	Não permitir acesso à aplicação
CT_{11}	Login não existente e senha exist. no LDAP	Não permitir acesso à aplicação
CT_{12}	Login e senha não exist. no LDAP	Não permitir acesso à aplicação
Acesso via mecanismo de autenticação padrão		
CT_{13}	Login e senha existente na base da aplicação	Permitir acesso à aplicação
CT_{14}	Login exist. e senha não exist. no BD	Não permitir acesso à aplicação
CT_{15}	Login não exist. e senha exist. no BD	Não permitir acesso à aplicação
CT_{16}	Login e senha não exist. no BD	Não permitir acesso à aplicação

Gabarito: Na Tabela 3.2 é apresentado um guia de gabarito do artefato "Documento de Casos de Teste de Unidade" que deve ser produzido nesta atividade para compor os

testes de autenticação e acesso à aplicação. Nele os testes estão divididos em mecanismos de autenticação e acesso via Web Service, LDAP e padrão da aplicação.

Atividade 5: Realizar teste de unidade

Objetivo: Executar testes de unidade, considerando a aplicação legada como a unidade, com o intuito de identificar possíveis erros na lógica de programação do processo de autenticação e acesso de usuários criados na aplicação legada.

Papéis: Testador Web.

Artefatos de Entrada: Documento de Casos de Teste de Unidade.

Artefatos de Saída: Relatório de Execução dos Casos de Teste de Unidade.

Passos:

1. Realizar teste de unidade no mecanismo de autenticação e acesso já existente da aplicação legada.
2. Realizar teste de unidade no mecanismo de autenticação e acesso criado na aplicação legada.
3. Elaborar o artefato "Relatório de Execução dos Casos de Teste de Unidade".
4. Caso a aplicação legada esteja funcional prossiga para a atividade 6. Caso contrário, se a arquitetura da aplicação é do tipo adaptável a SOA, prossiga para a atividade 4, caso contrário, prossiga para a atividade 3.

Tabela 3.3: Relatório de execução dos casos de teste de unidade da aplicação legada.

Id. Caso de Teste de Unidade	**Resultado da Execução**
CT_1	<*Correto ou Incorreto*>
CT_2	<*Correto ou Incorreto*>
CT_n	<*Correto ou Incorreto*>

Gabarito: Na Tabela 3.3 é apresentado um guia de gabarito do artefato "Relatório de Execução de Casos de Teste de Integração" que deve ser produzido nesta atividade para compor os testes de autenticação e acesso à aplicação. O documento está dividido em testes nos mecanismos de autenticação e acesso via Web Service, LDAP e padrão da aplicação.

Atividade 6: Registrar aplicação legada no ambiente integrador

Objetivo: Registrar, no ambiente integrador, informações gerais de autenticação e de acesso à aplicação legada para o correto funcionamento da aplicação legada no ambiente

integrador. Vincular manauis de uso (e de instalação quando a aplicação for do tipo *desktop*) sobre a aplicação legada no ambiente integrador.

Papéis: Desenvolvedor Web.

Artefatos de Saída: Ambiente integrador de aplicações atualizado.

Passos:

1. Registrar no ambiente integrador informações gerais da aplicação legada e vincular manuais de uso (e de instalação quando a aplicação for do tipo *desktop*) da aplicação, recuperados na Atividade 1, ao ambiente integrador para facilitar o entendimento sobre a aplicação.
2. Caso a plataforma da aplicação legada seja do tipo *Desktop*, então finalize a execução do processo.
3. Caso a plataforma da aplicação seja do tipo *Web* de acesso público, então, registre, no ambiente integrador, as informações de acesso (url, porta de comunicação, entre outras) da aplicação legada e prossiga para a atividade 11, já que a aplicação legada já está em produção.
4. Caso contrário, a plataforma é do tipo *Web* de acesso restrito, então, registre, no ambiente integrador, as informações de autenticação (método de autenticação: GET, POST ou WEBSERVICE) e de acesso (url, porta de comunicação, entre outras) da aplicação legada.
5. Elaborar o artefato "Documento de Casos de Teste de Integração" especificando valores de entrada e valores de retorno para o processo de integração entre a aplicação legada e o ambiente integrador, de acordo com o método de autenticação, via Web Service ou LDAP.
6. Prosseguir para a atividade 7.

Gabarito: Na Tabela 3.4 é apresentado um guia de gabarito do artefato "Documento de Casos de Teste de Integração" que deve ser produzido nesta atividade para compor os testes de integração da aplicação legada ao ambiente integrador. Nele os testes estão divididos em acesso à aplicação legada do tipo *desktop*, *web* de acesso público e *web* de acesso restrito. Além disso, tem-se as divisões das aplicações de acesso restrito em: acesso à aplicação *web* legada via web service e acesso à aplicação *web* legada via LDAP.

Atividade 7: Realizar teste de integração (desenvolvimento)

Objetivo: Executar testes de unidade por meio do ambiente integrador, considerando a aplicação legada como a unidade e que ela está em uma ambiente de desenvolvimento, a fim de identificar possíveis erros na implementação do processo de autenticação e acesso de usuários à aplicação legada.

Papéis: Testador Web.

Tabela 3.4: Documentação dos casos de teste de integração da aplicação legada ao ambiente integrador.

Id.	**Entrada**	**Saída**
Acesso à aplicação legada do tipo *desktop*		
CTI_1	Registro apenas de informações gerais	Não permitir acesso
Acesso à aplicação legada do tipo *web* de acesso público		
CTI_2	Registro apenas de informações gerais e de acesso	Permitir acesso
Acesso à aplicação legada do tipo *web* de acesso restrito		
CTI_3	Registro somente das informações gerais	Não permitir acesso
Acesso à aplicação *web* legada via Web Service		
CTI_4	Parâmetros de comunicação: login por GET e hash por GET. Parâmetros de acesso: url, porta e path existentes.	Permitir acesso
CTI_5	Parâmetros de comunicação: login por GET ou hash por GET. Parâmetros de acesso (url, porta e path) existentes.	Não permitir acesso
CTI_6	Nenhum parâmetro de comunicação. Parâmetros de acesso (url, porta e path) existentes.	Não permitir acesso
CTI_7	Quaisquer parâmetros de comunicação. Parâmetros de acesso (url, porta e path) inexistentes.	Não permitir acesso
Acesso à aplicação *web* legada via LDAP		
CTI_8	Parâmetros de comunicação: login por GET/POST e senha por GET/POST. Parâmetros de acesso (url, porta e path) existentes.	Permitir acesso
CTI_9	Parâmetros de comunicação: login por GET/POST ou senha por GET/POST. Parâmetros de acesso (url, porta e path) existentes.	Não permitir acesso
CTI_{10}	Quaisquer parâmetros de comunicação. Parâmetros de acesso (url, porta e path) inexistentes.	Não permitir acesso

Artefatos de Entrada: Documento de Casos de Teste de Integração.

Artefatos de Saída: Relatório de Execução dos Casos de Testes de Integração.

Passos:

1. Executar teste de unidade no mecanismo de autenticação e acesso criado na aplicação legada utilizando o ambiente integrador e o Documento de Casos de Teste de Integração, de acordo com o método de integração utilizado na aplicação legada: via Web Service ou LDAP
2. Elaborar o artefato "Relatório de Execução dos Casos de Testes de Integração".
3. Caso a aplicação legada esteja funcional, prossiga para a atividade 8.
4. Caso contrário, retorne para a atividade 3 identificando possíveis erros de implementação na aplicação legada quanto aos novos mecanismos de autenticação e acesso criados na aplicação.

Tabela 3.5: Relatório de execução dos casos de teste de integração da aplicação legada ao ambiente integrador.

Id. Caso de Teste de Integração	**Resultado da Execução**
CTI_1	*<Correto ou Incorreto>*
CTI_2	*<Correto ou Incorreto>*
CTI_n	*<Correto ou Incorreto>*

Gabarito: Na Tabela 3.5 é apresentado um guia de gabarito do artefato "Relatório de Execução de Casos de Teste de Integração" que deve ser produzido nesta atividade para compor os testes de autenticação e acesso à aplicação.

Atividade 8: Implantar aplicação legada

Objetivo: Implantar a aplicação legada em um ambiente de produção.

Papéis: Desenvolvedor Web.

Passos:

1. Implantar, em um ambiente de produção, os arquivos alterados da aplicação legada. Com isto, a aplicação legada poderá atender às novas requisições de autenticação e acesso do ambiente integrador em um ambiente produção, ou seja, em um ambiente real de utilização da aplicação.
2. Prosseguir para a atividade 9.

Atividade 9: Atualizar registro da aplicação legada no ambiente integrador

Objetivo: Atualizar registro da aplicação legada no ambiente integrador.

Papéis: Desenvolvedor Web.

Artefatos de Saída: Nova versão do ambiente integrador.

Passos:

1. Atualizar registro da aplicação legada no ambiente integrador alterando os parâmetros de acesso à aplicação, ou seja, apontando-os para o ambiente de produção no qual a aplicação legada foi implantada.
2. Prosseguir para a atividade 10.

Atividade 10: Realizar teste de integração (produção)

Objetivo: Executar testes de unidade através do ambiente integrador, considerando a aplicação legada como a unidade e que ela está em uma ambiente de produção, a fim

de identificar possíveis erros nas informações da aplicação legada registradas no ambiente integrador.

Papéis: Testador Web.

Artefatos de Entrada: Documento de Casos de Teste de Integração.

Artefatos de Saída: Relatório de Execução dos Casos de Testes de Integração.

Passos:

1. Executar teste de unidade nos mecanismos de autenticação e acesso criados na aplicação legada utilizando o ambiente integrador e considerando que a aplicação legada está implantada em um ambiente de produção. Para realizar este teste utilize uma nova instância do "Documento de Casos de Testes de Integração", definido na Tabela 3.4. Se a aplicação legada for do tipo web de acesso público, realize o teste apenas no mecanismo de acesso à aplicação já que a aplicação não possui acesso restrito, ou seja, não exige autenticação.

2. Elaborar o artefato "Relatório de Execução dos Casos de Testes de Integração" utilizando uma nova instância do artefato definido na Tabela 3.5.

3. Em caso de sucesso na execução dos testes, encerre a execução do processo. Caso contrário, prossiga para a atividade 9 e reajuste os parâmetros de acesso da aplicação legada.

3.3 AIA: Ambiente Integrador de Aplicações

O ambiente integrador de aplicações AIA foi desenvolvido com a finalidade de integrar aplicações legadas e servir de suporte ao processo PIBSAWL. Para aplicações legadas do tipo *web*, o AIA permite o registro da aplicação, a dispobilização para *download* de manuais de uso sobre a aplicação e o seu acesso via *browser*. Já para aplicações legadas do tipo *desktop*, o AIA permite o registro da aplicação e a disponibilização para *download* da aplicação e manuais de uso e instalação da aplicação.

A seguir são apresentadas as principais tecnologias adotadas para a construção do ambiente proposto, bem como as justificativas para a utilização delas. Em seguida, é apresentado a arquitetura do ambiente e artefatos que descrevem mais detalhadamente como ele foi desenvolvido ambiente.

3.3.1 Principais Tecnologias Adotadas

Para agilizar e facilitar o processo de construção do AIA, foram definidas as seguintes características que o ambiente deve possuir: um framework de código aberto para a instanciação de gerenciadores de conteúdo com suporte à Web Service e LDAP; um sistema de gerenciamento de banco de dados para armazenamento das informações das aplicações;

um serviço de diretórios (tal como uma base LDAP) para armazenamento centralizado das informações dos usuários do ambiente e das aplicações integradas ao ambiente.

O uso de software livre e a reutilização de software foram as estratégias tomadas para a escolha das tecnologias e ferramentas utilizadas no desenvolvimento e modelagem do ambiente proposto, sendo cada uma delas apresentadas e justificadas a seguir.

Titan Framework

O Titan [13] [12] [49] é um *framework* para instanciação de CMSs (*Content Management Systems*), ou seja, gerenciadores de conteúdo, para sistemas Web. A proposta do Titan Framework é ser uma solução simples e completa para a instanciação de gerenciadores de conteúdo de forma rápida e fácil. Entende-se por gerenciador de conteúdos uma aplicação que tem por característica fundamental prover uma interface para gerenciamento de dados de um determinado sistema. Este gerenciamento inclui a criação, recuperação, edição e exclusão (CRUD[5]) de dados em SGBDs ou arquivos do sistema. Um gerenciador de conteúdo instanciado pelo Titan Framework também provê um sistema de segurança e uma auditoria (sistema de log) para as ações dos usuários sobre estes dados.

Entre os vários recursos do Titan estão a arquitetura baseada em componentes, suporte a XML/XSL *Transformation*, independência de Banco de Dados por meio da camada de persistência, implementação totalmente Orientada a Objetos, suporte completo a AJAX (*Asynchronous Javascript and XML*) e suporte a LDAP.

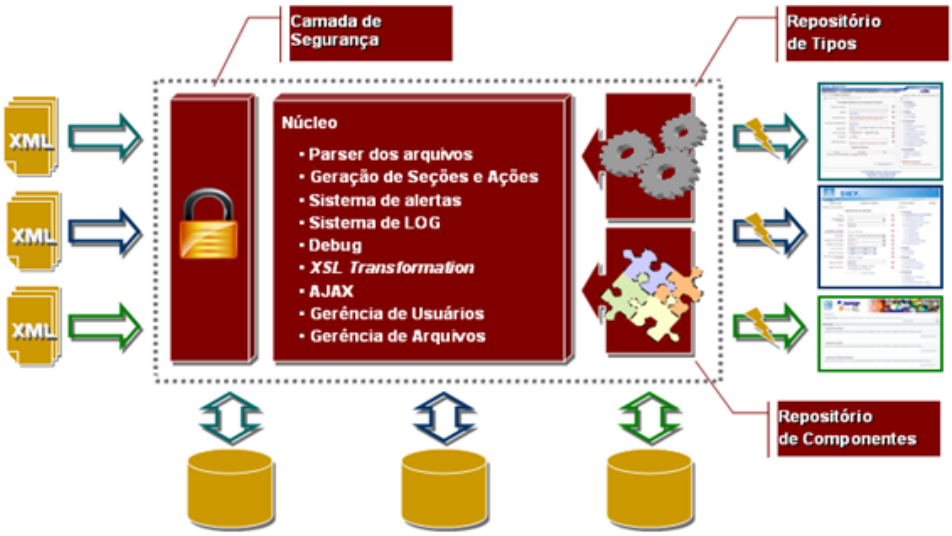


Figura 3.2: Arquitetura do Titan Framework [13].

O Titan possui uma arquitetura, ilustrada na Figura 3.2, que recebe como entrada uma linguagem de marcação (XML [81]) e a transforma, em tempo de execução, em um geren-

[5]CRUD é o acrônimo da expressão em língua Inglesa *Create*, *Retrieve*, *Update* e *Delete*, usada para definir quatro operações básicas utilizadas em bancos de dados relacionais ou em interface para usuários para criação, consulta, atualização e destruição de dados.

ciador de conteúdo. A arquitetura do Titan possui como característica ser Caixa-Branca, podendo ser flexível e extensível para outros domínios. A arquitetura do framework é formada por um núcleo e um repositório. Este núcleo, ou *core*, tem por característica ser imutável independente da configuração da instância. Assim, apenas um núcleo é necessário para executar todas as aplicações instanciadas[6] pelo framework em um mesmo servidor. Desta forma, se uma determinada funcionalidade não pode ser instanciada pelo framework é possível criar um novo componente para a respectiva funcionalidade. A linguagem genérica de entrada se encarregará de garantir que a configuração deste novo componente possa ser realizada como nos demais e que funcione de maneira harmônica com os outros componentes. Já o repositório é composto por tipos, *templates* e componentes que podem ser estendidos do framework para a instanciação de aplicações.

O uso do Titan Framework na construção do ambiente proposto deve-se ao fato dele ser um software livre de fácil extensão possibilitando a criação de forma simples de novos componentes, além de possuir suporte às tecnologias Web Service e LDAP. Com isto, é possível instanciar sistemas gerenciadores de conteúdo e customizá-los de acordo com as necessidades.

Outras ferramentas para instanciação de gerenciadores de conteúdo poderiam ser utilizadas para construção do ambiente integrador, como por exemplo: Joomla[7], Wordpress[8], Moodle[9] e Drupal[10], já que todas estas ferramentas utilizam tecnologias que suportam o uso de Web Services e LDAP.

Web Service

Conforme observado no Capítulo 2, a tecnologia Web Service possibilita a integração de aplicações web distintas, acessíveis pela rede, por meio da troca de mensagens baseada em XML, utilizando um protocolo padrão de comunicação entre as aplicações.

Como o Titan Framework foi desenvolvido em PHP então utilizou-se a implementação SOAP[11] de Web Service fornecida pelo pacote de instalação da própria linguagem PHP. Esta implementação de Web Service em PHP foi escolhida por ser simples e fornecer os mecanismos necessários para a construção do ambiente de integração. Além disso, outro ponto que contribuiu para a escolha de SOAP foi que as aplicações Web são legadas e neste caso é mais simples realizar as adaptações nelas utilizando SOAP do que XML-RPC[12], por exemplo.

[6] Tutorial para instanciação de aplicações no Titan Framework disponível em http://wiki.ledes.net/index.php/Titan_Framework, acessado em: 03/2010.
[7] http://www.joomla.org/
[8] http://br.wordpress.org/
[9] http://www.moodle.org/
[10] http://www.drupal.org/
[11] http://www.php.net/manual/pt_BR/book.soap.php
[12] http://www.php.net/manual/pt_BR/book.xmlrpc.php

LDAP

O LDAP [2], do inglês *Lightweight Directory Access Protocol*, ou Protocolo Leve de Acesso a Diretórios é um protocolo de simples implementação que oferece facilidade e leveza no acesso aos serviços de diretórios, capaz de se integrar, de forma centralizada, a diversos serviços de rede.

O LDAP define um protocolo de troca de mensagens usado por clientes/servidores de diretórios. Assim, o cliente envia uma requisição de informações ao servidor que a processa e realiza as operações necessárias em seu diretório de busca. Com isso, é devolvida uma resposta contendo as informações solicitadas ou erros para a requisição desse cliente.

Por ser leve, rápido e de simples implementação, o LDAP não é considerado somente um protocolo, mas sim um diretório de serviço escalonável, tornando-o completamente aceito como o padrão Internet para serviços de diretórios que executam sobre a arquitetura TCP/IP.

Por essa razão e, em função de suas características, principalmente por ser padrão aberto, o protocolo LDAP oferece suporte a diversos serviços, tais como: aplicações web, Intranets, servidores IMAP (*Internet Message Access Protocol*) e bancos de dados, conforme ilustrado na Figura 3.3. Trigo [71] ressalta ainda a integração do LDAP com outros serviços, complementando a infraestrutura de redes, fornecendo novos recursos e, especialmente, maior integração, diferentemente de outros protocolos e linguagens estabelecidos como, SNMP (*Simple Network Management Protocol*), HTTP, SMTP (*Simple Mail Transfer Protocol*) e IMAP, conforme ilustrado na Figura 3.3.

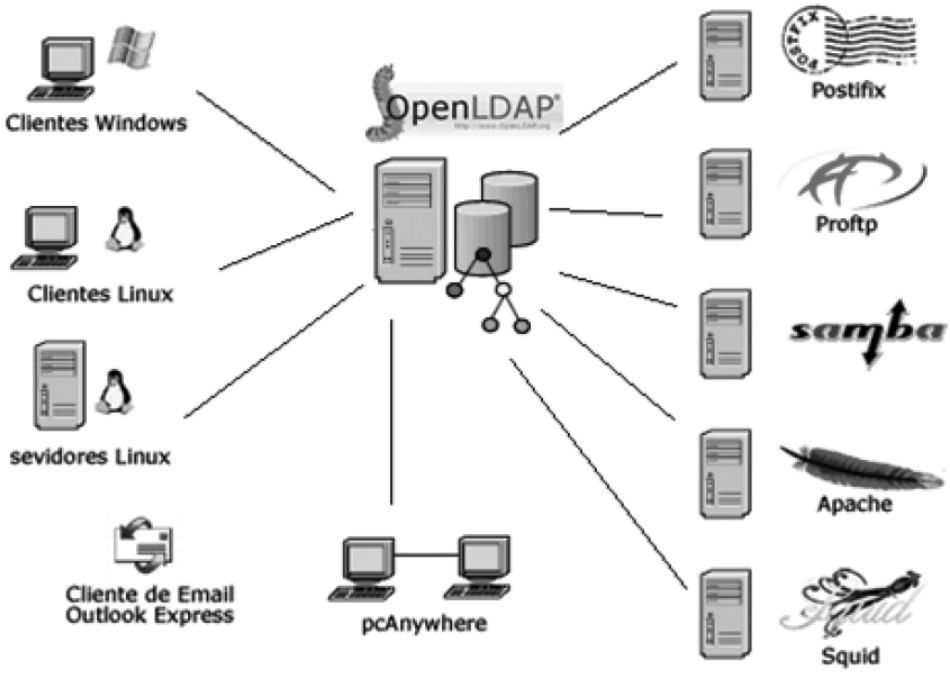


Figura 3.3: Utilização do LDAP como base para diversos serviços na rede [2].

Segundo Trigo [71] a flexibilidade do LDAP se dá em função da organização das

informações de maneira hierárquica, possuindo um elemento raiz - local por onde as buscas se iniciam - e nós filhos que são percorridos até que o elemento desejado seja encontrado.

Nesse sentido, a raiz e os nós são representações de diretórios, sendo que cada um pode conter outros diretórios ou elementos denominados entradas. Cada entrada pode conter um ou mais atributos que farão referência a um ou mais valores associados a eles, de acordo com o tipo de dado.

Como exemplo, para diretórios, tem-se os atributos apresentados na Tabela 3.6 e para entradas, tem-se os atributos apresentados na Tabela 3.7.

Tabela 3.6: Lista de atributos para diretórios de uma base LDAP. [71]

Atributo	**Descrição**
c	Para diretórios que representam países (do inglês *country*).
o	Para o nome da empresa (do inglês *organization*).
ou	Para departamentos (do inglês *organization unit*).

Tabela 3.7: Lista de atributos para entradas de uma base LDAP. [71]

Atributo	**Descrição**
cn	Como atributo de nome (do inglês *common name*).
dc	Como atributo de Componente de Domínio (do inglês *Domain Component*).
uid	Para identidade de usuários (do inglês *user* ID).
gn	Para nome próprio de uma pessoa (do inglês *given name*).
sn	Para o sobrenome de uma pessoa (do inglês *surname*).

Devido as características apresentadas, a tecnologia LDAP foi utilizada no AIA para fornecer um mecanismo de integração entre os dados dos usuários presentes no AIA e das aplicações a serem integradas nela. Para instanciar a base LDAP responsável pela centralização das insformações dos usuários, foi utilizado o OpenLDAP [52] [71], que é uma implementação *open source* do protocolo LDAP com versões para alguns sistemas operacionais, tais como: Windows[13], Linux[14], MAC OS X[15] e FreeBSD[16], tornando assim o ambiente proposto ainda mais abrangente.

3.3.2 Arquitetura do Ambiente Integrador de Aplicações

O ambiente integrador de aplicações é uma instância do Titan Framework que possui todas as funcionalidades básicas que o Titan fornece mais as funcionalidades definidas pelo "Componente Integrador de Aplicações" que foi desenvolvido neste trabalho e será apresentado a seguir. Além disso, o ambiente também possui uma base de dados relacional

[13]http://windows.microsoft.com/
[14]http://www.linuxfoundation.org/
[15]http://www.apple.com/br/macosx/
[16]http://www.freebsd.org/

para armazenamento das informações das aplicações legadas integradas a ele e um banco de diretórios LDAP para armazenamento de informações dos usuários do ambiente e das aplicações integradas a ele.

Para uma melhor compreensão do que é o ambiente integrador de aplicações proposto neste trabalho, foi projetada a arquitetura ilustrada na Figura 3.4. Ela é composta por um componente, intitulado "Componente Integrador de WebApps", que foi construído a partir do reuso de componentes e tipos de dados do Titan Framework. Este componente é responsável pela gerência (cadastro, exclusão, edição e acesso) das aplicações legadas integradas à instância do Titan. Sendo que as informações das aplicações são armazenadas em uma base de dados exclusiva da instância do Titan. Foi desenvolvido também um web service que é responsável por autenticar o usuário, que está atualmente logado na instância, nas aplicações que forem integradas a ela, quando este solicitar acesso a tais aplicações. Sendo que este processo de autenticação utiliza uma base LDAP que contém dados específicos de todos os usuários da instância do Titan. Além disso, cada *webapp* integrada à instância pode ter um banco de dados exclusivo ou compartilhado com outras webapps integradas à instância.

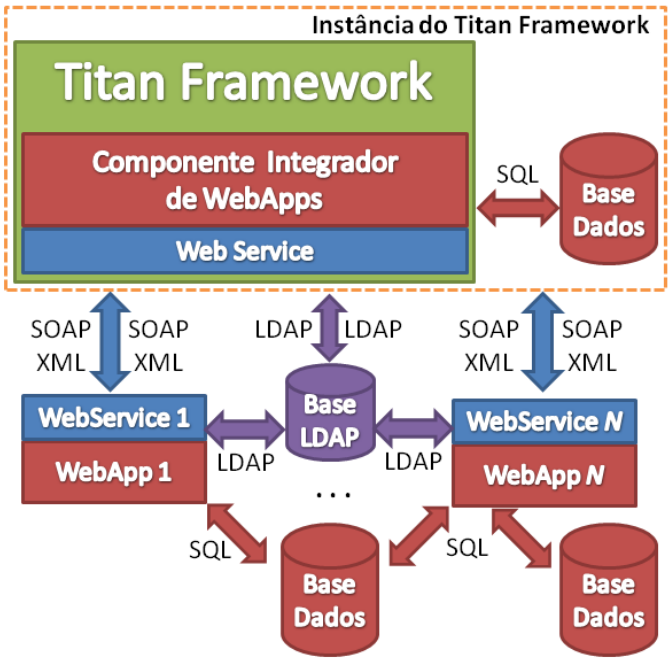


Figura 3.4: Arquitetura do AIA.

A base LDAP projetada para o AIA, ilustrada na Figura 3.5, é representada como uma árvore e possui o elemento "dc = ledes, dc = net" como raiz, que representa o domínio principal da base LDAP. A raiz por sua vez possui três filhos "cn = admin", "ou = groups" e "ou = users". O primeiro representa a entrada para o usuário administrador da base LDAP. O segundo representa o diretório que agrega os grupos de usuários possuindo como filho uma única entrada "cn = gestores" que representa o grupo de usuários gestores da instância do Titan. O terceiro representa o diretório que agrega os usuários da base LDAP possuindo como filhos elementos indexados pela entrada *uid*, no caso o *login* dos usuários da instância.

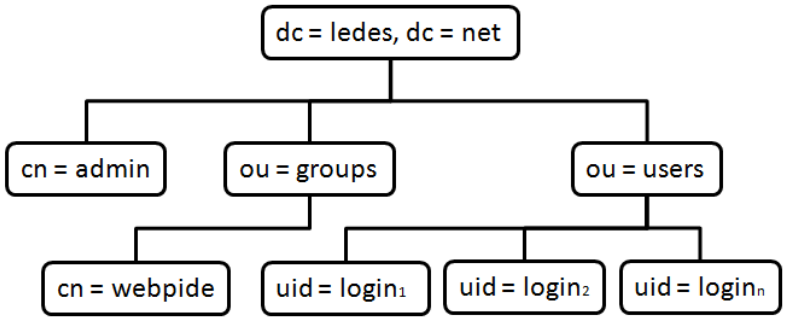


Figura 3.5: Estrutura hieráquica da base LDAP que compõe o AIA.

Para o desenvolvimento do AIA foi utilizada a estratégia de desenvolvimento de serviços *meet-in-the-middle* que define uma série de procedimentos para a construção eficiente de aplicações orientadas a serviço. Foi escolhida esta abordagem para a construção do ambiente de integração pois ela abrange outras duas abordagens, *top-down* e *bottom-up*. Sendo que a *top-down* foi útil para a construção do componente integrador de webapps e a *bottom-up* útil para o processo de adaptação das aplicações web legadas integradas à instância do Titan.

Com a execução do processo *meet-in-the-middle* no Titan Framework foi possível criar um componente genérico e um web service para integração de aplicações web. Sendo que eles podem ser utilizados em qualquer instância do Titan que possua uma base LDAP associada e estruturada conforme a Figura 3.5.

A seguir, são apresentados demais artefatos produzidos durante a construção do AIA.

A Figura 3.6 ilustra o Diagrama de Caso de Uso do AIA. Nela é possível identificar as oito funcionalidades que o AIA fornece, sendo cada uma delas descritas a seguir:

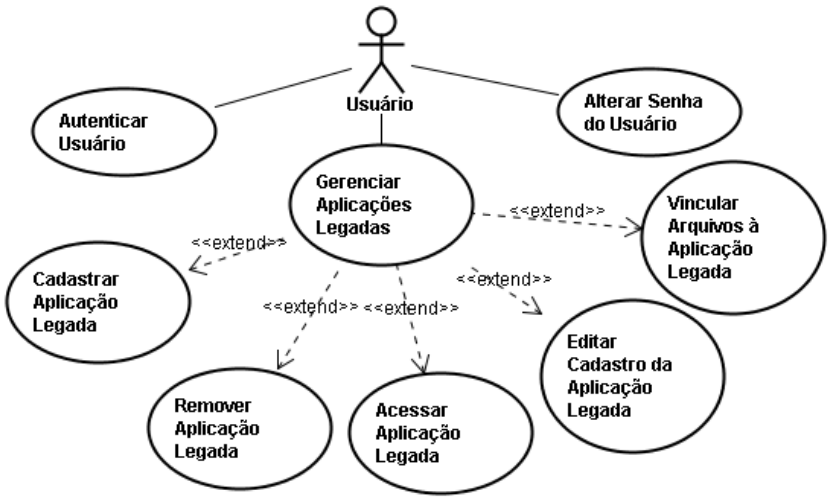


Figura 3.6: Diagrama de Casos de Uso do AIA.

- **Autenticar Usuário**: responsável por autenticar o usuário na instância do Titan utilizando para isso o mecanismo de autenticação do próprio Titan adaptado para

autenticar os usuários em uma base LDAP, já que o Titan possui suporte a tecnologia LDAP.

- **Alterar Senha do Usuário**: responsável por permitir a troca da senha do usuário atualmente logado na instância do Titan, senha esta que é atualizada na base LDAP da instância.
- **Gerenciar Aplicações Legadas**: responsável pela gerência das aplicações legadas que estão ou que serão integradas à instância do Titan.
- **Cadastrar Aplicação Legada**: responsável pelo cadastro dos dados gerais, de autenticação e de acesso à aplicação legada que será integrada à instância do Titan.
- **Remover Aplicação Legada**: responsável pela exclusão do cadastro da aplicação legada na instância do Titan.
- **Editar Cadastro da Aplicação Legada**: responsável pela edição do cadastro da aplicação legada na instância do Titan.
- **Acessar Aplicação Legada**: responsável pelo processo de autenticação e acesso à aplicação web legada integrada na instância do Titan utilizando para isto o web service do componente integrador de webapps e o web service da instância, além da base LDAP da instância do Titan.
- **Vincular Arquivos à Aplicação Legada**: responsável pelo vínculo de arquivos (manuais de uso e instalação da aplicação, relatórios técnicos e outras documentações sobre a aplicação) à instância do Titan.

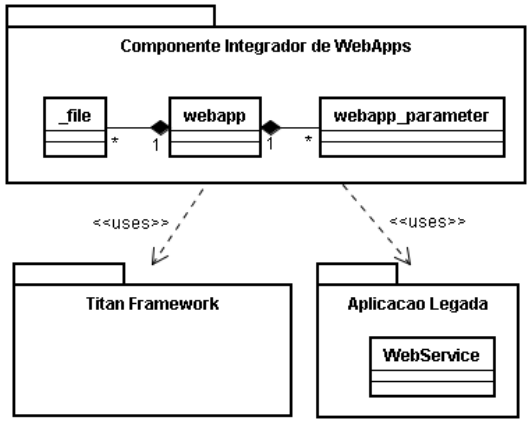


Figura 3.7: Diagrama de Classes do AIA.

A Figura 3.7 ilustra o Diagrama de Classes do ambiente proposto. Nela é possível identificar três pacotes de classes `Componente Integrador de WebApss`, `Titan Framework` e `Aplicacao Legada`. Sendo que o pacote `Componente Integrador de WebApps` possui três classes que se relacionam da seguinte forma: a classe `App` armazena informações sobre a aplicação integrada à instância do Titan. Ela é responsável por invocar alguns métodos de classes presentes no pacote `Titan Framework` e invocar também o serviço web (método

da classe `WebService` pertencente ao pacote `Aplicacao Remota`) que é responsável pela autenticação remota do usuário (que está logado na instância do Titan) na aplicação legada; a classe `_file` armazena informações sobre os arquivos (manuais de uso e instalação da aplicação legada, por exemplo) vinculados a uma aplicação legada (objeto da classe `webapp`); e a classe `webapp_parameter` armazena informações quanto aos parâmetros de acesso associados a uma aplicação legada (objeto da classe `webapp`).

Os pacotes `Titan Framework` e `Aplicacao Legada` representam o conjunto de classes e métodos presentes, respectivamente, no Titan Framework e na aplicação legada que está integrada à instância do Titan, com o devido destaque para a classe `WebService` do pacote `Aplicacao Legada` que é responsável por fornecer o mecanismo de autenticação remota na aplicação legada.

A Figura 3.8 ilustra a modelagem do banco de dados do componente integrador de webapps, que possui quatro relações: `webapp`, `webapp_parameter`, `_file` e `webapp_file`. A relação `webapp` é responsável por manter o cadastro da aplicação no banco de dados da instância do Titan. A relação `_file` é responsável por manter todos os arquivos anexados na instância do Titan sendo que no contexto do ambiente proposto ela é utilizada para manter os anexos (documentos, manuais, relatórios técnicos, etc) das aplicações legadas integradas, além da imagem representando o *logo* da aplicação, que é definida na relação `webapp`. A relação `webapp_file` é responsável por manter o vínculo entre os arquivos e as aplicações. Esta relação representa o relacionamento *n..n* das relações `webapp` e `_file`. Por fim, a relação `webapp_parameter` é responsável por manter os parâmetros (login do usuário autenticado na instância, senha do usuário autenticado na instância ou hash retornada pelo web service da aplicação), criptografados (utilizando algoritmos MD5 ou SHA1) ou não, que são enviados por GET ou POST por meio da Web para realizar a autenticação e acesso à aplicação remota.

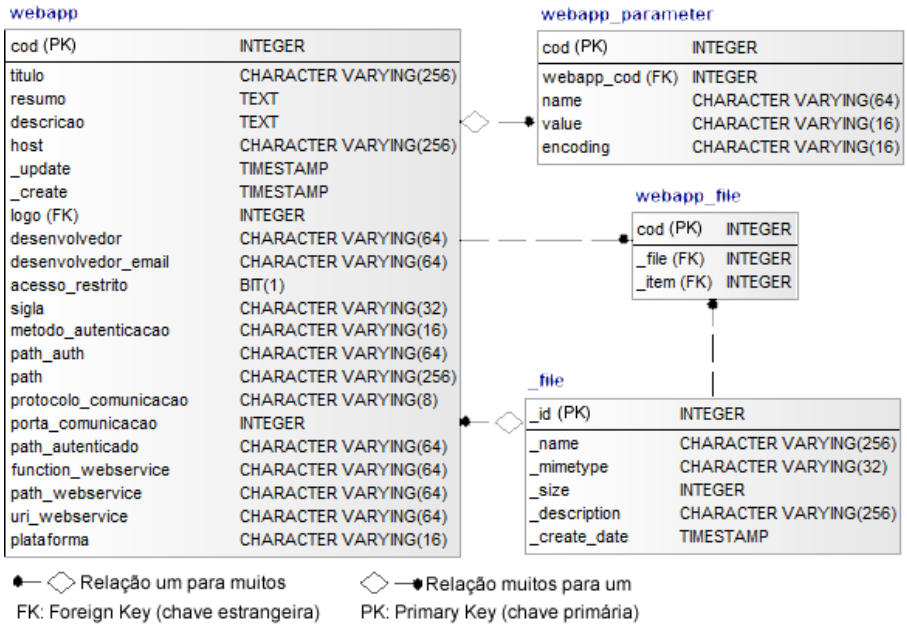


Figura 3.8: Modelagem do banco de dados do AIA.

Tabela 3.8: Descrição dos campos da relação `webapp`.

Campo	**Descrição**
desenvolvedor	Nome do responsável pelo desenvolvimento da aplicação.
desenvolvedor_email	Email do responsável pela aplicação.
plataforma	Tipo de Plataforma: *Desktop* ou *Web*.
logo	Imagem representando a *logo* da aplicação. Arquivo referenciado da relação `_file`.
titulo	Nome da aplicação.
sigla	Sigla da aplicação.
resumo	Descrição resumida das principais características e funcionalidades da aplicação.
descricao	Descrição detalhada de como funciona e quais funcionalidades estão presentes na aplicação.
protocolo_ comunicacao	Protocolo de comunicação utilizado para acesso à aplicação. Somente o protocolo HTTP está disponível entretanto futuramente outros protocolos podem ser utilizados como, FTP e HTTPS.
host	Nome ou IP do *host* que hospeda a aplicação.
porta_comunicacao	Porta de comunicação com a aplicação.
path	Caminho da aplicação, relativo ao host, no diretório de arquivos do servidor que hospeda a aplicação.
acesso_restrito	Indica se a aplicação possui acesso restrito ou não. O restante dos campos não precisam ser informados, no caso da aplicação possuir apenas acesso público.
path_auth	Caminho para autenticação na aplicação, relativo ao host, sem os parâmetros de autenticação.
path_autenticado	Caminho na aplicação, relativo ao host, para o acesso de usuário já autenticado na aplicação.
metodo_ autenticacao	Método de autenticação e acesso à aplicação, ou seja, forma como os parâmetros de autenticação e acesso são enviados à aplicação. Pode assumir os seguintes valores: **_GET_** (parâmetros enviados por GET), **_POST_** (parâmetros enviados por POST) e **_WEBSERVICE_** (parâmetros enviados por meio do Web service da aplicação).
path_webservice	Caminho completo para o Web Service de autenticação da aplicação incluindo protocolo HTTP e porta de comunicação (caso seja diferente de 80), mas sem os parâmetros de autenticação.
uri_webservice	URI completo para o Web Service de autenticação da aplicação incluindo protocolo HTTP e porta de comunicação (caso seja diferente de 80), mas sem os parâmetros de autenticação.
function_webservice	Nome da função ou método, presente no web service da aplicação, responsável pela autenticação do usuário na aplicação.

Tabela 3.9: Descrição dos campos da relação `webapp_parameter`.

Campo	Descrição
name	Nome do parâmetro enviado à aplicação.
value	Valor do parâmetro enviado à aplicação. Pode assumir os seguintes valores: _**LOGIN**_ (Login do usuário autenticado na instância do Titan), _**PASSWORD**_ (Senha do usuário autenticado na instância do Titan) ou _**HASH**_ (*Hash* randômica e temporária enviada pelo Web Service da Aplicação).
encoding	Indica se o valor do parâmetro será criptografado ou não. Pode assumir os seguintes valores: _**SHA1**_ (Criptografado utilizando o algoritmo SHA1), _**MD5**_ (Criptografado utilizando o algoritmo MD5) ou **NULL** (Não será criptografado).

As Tabelas 3.8 e 3.9 apresentam detalhadamente cada um dos campos das relações `webapp` e `webapp_parameter`, respectivamente, ilustradas na Figura 3.8, pois estas são as principais relações do componente integrador de webapps proposto neste trabalho, ressaltando-se que o campo `protocolo_comunicacao` foi mantido na relação `webapp` para armazenar o protocolo HTTP mas que futuramente podem ser incorporados outros protocolos ao AIA, como por exemplo: FTP e HTTPS.

Para compreender melhor o processo de autenticação e acesso à uma aplicação Web que está integrada à instância do Titan foi definido o diagrama de atividades, representado na Figura 3.9, para o caso de uso "Acessar Aplicação". Nela é possível identificar 4 atores ou entidades que se interagem e 7 atividades cada uma rotulada com uma numeração crescente para indicar a organizar a execução das atividades.

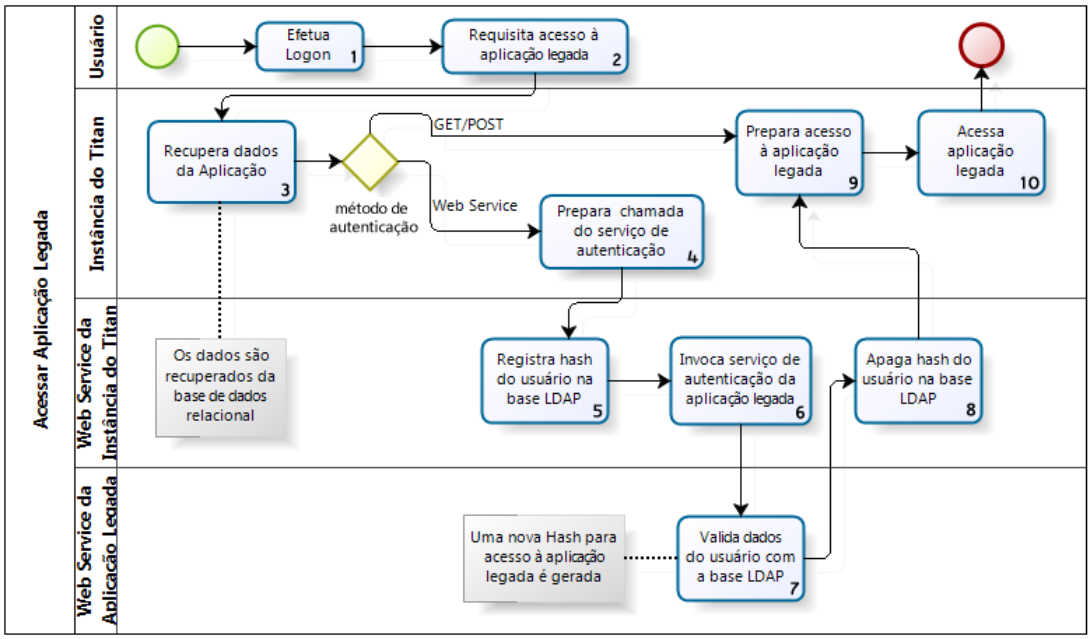


Figura 3.9: Diagrama de atividades do caso de uso "Acessar Aplicação".

A sequência inicia com a autenticação (1) do usuário na instância do Titan que verifica se as informações estão corretas quanto ao *login* e senha informados pelo usuário e submetidos à base LDAP da instância do Titan. Caso o login e senha estejam corretos o usuário é então autenticado na instância. Em seguida, o usuário requisita acesso (2) a uma aplicação legada que esteja cadastrada na instância. Neste momento, a instância do Titan recupera (3) as informações cadastradas em seu banco de dados quanto à aplicação legada. Caso o método de autenticação definido para a aplicação seja do tipo **_GET_** ou **_POST_** a instância do Titan prepara (9) o acesso à aplicação legada e, em seguida, efetua o acesso (10) à aplicação utilizando os parâmetros definidos no cadastro da aplicação (note que neste caso são enviados apenas o login e a senha do usuário, criptografados ou não em MD5 ou SHA). Caso contrário, com base nas informações da aplicação, o sistema prepara a execução (4) do serviço remoto de autenticação, neste caso o método de autenticação escolhido é **_Web Service_**. Na sequência, o WS da instância gera e registra (5) uma *hash* randômica na base LDAP da instância associado ao login do usuário que solicita acesso à aplicação legada. Em seguida, o WS da instância invoca (6) o serviço remoto de autenticação da aplicação definido pelo campo ***function_webservice*** do cadatro da aplicação com os seguintes parâmetros: *login* do usuário e uma nova *hash* composta pela criptografia em MD5 do *login* e *hash* randômica gerada anteriormente. O WS da aplicação então valida (7) os dados submetidos com os dados presentes na base LDAP da instância e retorna um padrão de resposta composto pelos seguintes campos:

- **STATUS**: valor 1 indicando que a autenticação ocorreu com sucesso ou 0 caso contrário;
- **MESSAGE**: texto informando motivo do erro no processo de autenticação;
- **HASH**: nova *hash* randômica e temporária que dará acesso autenticado à aplicação.

Neste momento, o WS da instância apaga (8) a *hash* que foi registrada por ele na base LDAP para garantir que não ocorram acessos indevidos. Em seguida, o WS da instância analisa a resposta enviada pelo WS da aplicação e caso a resposta seja inválida então o acesso à aplicação é finalizado informando uma mensagem de erro para o usuário, caso contrário inicia-se a preparação (9) para o acesso à aplicação. Por fim, utilizando a *hash* enviada pelo WS da aplicação e o *login* do usuário o sistema efetiva o acesso (10) autenticado do usuário na aplicação (note que neste caso não deve ser enviado a senha do usuário, mas sim a hash gerada pelo web service da aplicação), que consiste na aplicação remota verificar se a *hash* ainda está válida e permitir ou não o acesso do usuário à aplicação.

3.4 Considerações Finais

Neste capítulo foram apresentados, o processo para integração baseado em serviços de aplicações web legadas, PIBSAWL e o ambiente integrador de aplicações, AIA. Sendo que o PIBSAWL pretende servir como um guia na integração de aplicações Web à instâncias do Titan. Já o AIA fornece os mecanismos necessários para concretizar tal integração. O

estudo de caso da integração do SIGFAP ao portal Web-PIDE detalhado no Capítulo 5 exemplifica e avalia o uso do processo PIBSAWL definido neste capítulo.

Durante a construção do AIA foram utilizadas algumas tecnologias para solucionar algumas necessidades, entretanto foi constatado que poderiam ser utilizados outras tecnologias, tornando assim o AIA uma proposta mais flexível que pode ser replicada para outros domínios que utilizam tais tecnologias em seus ambientes. A Tabela 3.10 agrega as possíveis tecnologias que poderiam ser utilizadas para atender as necessidades principais para construção do AIA. Ressaltando que a implementação Web Service deve necessariamente estar associada a linguagem de programação utilizada pelo framework de instanciação de gerenciadores de conteúdo durante o desenvolvimento do AIA.

Tabela 3.10: Demais tecnologias possíveis para construção do AIA.

Necessidades	**Possíveis Tecnologias**
Framework para instanciação de gerenciadores de conteúdo	Titan Framework, Joomla, Wordpress, Moodle e Drupal
Linguagem de programação	PHP, JSP[17], ASP.NET[18]
Solução em SGBD	PostgreSQL, MySQL, SQLite[19], MS SQL[20]
Solução em LDAP	OpenLDAP e Active Directory Lightweight Directory Services[21]

O ambiente de integração desenvolvido neste trabalho não permite o acesso à aplicações *desktop* pois uma instância do Titan Framework possui plataforma *Web* e não foi considerado este tipo de integração neste trabalho. Entretanto, o ambiente possibilita o registro de tais aplicações para que em um trabalho futuro possam ser aproveitadas como estudos de caso estendendo o processo PIBSAWL para possibilitar a integração de aplicações *desktop* legadas e não somente aplicações *web* legadas.

É importante ressaltar que tanto o processo PIBSAWL quanto o AIA propostos neste trabalho não levaram em consideração a definição de níveis de acesso nas aplicações integradas, foi considerado apenas a autenticação. Portanto, quando um usuário logado no ambiente integrador requisita acesso a uma aplicação, a própria aplicação deve tomar os devidos cuidados no mecanismo de acesso para garantir quais serão os níveis de acesso do usuário na aplicação e isto é feito automaticamente pela própria aplicação após o acesso do usuário a ela.

Capítulo 4

Estudos de Caso

4.1 Considerações Iniciais

Neste capítulo são apresentados dois estudos de caso para avaliar o PIBSAWL e o AIA, propostos neste trabalho. Para realizar tais estudos foram avaliadas as aplicações desenvolvidas pela equipe do projeto Web-PIDE de acordo com critérios definidos no Apêndice A) sendo por fim, escolhida a aplicação Pentaho/Web-PIDE como primeiro estudo de caso. Além disso, para avaliar o processo PIBSAWL fora do domínio educacional (que é o fodo do projeto Web-PIDE) e como uma abordagem genérica (já que o AIA pode ser uma instância qualquer do Titan Framework), foi escolhida, como segundo estudo de caso, a aplicação SIGFAP, que é uma aplicação desenvolvida pela equipe de Engenharia de Software do Laboratório de Engenharia de Software (LEDES)[1] da UFMS.

Os motivos para escolha do Pentaho/Web-PIDE foram o suporte à tecnologia LDAP e também por ser uma aplicação web de acesso restrito e de código aberto. Já o SIGFAP foi escolhido por ser uma aplicação web de acesso restrito, pela experiência do candidato no desenvolvimento desta aplicação e pela facilidade de acesso ao código fonte da aplicação pois o candidato compõe a equipe de desenvolvimento do SIGFAP.

Como o processo PIBSAWL é apoiado pelo AIA, então foi adaptado o portal Web-PIDE (instância do Titan Framework e principal ferramenta de acesso à plataforma Web-PIDE). Ele sofreu as adequações necessárias, conforme orientações da Seção 4.3 do Capítulo 4 incorporando principalmente um novo componente, intitulado "Componente Integrador de WebApps", uma base LDAP e novas relações em seu banco de dados. Com isto, o Portal Web-PIDE foi utilizado como AIA nos estudos de caso, entretanto, outras instâncias do Titan Framework poderiam ser utilizadas como AIA, desde que possuam a arquitetura definida na Seção 4.3 do Capítulo 4.

A seguir, são descritos os estudos de caso da integração do Pentaho/Web-PIDE e do SIGFAP ao Portal Web-PIDE.

[1] http://portal.ledes.net/

4.2 Estudo 1: Integração do Pentaho/Web-PIDE ao Portal Web-PIDE

O Pentaho/Web-PIDE é uma das aplicações desenvolvidas pela equipe do projeto Web-PIDE e possibilita realizar consultas tabulares e gráficas no DW do CEB. Ela foi desenvolvida com tecnologias Java, Mondrian e Pentaho BI.

A seguir é apresentado o processo de integração do Pentaho/Web-PIDE ao portal Web-PIDE, utilizando para isso o PIBSAWL e o AIA no estudo de caso.

Ao executar a atividade 1 do PIBSAWL foram recuperados, o código fonte da aplicação em Java, um relatório técnico sobre o Pentaho/Web-PIDE [47], contendo diversos artefatos sobre a aplicação e uma comunidade *online*[2] sobre a principal tecnologia utilizada pela aplicação, Pentaho BI. Além disso, foi recuperado um manual [28] do processo de instalação e utilização do Pentaho BI. Como foi possível instalar e executar em um ambiente próprio de desenvolvimento a aplicação, seguindo orientações dos artefatos recuperados, prosseguiu-se para a atividade 2 do PIBSAWL. Sendo que o contato com o desenvolvedor da aplicação ocorreu através de emails e reuniões *online* e presencial. Já o contato com a comunidade *online* ocorreu por meio da Internet através de pesquisas no sítio da comunidade. Estes contatos mantiveram-se durante a execução do restante do processo e foram muito importantes para conclusão do processo de integração da aplicação Pentaho/Web-PIDE ao Portal Web-PIDE.

Ao executar a atividade 2 constatou-se que a aplicação já possuía suporte à tecnologia LDAP, sendo necessário algumas configurações para que a aplicação pudesse autenticar seus usuários utilizando essa tecnologia. Em seguida, foi realizada a análise da arquitetura da aplicação na qual constatou-se que as alterações necessárias para a implementação da tecnologia Web Service na aplicação seriam bastante dispendiosas principalmente pela complexidade do Pentaho BI, da complexidade da linguagem de programação Java e da pouca documentação técnica quanto ao Pentaho BI. Nesta atividade foi projetada para a aplicação a arquitetura Web não adaptável a SOA, ilustrada na Figura 4.1. Nessa arquitetura, tanto o Pentaho-Web-PIDE quanto o Portal Web-PIDE se comunicam através do protocolo LDAP com a base LDAP/Web-PIDE e entre si através de requisições GET utilizando o protocolo HTTP. Com isso, prosseguiu-se para a atividade 4 do PIBSAWL.

Na atividade 4 foram adaptados os mecanismos de autenticação e acesso da aplicação, sendo que os usuários já existentes da aplicação foram transferidos para a base LDAP e o mecanismo padrão de autenticação e acesso da aplicação foi adaptado e não replicado, logo todo usuário do Pentaho/Web-PIDE passou a autenticar-se na base LDAP e não mais em sua própria base de dados. Nesta atividade foram escolhidos os testes de unidade com Ids: 9, 10, 11, 12, 13, 14, 15 e 16, presentes no gabarito, definido na Tabela 3.2, para serem executados na atividade 5, já que na aplicação não foi possível implantar o serviço de autenticação através de um web service. Em seguida, prosseguiu-se para a atividade 5 do PIBSAWL.

Na atividade 5 foram realizados os testes de unidade, escolhidos na atividade 4,

[2]http://community.pentaho.com/

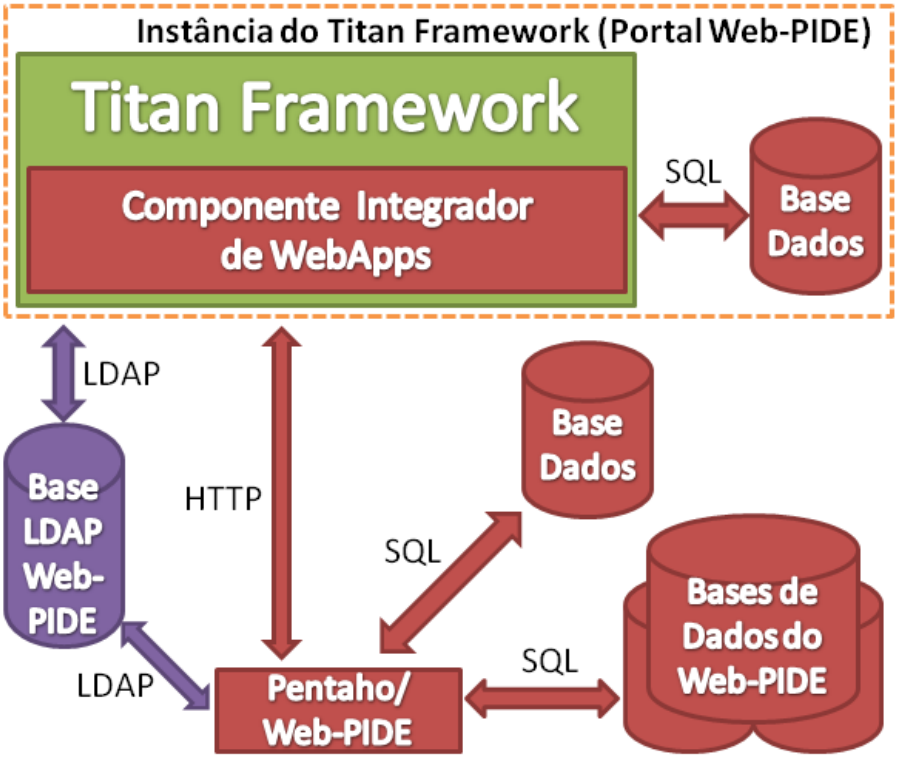


Figura 4.1: Arquitetura projetada para a integração do Pentaho/Web-PIDE ao Portal Web-PIDE.

gerando-se o artefato ilustrado na Figura 4.1. Todos os testes de unidade realizados foram bem sucedidos e o Pentaho/Web-PIDE continuou funcional, com isto, prosseguiu-se para a atividade 6 do PIBSAWL.

Tabela 4.1: Relatório de execução dos casos de teste de unidade do Pentaho/Web-PIDE.

Id. Caso de Teste de Unidade	**Resultado da Execução**
CT_9	Correto
CT_{10}	Correto
CT_{11}	Correto
CT_{12}	Correto
CT_{13}	Correto
CT_{14}	Correto
CT_{15}	Correto
CT_{16}	Correto

Na atividade 6, o Pentaho/Web-PIDE foi registrado no Portal Web-PIDE, já que o Pentaho/Web-PIDE fornecia um modo de autenticação por GET no qual poderiam ser transmitidos o *login* e a senha do usuário para a aplicação. As informações registradas no portal podem ser conferidas na Figura 4.2. Foram anexados também ao registro do Pentaho/Web-PIDE os manuais de uso da aplicação, recuperados na atividade 1. Apesar do método de autenticação disponibilizado pelo Pentaho/Web-PIDE não ser tão seguro

(já que ele recebe por GET os dados do usuário em texto puro e estes dados podem ser intercepetados por outras aplicações na rede), a aplicação foi registrada no Portal Web-PIDE e utilizada apenas para fins de estudo de caso. Em seguida, foi elaborado o "Documento de Casos de Teste de Integração", onde foram escolhidos os casos de teste de integração com Ids: 3, 8, 9 e 10, presentes no gabarito, definido na Tabela 3.4. Por fim, prosseguiu-se para a atividade 7 do PIBSAWL.

Na atividade 7 foram realizados os testes definidos na atividade 6, verificando se o usuário atualmente logado no portal Web-PIDE obtinha acesso autenticado no Pentaho/Web-PIDE, além de utilizar a aplicação sem problemas. Isto ocorreu com sucesso, conforme observado nas telas do Portal Web-PIDE e Pentaho/Web-PIDE ilustradas na Figura 4.3 e no artefato produzido nesta atividade, ilustrado na Tabela 4.2. Um fato interessante durante os testes foi que o Pentaho/Web-PIDE não exigiu que o usuário fosse registrado (com informações oriundas da base LDAP/Web-PIDE) em seu próprio banco de dados para obter acesso autenticado na aplicação, diferentemente do próximo estudo de caso, em que foi necessário realizar tal cadastro automatizado para que o usuário conquistasse tal acesso. Por fim, o usuário obteve acesso autenticado ao Pentaho/Web-PIDE a partir do Portal Web-PIDE e pôde utilizar a aplicação sem problemas então, prosseguiu-se para a atividade 8.

Tabela 4.2: Relatório de execução dos casos de teste de integração do Pentaho/Web-PIDE.

Id. Caso de Teste de Integração	**Resultado da Execução**
CT_3	Correto
CT_8	Correto
CT_9	Correto
CT_{10}	Correto

Na atividade 8, como a aplicação Pentaho/Web-PIDE estava funcional então ela foi implantada em um ambiente de produção e prosseguiu-se para a atividade 9.

Na atividade 9, o registro da aplicação Pentaho/Web-PIDE foi atualizado no portal Web-PIDE apontando-a para o ambiente de produção da aplicação Pentaho/Web-PIDE. Em seguida, prosseguiu-se para a atividade 10.

Por fim na atividade 10, foram realizados novos testes de integração similares aqueles executados na atividade 7 e a aplicação Pentaho/Web-PIDE foi, enfim, integrada ao Portal Web-PIDE. Com isso, a execução do PIBSAWL foi encerrada e verificou-se que mesmo aplicações não possuindo um método de autenticação por meio de Web Service podem ser integradas ao portal Web-PIDE, necessitando, de alguns ajustes, à nível de código fonte, para realizar a autenticação baseado nos dados da base LDAP/Web-PIDE e não mais em sua própria base de dados.

Quanto ao Desenvolvedor

Nome: Fernando Maia da Mota

Email: mota.fernandomaia@gmail.com

Quanto à Aplicação

Tipo de Plataforma: Web

Logo: 

pentaho_webpide.png
31108 Bytes
image/png

Título: WebPIDE - Pentaho

Sigla: WebPIDE - Pentaho

Resumo: O Pentaho/Web-PIDE é uma das aplicações desenvolvidas pela equipe do projeto Web-PIDE e possibilita realizar consultas tabulares e gráficas no DW do CEB e ela foi desenvolvida com tecnologias Java, Mondrian e Pentaho BI.

Descrição: O Pentaho/Web-PIDE é uma das aplicações desenvolvidas pela equipe do projeto Web-PIDE e possibilita realizar consultas tabulares e gráficas no DW do CEB e ela foi desenvolvida com tecnologias Java, Mondrian e Pentaho BI.

Quanto ao Acesso

Protocolo de Comunicação: HTTP

Host: maxwellubuntu

Porta: 8180

Path: /pentaho/

Possui Acesso Restrito: Sim

Path para autenticação: j_spring_security_check

Path usuário autenticado:

Método de Autenticação: GET

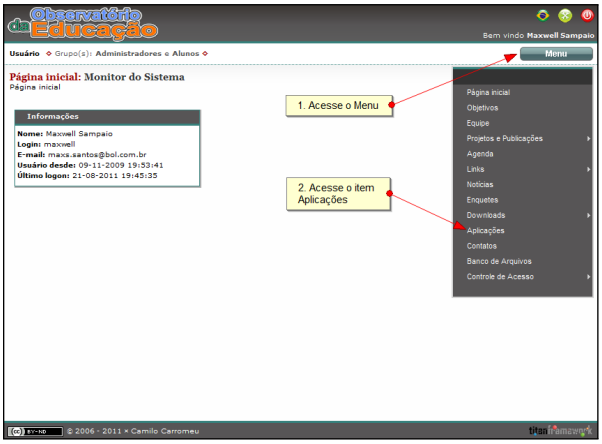
Quanto aos Parâmetros de Acesso

Lista de Parâmetros de Acesso:

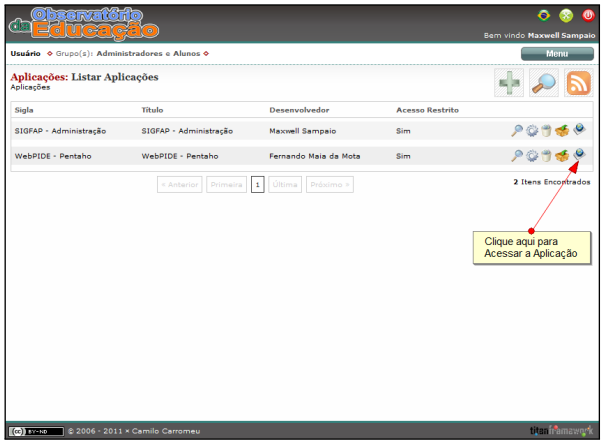
Name	Valor	Codificação
j_password	Senha do Usuário Atualmente Logado no Sistema	
j_username	Login do Usuário Atualmente Logado no Sistema	

Figura 4.2: Informações do Pentaho/Web-PIDE cadastradas no Portal Web-PIDE.

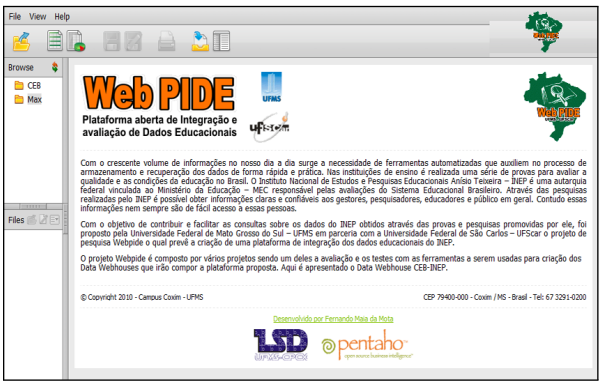
(a) Logando no Portal Web-PIDE.



(b) Listando as aplicações integradas ao Portal.



(c) Acessando o Pentaho/Web-PIDE pelo Portal.



(d) Executando o Pentaho/Web-PIDE.

Figura 4.3: Telas da execução do teste de integração do Pentaho/Web-PIDE ao Portal Web-PIDE.

4.3 Estudo 2: Integração do SIGFAP ao Portal Web-PIDE

O SIGFAP [73] ou Sistema de Informação e Gestão de Projetos das Fundações de Amparo à Pesquisa é uma ferramenta genérica, que pode ser instanciada e customizada para atender às Fundações de Amparo à Pesquisa (FAPs) respeitando suas características regionais. Utiliza modelo de desenvolvimento incremental e está constantemente sendo evoluída com a finalidade de atender às novas demandas das FAPs. Atualmente são seis estados brasileiros que fazem uso do SIGFAP, sendo que em cada FAP são utilizadas siglas customizadas: Mato Grosso do Sul (SIGFundect), Amazonas (SIGFapeam), Pará (SIGFapespa), Piauí (SIGFapepi), Sergipe (SIGFapitec) e Mato Grosso (SIGFapemat).

O SIGFAP está organizado em duas áreas: área restrita do pesquisador e área administrativa (gestão da FAP), e contempla três grupos de usuários: pesquisadores, consultores ad-hoc e administradores. As funcionalidades, disponibilizadas pela ferramenta via Inter-

net para cada grupo de usuário, estão listadas a seguir.

- **Pesquisadores**: são os coordenadores e membros de projetos e candidatos às bolsas. Podem submeter propostas de projetos, submeter relatórios parciais e finais, acompanhar o julgamento das propostas, visualizar o histórico de suas atividades, preencher os formulários online de prestação de contas, verificar datas de prestação de contas e gerar relatórios.
- **Consultores ad-hoc**: são pesquisadores especializados, responsáveis pelo julgamento do mérito de propostas. Recebem convites de propostas para serem avaliadas, podendo aceitar ou recusar. Podem emitir pareceres e visualizar o histórico de revisões efetuadas.
- **Administradores**: são gestores e funcionários das FAPs. Podem avaliar a proposta, em função do mérito e do enquadramento em função do edital, podem acompanhar e visualizar todas as submissões, alterar o estado de cada proposta, gerar relatórios administrativos, enviar projetos e relatórios para serem avaliados pelos consultores ad-hoc e efetuar gestão financeira e administrativa.

Quanto ao processo de desenvolvimento do SIFAP, é utilizado o modelo combinado incremental e evolutivo. As tecnologias utilizadas são de uso livre e de código aberto: linguagem PHP, banco de dados PostgreSQL, SQLite, Web Services, sistema operacional Linux Debian e servidor Web Apache.

A seguir é apresentado o processo de integração do SIGFAP ao portal Web-PIDE, utilizando o PIBSAWL e o AIA no estudo de caso.

Ao iniciar a execução do PIBSAWL, na atividade 1, foram recuperados diversos artefatos, manual de usuário [63], código fonte em PHP, banco de dados em PostgreSQL, entre outros. Deve-se levar em consideração que o acesso a estes documentos só foi possível pois o candidato faz parte da equipe de desenvolvimento do SIGFAP. Como foi possível instalar e executar o SIGFAP em um ambiente de desenvolvimento, prosseguiu-se para a atividade 2.

Na atividade 2 do PIBSAWL, foi realizada a análise da arquitetura da aplicação e constatou-se que seria possível implementar as funcionalidades necessárias para fornecer um modo de autenticação e acesso remoto à aplicação por meio de web services. Com isso, foi projetada a arquitetura orientada a serviço ilustrada na Figura 4.4, onde tanto o SIGFAP quanto o Portal Web-PIDE se comunicam através do protolo LDAP com a base LDAP/Web-PIDE e entre si através de mensagens XML utilizando o protocolo SOAP e requisições GET utilizando o protocolo HTTP. Em seguida, foram escolhidos os requisitos para integração da aplicação com Ids: 1, 2, 3 e 4 presentes no gabarito do processo PIBSAWL, representado pela Tabela 3.1. Estes requisitos foram escolhidos com base no tipo de aplicação que é uma aplicação web de acesso restrito com arquitetura adaptável a SOA, entretanto outros requisitos poderiam ser definidos diferentemente destes especificados no gabarito do processo PIBSAWL, representado pela Tabela 3.1. Por fim, prosseguiu-se para a atividade 3 do PIBSAWL.

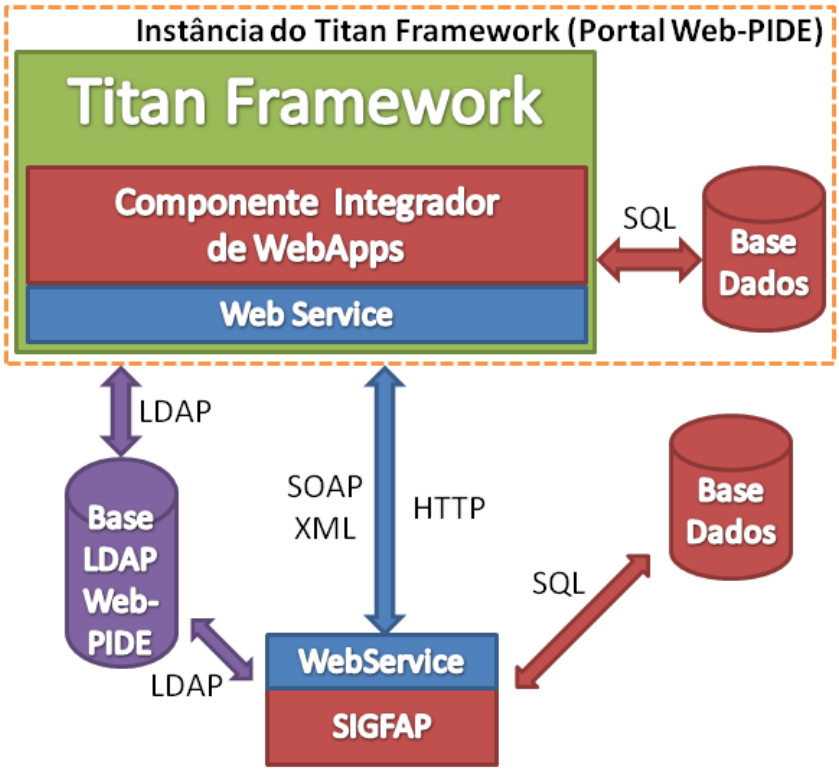


Figura 4.4: Arquitetura projetada para a integração do SIGFAP à plataforma Web-PIDE.

Na atividade 3, foi implementado o web service, responsável pela autenticação de usuários provenientes do AIA, neste caso o Portal Web-PIDE, conforme os requisitos definidos na atividade 2. Em seguida, foi criado o arquivo `admin/ws/logon.php` no diretório raiz da aplicação utilizando linguagem PHP que implementa tal web service, composto pela instanciação de um objeto da classe nativa do PHP, `SoapServer`, e a geração de uma classe `Logon` composto pelo método `toLogon` que representa o serviço de autenticação, propriamente, conforme ilustrado na Figura 4.5.

Na atividade 4, foi criado o arquivo `admin/toLogon.php` no diretório raiz da aplicação utilizando a linguagem PHP, ilustrado na Figura 4.6. Este arquivo implementa o processo de acesso de usuários autenticados do portal Web-PIDE via web service na aplicação utilizando para isto requisições GET com o *login* do usuário e a hash randômica e temporária gerada pelo web service da aplicação. Além disso, foi necessário a inclusão do arquivo `admin/toLogon.php` no arquivo principal da aplicação `admin/index.php` para que fossem capturadas as requisições GET enviadas pelo portal Web-PIDE para o SIGFAP. Nesta atividade foram escolhidos os casos de teste de unidade com Ids: 1, 2, 3, 4, 5, 6, 7, 8, 13, 14, 15 e 16, presentes no gabarito, definido pela Tabela 3.2 do PIBSAWL, para serem executados na atividade 5. Os casos de teste foram escolhidos com base no tipo de aplicação que é uma aplicação web de acesso restrito e arquitetura adaptável a SOA, entretanto outros casos de teste poderiam ser gerados para validar o processo de acesso dos usuários. Em seguida, prosseguiu-se para a atividade 5 do PIBSAWL.

Na atividade 5, foram testados o WS da aplicação para verificar se o mesmo estava

```
<?
require ("../../config.inc.php"); require ("../../conecta.php");
require ("../../classes/classe.ldap.php");
$server = new SoapServer (NULL, array ('uri' => $FUNDECT ['url'] .'admin/ws'));
class Logon {
    function toLogon($login, $hashSecurity) {
        global $FUNDECT; $hash = '';
        try {
            //Instanciando classe LDAP responsável pela manipulação da base LDAP
            $ldap = new Ldap( array ( 'id' => $FUNDECT['ldapId'], 'host' => $FUNDECT['ldapHost'],
                'user' => $FUNDECT['ldapUserAdmin'], 'password' => $FUNDECT['ldapKeyAdmin'],
                'gid' => $FUNDECT['ldapGID'], 'dn' => $FUNDECT['ldapDN'],
                'ou' => $FUNDECT['ldapOU'], 'update' => $FUNDECT['ldapUpdate']));
            //Recuperando informações do usuário na base LDAP
            $ldap->connect();
            $userLdap = $ldap->load($login, array('description', 'uid', 'mail', 'cn', 'pager'));
            //Tratando casos de acesso indevido
            if (!isset($userLdap['description'])
                    || $hashSecurity != md5($login.$userLdap['description'])) {
                $ldap->close();
                return array('STATUS' => 0, 'MESSAGE' => 'Hash Inválida!', 'HASH' => $hash);
            }
            //Tratando caso já exista na base da aplicação outro usuário cadastrado com o mesmo CPF
            $sqlBusca = "SELECT * FROM usuario WHERE usuario_cpf LIKE '".$userLdap['pager']."'
                        AND usuario_login NOT LIKE '".$userLdap['uid']."'";
            $queryBusca = pg_query ($sqlBusca);
            if ($userLdap['pager'] != '' && (int)pg_num_rows($queryBusca)){
                $ldap->close();
                return array('STATUS' => 0, 'HASH' => $hash,
                'MESSAGE' => 'Não foi possível criar usuário na aplicação remota, já existe outro
                    usuário associado ao CPF: '.$userLdap['pager'].'!');
            }
            if ($userLdap['pager'] != '' && (int)pg_num_rows($queryBusca)){
                $ldap->close();
                return array('STATUS' => 0, 'HASH' => $hash,
                'MESSAGE' => 'Não foi possível criar usuário na aplicação remota, já existe outro
                    usuário associado ao CPF: '.$userLdap['pager'].'!');
            }
            //Tratando caso o usuário ainda não tenha cadastro na aplicação
            $sqlBusca = "SELECT * FROM usuario WHERE usuario_login LIKE '".$userLdap['uid']."'";
            $queryBusca = pg_query ($sqlBusca);
            if (!(int)pg_num_rows($queryBusca)) {
                $sqlIns = "INSERT INTO usuario (usuario_nome,usuario_login,usuario_email,usuario_cpf)
                        VALUES ('".$userLdap['cn']."','".$userLdap['uid']."','".$userLdap['mail'].
                            "','".$userLdap['pager']."')";
                if (!pg_query ($sqlIns)) {
                    $ldap->close();
                    return array('STATUS' => 0, 'HASH' => $hash,
                        'MESSAGE' => 'Não foi possível criar usuário na aplicação remota!');
                }
            }
            //Gerando e retornando a nova hash para realizar o posterior acesso à aplicação
            $ldap->close();
            $new_hash = hash('sha512',sha1($login).$FUNDECT['hashSecurity'].floor(time()/10));
            return array('STATUS' => 1, 'MESSAGE' => 'Logon com sucesso!', 'HASH' => $new_hash);
        }
        catch (Exception $e) {
            return array('STATUS' => 0, 'MESSAGE' => $e->getMessage (), 'HASH' => $hash);
        }
    }
}
$server->setClass ('Logon'); $server->setPersistence(SOAP_PERSISTENCE_SESSION); $server->handle ();
?>
```

Figura 4.5: Arquivo `admin/ws/logon.php` do SIGFAP responsável pelo processo de autenticação na aplicação.

```
<?
//Verifica se a hash e o login de autenticação baseados em Web Service
//estão sendo passados por GET. Caso eles estejam corretos o usuário
//é autenticado e recebe acesso à aplicação. Caso contrário, o acesso
//do usuário é bloqueado.
if (isset($_GET['hashWS']) && isset($_GET['login']))
{
    $hash = hash('sha512',sha1($_GET['login']).$FUNDECT['hashSecurity'].floor(time()/10));

    if ($_GET['hashWS'] == $hash):

        $usuario = new Controle;

        $usuario->AutenticarWS($_GET['login']);

        $_SESSION["usuario"] = $usuario;

        header('Location: admin.php');

    else:

        if (isset($_SESSION["usuario"]))
            unset($_SESSION["usuario"]);

        die('Acesso Inválido');

    endif;
}
?>
```

Figura 4.6: Arquivo `admin/toLogon.php` do SIGFAP responsável pelo processo de acesso à aplicação.

realizando a autenticação correta de apenas usuários presentes na base LDAP/WEB-PIDE e se a aplicação estava permitindo o acesso apenas de usuários autenticados pelo WS da aplicação. Durante a execução desta atividade a aplicação custou algumas horas para ficar totalmente funcional, devido a erros na execução dos casos de teste. Com isso, foram precisos alguns ciclos de programação, à nível de código fonte, nos mecanismos de autenticação (desenvolvido na atividade 3) e de acesso (desenvolvido na atividade 4), ou seja, foram executadas algumas iterações das atividades 3, 4 e 5 até que a aplicação se tornasse funcional, sendo que os casos de teste, definidos na atividade 4, com Ids iguais a 1, 5 e 13 foram os mais problemáticos. Por fim, foram satisfaroriamente executados todos os casos de teste de unidade definidos na atividade 4 e o SIGFAP se tornou funcional, conforme ilustrado na Tabela 4.3, então, prosseguiu-se para a atividade 6 do PIBSAWL.

Tabela 4.3: Relatório de execução dos casos de teste de unidade do SIGFAP.

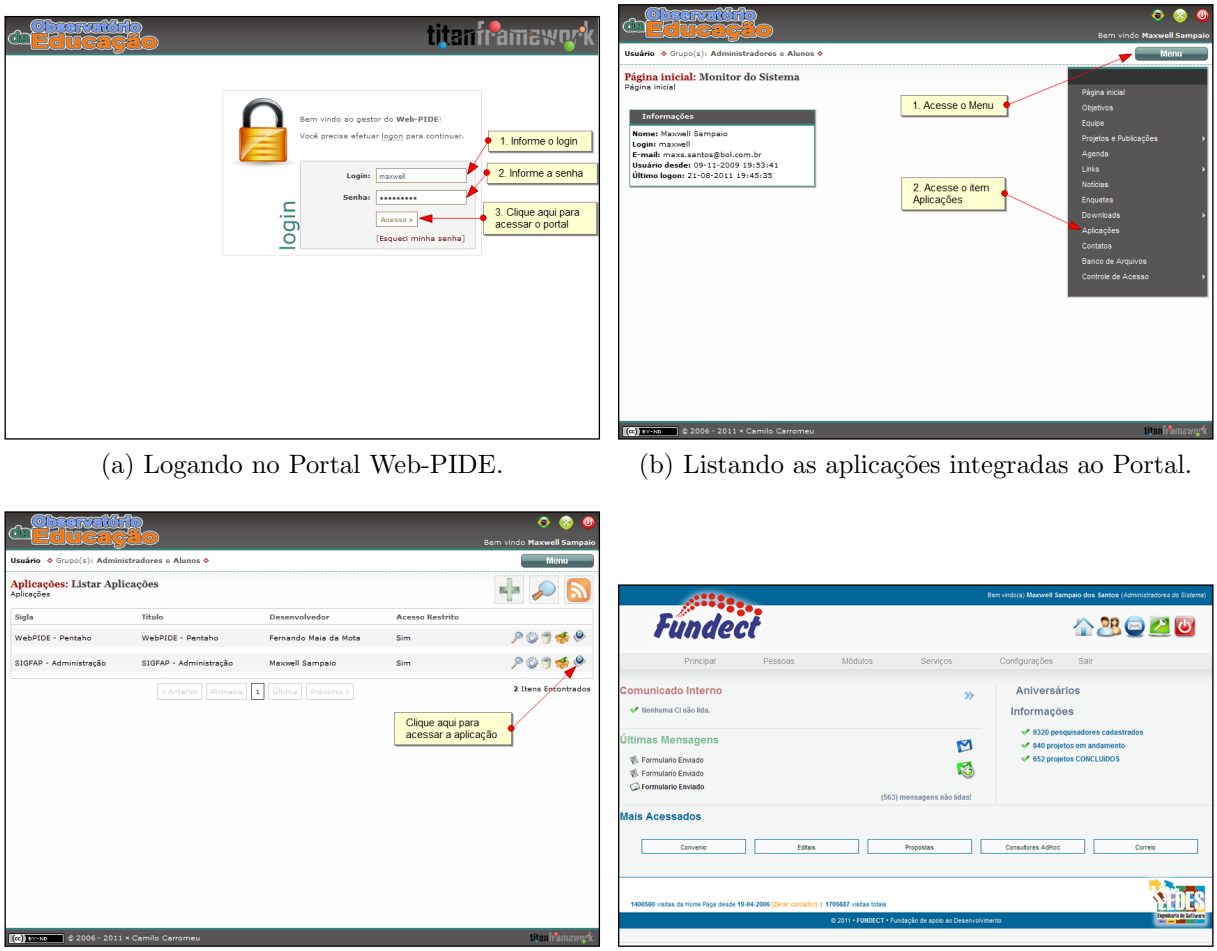
Id. Caso de Teste de Unidade	**Resultado da Execução**
CT_1, ..., CT_8, CT_{13}, ..., CT_{16}	Correto

Na atividade 6, o SIGFAP foi registrado no portal Web-PIDE. As informações registradas no portal podem ser conferidas na Figura 4.8. Foram anexados também ao registro do SIGFAP os manuais de uso da aplicação, recuperados na atividade 1. Em seguida, foi elaborado o "Documento de Casos de Teste de Integração", escolhendo os casos de teste de integração de Ids: 3, 4, 5, 6 e 7, presentes no gabarito, definido na Tabela 3.2,

para serem executados na atividade 7. Com isso, prosseguiu-se para a atividade 8 do PIBSAWL.

Tabela 4.4: Relatório de execução dos casos de teste de integração do SIGFAP.

Id. Caso de Teste de Integração	**Resultado da Execução**
CT_3, CT_4, CT_5, CT_6, CT_7	Correto



(a) Logando no Portal Web-PIDE.

(b) Listando as aplicações integradas ao Portal.

(c) Acessando o SIGFAP pelo Portal.

(d) Executando o SIGFAP.

Figura 4.7: Telas da execução do teste de integração do SIGFAP ao Portal Web-PIDE.

Na atividade 7, foram realizados os testes projetados na atividade 6, verificando se o usuário logado no Portal Web-PIDE obtinha acesso autenticado no SIGFAP, além de utilizar a aplicação sem demais problemas. Isto ocorreu com sucesso, conforme observado nas telas do Portal Web-PIDE e SIGFAP ilustradas na Figura 4.7 e no artefato produzido nesta atividade, definido na Tabela 4.4. Com isso prosseguiu-se para a atividade 8.

Na atividade 8, como o SIGFAP estava funcional então ele foi implantado em um ambiente de produção e prosseguiu-se para a atividade 9.

Na atividade 9, o registro da aplicação SIGFAP foi atualizado no portal Web-PIDE redirecionando para o ambiente de produção do SIGFAP, ou seja, foram atualizados os

Quanto ao Desenvolvedor

Nome: Maxwell Sampaio

Email: maxs.santos@gmail.com

Quanto à Aplicação

Tipo de Plataforma: Web

Logo: 

logo_sigfundect.png
16907 Bytes
image/png

Título: SIGFAP - Administração (Local)

Sigla: SIGFAP - Administração (Local)

Resumo: O SIGFAP ou Sistema de Informação e Gestão de Fundações de Amparo à Pesquisa é uma ferramenta genérica, que pode ser facilmente instanciada e customizada para atender às fundações de amparo à pesquisa (FAPs) respeitando suas características regionais. Utiliza modelo de desenvolvimento incremental e está constantemente sendo evoluída com a finalidade de atender às novas demandas das FAPs.

Descrição: O SIGFAP ou Sistema de Informação e Gestão de Fundações de Amparo à Pesquisa é uma ferramenta genérica, que pode ser facilmente instanciada e customizada para atender às fundações de amparo à pesquisa (FAPs) respeitando suas características regionais. Utiliza modelo de desenvolvimento incremental e está constantemente sendo evoluída com a finalidade de atender às novas demandas das FAPs. Atualmente são seis estados brasileiros que fazem uso do SIGFAP, sendo que em cada FAP são utilizadas siglas customizadas: Mato Grosso do Sul (SIGFundect), Amazonas (SIGFapeam), Pará (SIGFapespa), Piauí (SIGFapepi), Sergipe (SIGFapitec) e Mato Grosso (SIGFapemat).

O Sigfap está organizado em duas áreas: área restrita do pesquisador e área administrativa (gestão FAP), e contempla três grupos de usuários: pesquisadores, consultores ad-hoc e administradores.

Quanto ao processo de desenvolvimento do SIFAP, é utilizado o modelo combinado incremental e evolutivo. As tecnologias utilizadas são de uso livre e de código aberto: linguagem PHP, banco de dados PostgreSQL, SQLite, Web Services, sistema operacional Linux Debian e servidor Web Apache.

Quanto ao Acesso

Protocolo de Comunicação: HTTP

Host: maxwellubuntu

Porta: 80

Path: /sigfap/app/trunk/admin

Possui Acesso Restrito: Sim

Path para autenticação: /admin.php

Path usuário autenticado: /admin.php

Método de Autenticação: WebService

Quanto ao Acesso (Via WebService)

Path para o WebService: http://maxwellubuntu:80/sigfap/app/trunk/admin/ws/logon.php

URI para o WebService: http://maxwellubuntu:80/sigfap/app/trunk/admin/ws

Nome da Função ou Método no WebService para Autenticação: toLogon

Quanto aos Parâmetros de Acesso

Lista de Parâmetros de Acesso:

Name	Valor	Codificação
hashWS	Hash enviada pelo WebService da Aplicação Remota	
login	Login do Usuário Atualmente Logado no Sistema	

Figura 4.8: Informações do SIGFAP cadastradas no Portal Web-PIDE.

dados de acesso à aplicação SIGFAP tais como: url e porta de acesso à aplicação. Em seguida, prosseguiu-se para a atividade 10.

Na atividade 10, foram realizados novos testes de integração similares aqueles executados na atividade 7 e a aplicação SIGFAP foi, enfim, integrada ao Portal Web-PIDE. Com isso, a execução do PIBSAWL foi encerrada e constatou-se que o SIGFAP pode ser acessado através do Portal Web-PIDE por qualquer usuário cadastrado no portal. Usuários cadastrados na base LDAP/Web-PIDE e que não estão cadastrados na aplicação foram tratados e receberam um cadastro automatizado e próprio no SIGFAP, pois a aplicação exigia que o usuário tivesse um registro em seu próprio banco de dados para permitir a utilização da mesma.

4.4 Considerações Finais

Neste capítulo foram apresentados dois estudos de caso para avaliar o processo e o ambiente de integração propostos neste trabalho. As aplicações escolhidas para isso foram o Pentaho/Web-PIDE e o SIGFAP.

Com a execução dos estudos de caso foi constatado que o PIBSAWL e o AIA representam uma abordagem genérica e podem ser aplicados em outros domínios como por exemplo, gerenciadores de projetos das FAPs, como é o caso do SIGFAP, e não somente, no domínio de avaliação educacional, que junto com o projeto Web-PIDE foram os motivadores deste trabalho.

O *site* da UFMS[3], que também é uma instância do Titan, poderia se tornar um AIA e possibilitar que funcionários, discentes e docentes da UFMS pudessem acessar de forma transparente diversas aplicações web legadas de acesso restrito integradas ao *site* da UFMS.

Demais aplicações do tipo *desktop* e *web*, tanto de acesso público quanto de acesso restrito, desenvolvidas pelos membros do projeto Web-PIDE poderão ser registradas e disponibilizadas no Portal Web-PIDE utilizando o processo PIBSAWL, apresentado neste capítulo.

[3]http://www.ufms.br/

Capítulo 5

Trabalhos Relacionados

5.1 Considerações Iniciais

Neste capítulo são apresentados alguns trabalhos relacionados com o tema abordado na presente pesquisa, sob duas perpectivas: abordagens para integração de aplicações legadas e ambientes computacionais de apoio à integração de aplicações legadas.

5.2 Abordagens para Integração de Aplicações Legadas

Zhang *et al* [82] propõem uma estratégia caixa-preta para exportar funcionalidade interativa e personalizada de sistemas legados para Web Services, utilizando uma metodologia de empacotamento adequada para sistemas legados baseados em *Graphical User Interface* (GUI).

As interfaces legadas são reusadas e integradas dentro de um framework para minimizar o acoplamento com um sistema legado específico e para prevenir a decomposição do sistema legado em componentes internos. Para fazer com que as aplicações se tornem acessíveis em uma rede distribuída, uma ferramenta de *Virtual Network Computing* (VNC) é empregada e empacotada como um Web Service para implementar remotamente a funcionalidade legada. Além disso, uma linguagem de *script*, Lua [32], também foi utilizada de forma inteligente para executar operações auxiliares na GUI em nome dos usuários.

Zhang *et al* afirmam que devido ao baixo acoplamento com os sistemas legados, a estratégia de migração, caixa-preta, pode ser aplicada de maneira flexível para vários sistemas legados baseados em GUI, sendo ela validada com alguns sistemas tradicionais,

tais como: Rational Rose[1], ERWin[2] e Visual C++[3]. Neste trabalho também é apresentado um processo, que utiliza *Business Process Execution Language* BPEL, para a integração de sistemas legados baseados em GUI, o que facilita a integração de novos sistemas legados baseado em GUI.

Considerando o processo PIBSAWL e o ambiente de integração, propostos no presente trabalho de mestrado, as técnicas e o processo utilizados para realizar a integração de sistemas legados baseados em GUI, criados por Zhang *et al*, poderiam ser adaptados, em um trabalho futuro, para enriquecer a presente proposta, já que foi considerado no escopo do presente trabalho apenas a integração de aplicações web legadas. O PIBSAWL também leva em consideração o uso da estratégia de empacotamento dos sistemas web legados, tornando assim, mais fácil a integração de novos sistemas ao AIA proposto no presente trabalho de mestrado.

Já Han e Tokuda [29] propõem um método para integração de aplicações web legadas baseado na extração de informação gerando web services nas aplicações legadas capazes de fornecer e recuperar diversos tipos de informações da aplicação, tais como: textos, imagens, links, campos de formulários. Com isso as aplicações podem trocar informações e gerar novas páginas web com conteúdos recuperados dinamicamente.

Han e Tokuda apresentam todo um processo para a criação destes web services, com a definição dos tipos de dados disponibilizados pela aplicação, definição de um web service para busca de dados na aplicação e definição de um web service para extração dos dados. Sendo apresentado ainda um estudo de caso com o site da CNN News.

Comparado a proposta do presente trabalho de mestrado, o método para integração de aplicações web legadas apresentado não leva em consideração o controle de acesso às informações extraídas logo toda informação que é compartilhada via web service é pública. Já no presente trabalho de pesquisa, as aplicações web legadas de acesso restrito que forem integradas ao ambiente integrador serão tratadas para fornecerem acesso autenticado de usuários remotos e as funcionalidades delas permanecerão inalteradas. Logo somente usuários antenticados no ambiente integrador poderão ter acesso à aplicação web legada.

5.3 Ambientes Computacionais de Apoio à Integração de Aplicações Legadas

Sarmento [61] propõe a integração de um Ambiente Virtual de Aprendizagem (AVA) com três aplicações educacionais móveis com o intuito de ampliar os mecanismos de notificação e comunicação com os usuários, presentes nos atuais ambientes web de aprendizagem.

Para isso, o autor propôs a melhoria do ambiente virtual de aprendizagem (AVA), SOLAR, possibilitando a criação de uma camada de serviços, baseada em *Web Services*, que possibilite a integração entre o sistema legado, SOLAR, com um conjunto de aplicações

[1]http://www.ibm.com/software/awdtools/developer/rose/
[2]http://erwin.com/
[3]http://www.microsoft.com/pt-br/visualc/

móveis.

A arquitetura proposta por Sarmento, ilustrada na Figura 5.1, apresenta três aplicações móveis interligadas com um AVA através de uma camada de integração baseada em Web Services.

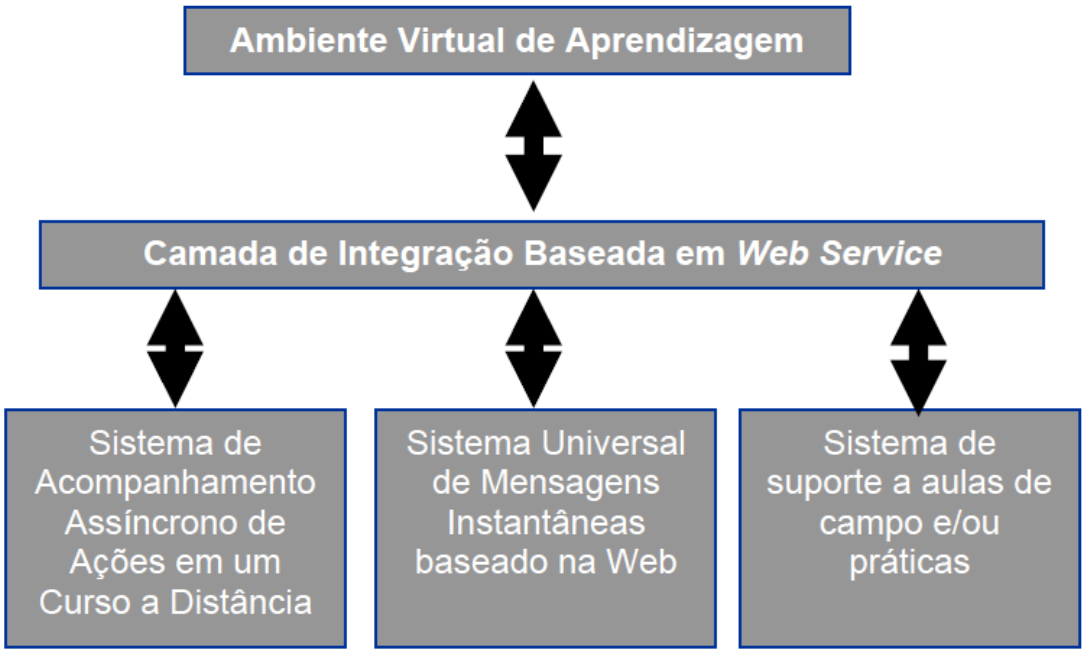


Figura 5.1: Arquitetura proposta por [61] para possibilitar a integração de aplicações móveis a AVA.

Para cada uma das aplicações móveis foi contruído um conjunto de Web Services reponsável pela comunicação entre a aplicação móvel e o AVA. Sendo que estes Web Services são responsáveis pela autenticação do usuário no AVA e codificação e decodificação dos dados trafegados entre uma aplicação móvel e o AVA.

Sarmento apresenta como foram construídas cada uma das aplicações móveis e também um estudo de caso sobre a execução delas integradas ao ambiente virtual de aprendizagem, SOLAR. Entretanto, comparado à proposta do presente trabalho de mestrado, Sarmento não apresentou um processo para a criação de novas aplicações móveis que possam ser integradas ao SOLAR deixando esta tarefa um tanto complicada para aqueles que forem criar novas aplicações para serem integradas ao SOLAR, pois, neste caso, seria necessário um estudo mais aprofundado sobre as técnicas e estratégias que foram utilizadas no desenvolvimento das aplicações móveis.

Além disso, o autor afirma que os Web Services desenvolvidos podem ser executados por qualquer AVA, entretanto ele não fornece as descrições WSDL dos Web Services e nem um processo de como seria a integração deles com outro AVA. Sendo este um dos focos do presente trabalho de mestrado, ou seja, fornecer caminhos e estratégias para realizar a integração baseada em serviços de sistemas legados mas, no caso, Web. Outro fator divergente entre as propostas (deste trabalho e do presente trabalho de mestrado) é que, no caso deste trabalho, toda vez que o usuário utilizar os aplicativos móveis precisa

se autenticar no AVA e isso não ocorre no ambiente proposto no presente trabalho, já que a autenticação do usuário é única e ele pode utilizar quaisquer sistemas legados que estiverem integrados ao ambiente sem necenssidade de uma nova autenticação, a menos que tenha expirado a sessão dele no ambiente.

Alkouz e El-Seoud [1], afirmam que plataformas de Ensino à Distância (EAD), que são consideradas como sistemas legados, enfrentam os seguintes desafios: 1) cada aplicação precisa implementar seu processo de autenticação no próprio diretório de armazenamento, 2) como o diretório de armazenamento cresce, a sobrecarga de desenvolvimento também aumenta, 3) como o número de plataformas de EAD cresce, torna-se difícil para os usuários relembrarem identificação e senha de acesso.

A solução apresentada por Alkouz e El-Seoud para enfrentar estes desafios é um sistema de autenticação baseado em web services, do inglês *Web Services Based Authentication System* (WSAS). O WSAS é flexível em termos de integração para plataformas heterogêneas e fácil para implantar pois é baseado em tecnologias Web Service (WS) como XML, WSDL, UDDI e SOAP. A tecnologia Web Service ajuda na implementação de um solução que resolve os três desafios mencionados. O mecanismo *single-sign-on* do WSAS permite autenticar os usuários em múltiplos diretórios de armazenamento de forma segura, unificada e centralizada. Três sistemas de autenticação mais comuns, SSL, Kerberos e Microsoft Passport foram analisados e considerados no WSAS.

A arquitetura do WSAS, ilustrada na Figura 5.2, consiste de quatro camadas, Usuário, Servidor de Aplicação, Servidor Principal e Diretórios de Armazenamento.

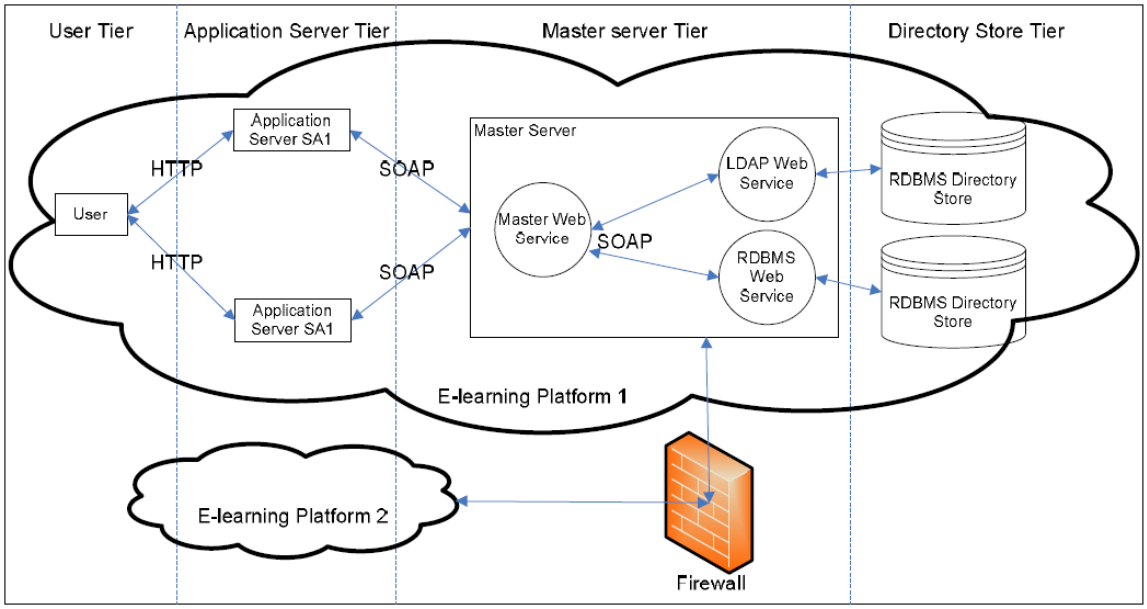


Figura 5.2: Arquitetura do WSAS com a integração de plataformas de EAD [1].

A camada de usuário é o local de interação do usuário com a plataforma de EAD que fornece acesso a um ou mais servidores de aplicação.

Usuários conectam na camada de servidores de aplicação usando mensagens HTTP. Eles precisam estar autenticados no servidor de aplicação. Então, baseado no resultado

da autenticação o servidor de aplicação pode conceder permissões aos usuários para acessarem diferentes recursos no servidor de acordo com os privilégios do usuário. Conceder privilégios é parte do processo de autorização no qual é específico de cada plataforma de EAD.

A camada do servidor principal é o núcleo do WSAS, ele atua como um servidor de confiança entre as plataformas de EAD. O servidor principal manipula o processo de autenticação de usuários, integrando múltiplos diretórios de armazenamento dentro do WSAS e fornecendo integridade, autenticidade e confidencialidade das mensagens transmitidas entre os usuários e os servidores de aplicação.

A camada de diretórios de armazenamento representam o banco de dados que é usado para manter os *profiles* dos usuários, ela pode ser um banco de dados relacional, diretórios de armazenameto LDAP ou um arquivo texto. Estes dados podem ser acessados baseado em informações de configuração enviadas ao servidor principal. Novos tipos de diretórios de armazenamento podem ser facilmente integrados ao WSAS, construindo um novo WS que possa interagir com este novo diretório de armazenamento, e implantá-lo no servidor principal com configurações apropriadas para se tornar comunicável com o WSP.

De acordo com Alkouz e El-Seoud, diferentes plataformas de EAD podem ser facilmente integradas ao WSAS, onde o servidor principal é construído baseado na tecnologia Web Service, o que representa um *firewall* amigável pois métodos podem ser chamados pelo protocolo SOAP, que não precisam abrir novas portas no *firewall* do próprio servidor. Plataformas de EAD podem se beneficiar do WSAS usando o servidor principal para autenticar seus próprios usuários ou disponibilizar serviço para seus servidores de aplicação.

Comparado à presente proposta de trabalho de mestrado, Alkouz e El-Seoud fornecem um ambiente computacional para integrar plataformas de EaD, entretanto não fornecem um processo bem explicativo de como deve ser realizada a integração de novas plataformas de EaD por meio deste ambiente, deixando esta tarefa para os desenvolvedores que forem realizar tais integrações terem de analisar como foram integradas outras plataformas de EaD e replicar as técnicas utilizadas em seu processo de integração, tornando complexa esta tarefa.

Arakaki [5] apresenta o projeto do FrameWork SAFE(*Software Engineering Available for Everyone*) que tem como objetivo a implementação de uma infra-estrutura para integrar ferramentas de Software Livre para auxiliar as atividades de Engenharia de Software, possibilitando um apoio automatizado ao processo de desenvolvimento de Projetos de Software Livre. O papel de cada ferramenta no framework é dar suporte a uma ou mais atividades do processo de Software Livre, dependendo dos objetivos de cada projeto. É relevante destacar que as ferramentas integradas neste ambiente são livres e independentes, como por exemplo Bugzilla[4], No Risk Planning[5] e Apache Subversion[6].

A arquitetura do SAFE, conforme ilustrada na Figura 5.3, apresenta o processo de

[4]http://www.bugzilla.org/
[5]http://sourceforge.net/projects/nrp/
[6]http://subversion.apache.org/

integração juntamente com as ferramentas presentes no SAFE, onde se destacam: o MINT (Módulo de Integração), uma interface de integração vertical, entre ferramentas, baseada em webservices entre o SAFE e as ferramentas que o compõem; o módulo de Gerências - que controla os serviços e recursos de integração no SAFE, englobando a integração horizontal e vertical.

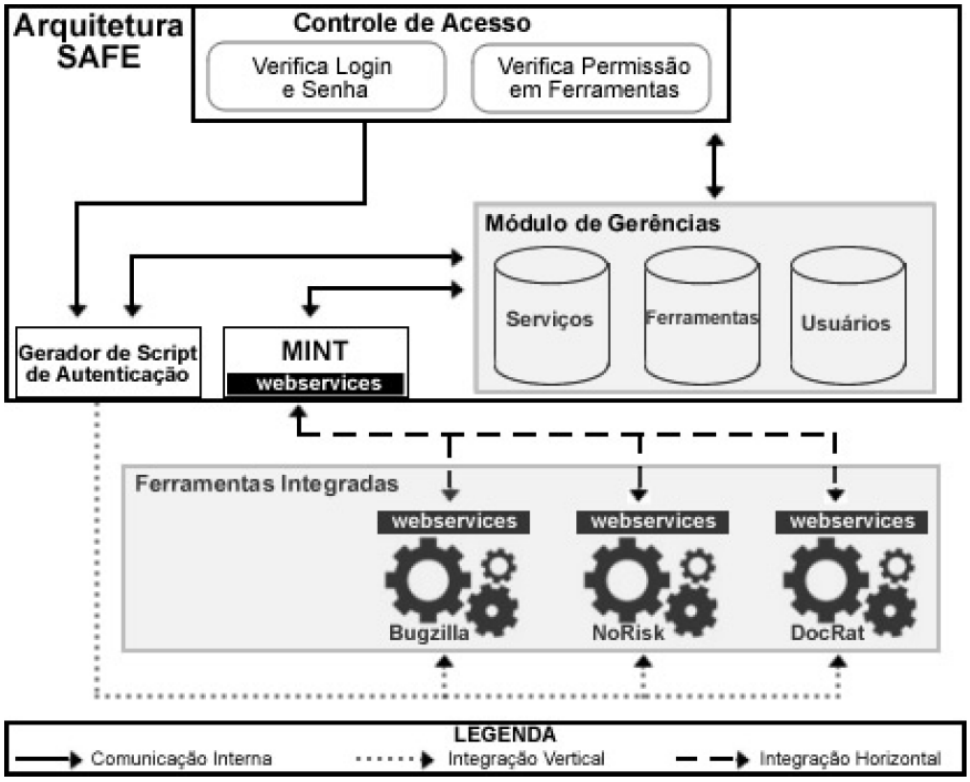


Figura 5.3: Arquitetura do SAFE [5].

A integração vertical tem o objetivo de facilitar o acesso às ferramentas que compõe o SAFE por meio da unificação na forma de controle de usuários e permissões em suas respectivas ferramentas, possibilitando ainda, uma única autenticação para acesso a ferramentas distintas.

Já a integração horizontal objetiva disponibilizar funcionalidades de ferramentas distintas em um ambiente integrado, a fim de possibilitar o aproveitamento de funcionalidades em comum para reduzir a redundância de tarefas relacionadas a determinada gerência. A integração de dados entre as ferramentas, ou integração horizontal, é realizada para permitir que os dados possam ser compartilhados e utilizados, de maneira transparente ao usuário, pelas ferramentas que compõem o ambiente SAFE.

Arakaki não fornece nenhum processo detalhado para a integração de novas ferramentas ao framework SAFE, apresentando apenas o ambiente computacional para integração de ferramentas de Software Livre que é o próprio Framework Safe. Entretanto, diferente da proposta do presente trabalho de mestrado, que utiliza uma base LDAP para centralizar os dados de cada usuário, o framework SAFE realiza o processo de autenticação utilizando *scripts* HTML para fornecer acesso único a todas as ferramentas integradas

a ele sendo, sendo que a cada novo usuário adicionado ao framework Safe é necessário replicar seus dados de acesso em cada uma das aplicações integradas ao framework, já que o framework não possui um local único com informações de todos os usuários, e isto, se torna um tanto complexo conforme são adicionados novos usuários e ferramentas ao Framework.

Segundo Cunha *et al* [18] é bastante comum os ambientes computacionais nas organizações, tanto públicas como privadas, serem heterogêneos. Ou seja, os diversos sistemas de informação são desenvolvidos em diferentes linguagens, executam em diferentes plataformas e possuem bases de dados em formatos distintos. Este é um cenário corriqueiro, inclusive entre empresas independentes, que possuem relacionamentos comerciais, do tipo parcerias temporárias ou fusões.

Ainda de acordo com Cunha *et al*, o setor público, por sua vez, está em posição desfavorável no que tange a integração de seus sistemas. As razões para isso ultrapassam a escassez de recursos e aportam na falta de pessoal técnico especializado, vontade política dos dirigentes e falta de visão estratégica dos processos de trabalho. Apesar desse cenário, o Governo Federal tem sentido a necessidade de executar significativas mudanças nas instituições públicas e sabe que, para isso, é necessário o aprimoramento dos sistemas de informação. Para tanto, definiu um conjunto mínimo de premissas, políticas e especificações técnicas que regulamentam a utilização da Tecnologia de Informação e Comunicação na interoperabilidade de Serviços de Governo Eletrônico e estabelecem as condições de interação com os demais poderes e esferas de governo, bem como com a sociedade em geral.

Dessa forma, Cunha *et al* apresentam uma proposta de arquitetura orientada a serviço para integração dos sistemas de informação do CEFET-AL (instituição pública do estado brasileiro Alagoas). A arquitetura proposta compreende a construção de um web service para cada sistema de informação que fornece a integração dos sistemas de informação com uma aplicação web que centraliza o acesso aos serviços de cada sistema de informação.

Comparado a presente pesquisa de mestrado, o trabalho realizado por Cunha *et al* não fornece detalhes sobre a implementação de cada um dos Web Services e muito menos um processo para a integração de novos sistemas de informação do CEFET-AL deixando esta tarefa de integração de novos sistemas do CEFET-AL complicada já que será necessário uma intensa comunicação com os responsáveis pelas integrações para esclarecimentos sobre as técnicas utilizadas na integração dos sistemas. Já a presente pesquisa de mestrado, prevê a definição de um processo genérico que pode ser aplicado para a integração de aplicações web legadas.

Diferentemente do AIA, proposto na presente pesquisa de mestrado, em que os sistemas legados são empacotados por meio de Web Services preservando todas suas funcionalidades, o ambiente proposto por Cunha *et al* teve o foco em serviços específicos de cada sistema de informação com isso, talvez não tenha sido possível criar uma solução genérica para integração dos sistemas de informação do CEFET-AL já que cada sistema possui suas particularidades.

5.4 Considerações Finais

A Tabela 5.1 apresenta uma síntese dos trabalhos relacionados e organiza-os sobre as perpectivas (Abordagem de Integração ou Ambiente Computacional de Apoio à Integração) e o tipo de aplicações legadas (*Desktop* ou *Web*) que podem ser integradas. Conforme observado na tabela não existem trabalhos voltados para a definição de uma abordagem para integração de aplicações web legadas associada com um ambiente computacional de apoio à integração deste tipo de aplicação sendo este o foco da presente pesquisa de mestrado.

Tabela 5.1: Síntese dos trabalhos relacionados ao tema da proposta de trabalho sob as perpectivas, (A) abordagens para integração de aplicações legadas ou (B) ambientes computacionais de apoio à integração de aplicações legadas, além do tipo de aplicação legada (*Desktop* ou *Web*) que pode ser integrada.

Autor do Trabalho	**Perpectiva**	**Tipo de aplicações integradas**
Alkouz e El-Seoud [1]	B	*Web*
Sarmento [61]	B	*Web*
Zhang *et al* [82]	A, B	*Desktop* de interface gráfica
Han e Tokuda [29]	A	*Web*
Arakaki [5]	B	*Web*
Cunha *et al* [18]	B	*Web*

De todos os trabalhos analisados o que teve maior similaridade com a presente proposta de mestrado foi o trabalho proposto por Alkouz e El-Seoud [1], em que foi desenvolvido um processo genérico para a integração de plataformas de EaD. A solução apresentada por Alkouz e El-Seoud utiliza tecnologias Web Service e LDAP para centralizar o processo de autenticação em cada plataforma de EaD, além de um *firewall* para promover mais segurança na troca de informações entre as plataformas de EaD.

O ambiente integrador proposto no presente trabalho também utiliza tecnologias Web Service e LDAP, entretanto foca principalmente no processo de autenticação e acesso de seus usuários a um sistema legado e, após a concretização deste acesso, a troca de informações ocorre apenas entre o sistema legado e o usuário sem a intervenção do ambiente integrador que concedeu o acesso do usuário ao sistema legado.

Capítulo 6

Conclusão

6.1 Contribuições

Como contribuições deste trabalho destacam-se, 1) definição de um processo para integração baseada em serviços de aplicações web legadas, 2) construção de um ambiente integrador de aplicações e 3) levantamento de aplicações desenvolvidas e trabalhos finalizados no projeto Web-PIDE do Observatório da Educação CAPES/INEP.

Conforme observado nos estudos de caso do Capítulo 5, o processo e o ambiente de integração propostos neste trabalho podem ser aplicados em outros domínios (gerenciadores de fundações de amparo à pesquisa, por exemplo) o que caracteriza uma abordagem genérica, pois o projeto de construção do ambiente levou em consideração a integração de aplicações web usando tecnologia Web Service. Tal solução permite a comunicação entre aplicações web baseada na troca de mensagens XML utilizando a combinação dos protocolos SOAP e HTTP, sendo que esta tecnologia pode ser implementada por meio de diversas linguagens de programação web tais como JAVA, PHP e Ruby.

O andamento e os resultados esperados do projeto Web-PIDE apresentados no Apêndice A deste trabalho também são importantes contribuições. O processo e o ambiente de integração adotados no portal Web-PIDE enriqueceram a proposta do projeto Web-PIDE, disponibilizando agora novos serviços aos usuários da plataforma, permitindo também que outras aplicações web no domínio de avaliação educacional possam ser integradas a ela, aumentando ainda mais os serviços oferecidos pela plataforma, como por exemplo, ferramentas desenvolvidas pelo INEP e pelos membros dos projetos do Observatório da Educação.

6.2 Limitações

As principais limitações da proposta são listadas a seguir:

- O ambiente de integração proposto neste trabalho levou em consideração apenas o

uso de tecnologia Web Service, mais especificamente o protocolo SOAP, entretanto, poderiam ser utilizados outras tecnologias Web Service como, REST e CORBA, por exemplo.

- O processo PIBSAWL e o AIA definidos neste trabalho não possibilitaram a integração de aplicações *desktop* legadas. O processo e o ambiente permitiram apenas o registro de informações de tais aplicações no AIA possibilitando o uso delas como estudos de caso de trabalhos futuros para realizar a integração destas aplicações.

- Infelizmente não foi possível a utilização de algumas aplicações web desenvolvidas pela equipe do projeto Web-PIDE como estudos de caso devido a complexidade de algumas aplicações e outras que não ficaram totalmente concluídas a tempo. Entretanto, isto não impede que o próprio desenvolvedor da aplicação possa integrá-la à plataforma Web-PIDE, posteriormente, utilizando, para itsto, o processo e ambiente propostos neste trabalho, já que ele possui o domínio necessário para realizar as alterações necessárias na aplicação.

- Conforme observado no estudo de caso do Pentaho/Web-PIDE a forma de comunicação escolhida entre o AIA e o Pentaho/Web-PIDE foi por GET, sendo transmitidos o *login* e a senha do usuário em texto puro. Como ambas aplicações funcionam sem criptografia dos dados transmitidos, essa forma de comunicação é insegura e foi apresentada apenas como estudo de caso, não sendo disponibilizada no ambiente de produção da plataforma Web-PIDE. Entretanto, com a implementação de uma camada de criptografia nos dados trasmitidos entre as aplicações, essa forma de comunicação poderia ser utilizada, conforme um dos trabalhos futuros mencionados a seguir.

6.3 Trabalhos Futuros

Como trabalhos futuros decorrentes deste trabalho, destacam-se:

- Realizar um estudo mais aprofundado sobre outras tecnologias de integração de aplicações como, Web Service utilizando REST, CORBA ou JavaBeans. Com isso, expandir o AIA identificando e implementando possíveis integrações com aplicações web legadas que utilizam tais tecnologias em suas arquiteturas.

- Pesquisar mais detalhadamente sobre processos de desenvolvimento de serviços a fim de readaptar o PIBSAWL para que seja possível desenvolver desde o princípio aplicações que utilizam uma arquitetura orientada a serviço e já estejam integradas ao AIA.

- Realizar levantamento bibliográfico quanto à integração de aplicações *desktop* legadas e adaptar o processo PIBSAWL e o ambiente propostos para contemplar tal integração. Uma possível solução para isto seria a utilização de ferramentas web de acesso remoto instaladas no servidor que hospeda o AIA realizando ajustes no servidor para que autentique seus usuários na mesma base LDAP do AIA.

- Realizar um estudo mais aprofundado quanto aos níveis de permissão dos usuários nas aplicações integradas pelo processo PIBSAWL com o intuito de permitir a definição de grupos de acesso personalizáveis e integrados à base LDAP do AIA. Grupos estes criados para serem utilizados pelas aplicações integradas ao AIA. Com isso, o AIA, por exemplo, atuaria como um sistema de controle geral de permissões nas aplicações e não somente como um controlador de acesso a elas.

- Efetuar levantamento bibliográfico quanto aos métodos de criptografia SSL (*Secure Socket Layer*) e TLS (*Transport Layer Security*) para serem utilizados no AIA. Dessa forma, tanto instâncias do Titan quanto as aplicações que forem integradas a ela teriam uma camada de criptografia oferecendo ainda mais segurança na troca de informações entre elas. O uso do protocolo HTTPS (*Hypertext Transfer Procotol Secure*) pode ser levado em consideração neste caso.

- Construir e disponibilizar um Web Service e um conjunto de mapas geográficos com indicadores quanto ao ensino no Brasil para serem utilizadas, por exemplo, por aplicações Web que não pertençam à plataforma Web-PIDE, assim como acontece com a ferramenta Google Maps[1] que permite exibir mapas geográficos personalizados em uma aplicação Web. Com isto, estas aplicações poderiam exibir mapas personalizáveis baseado em dados da plataforma Web-PIDE, como por exemplo: quantidade de alunos aprovados na 1ª série do Ensino Médio em cada estado do Brasil. Sendo que as aplicações precisariam requisitar acesso (baseado em login e senha, por exemplo) ao Web Service e só então poderiam exibir os mapas geográficos com tais informações em suas interfaces com o usuário.

- Adaptar a plaforma Web-PIDE a fim de disponibilizá-la em aparelhos celulares e televisores que possuam acesso à Internet, expandindo ainda mais o acesso à plataforma.

- Realizar um levantamento e análise das ferramentas desenvolvidas e utilizadas pelo MEC/INEP e articular junto ao INEP a integração delas à plataforma Web-PIDE utilizando o processo e ambiente propostos neste trabalho. Por exemplo, o sistema de inscrição, acompanhamento e desempenho do ENEM[2] poderia ser integrado à plataforma Web-PIDE, possibilitando assim que o aluno realizasse um único login para ter acesso às incrições, acompanhamentos e desempenhos dele em todos os anos que ele participou do ENEM.

[1] http://code.google.com/intl/pt-BR/apis/maps/documentation/javascript/v2/articles.html
[2] http://sistemasenem.inep.gov.br/

Apêndice A

Projeto Web-PIDE

Considerações Iniciais

Neste capítulo é apresentada uma visão geral do projeto Web-PIDE que foi utilizado como estudo de caso no ambiente de integração proposto neste trabalho. Em seguida, é apresentado como surgiu o projeto, quais trabalhos foram desenvolvidos pela equipe do projeto. Por fim, são apresentados os resultados parciais e esperados do projeto.

Visão Geral

Em 2007, a CAPES em parceria com o INEP iniciaram o Programa Observatório da Educação, uma iniciativa para fomentar o desenvolvimento de estudos e pesquisas em educação, com a finalidade de estimular a produção acadêmica e a formação de recursos pós-graduados, em nível de mestrado e doutorado.

Com isso em janeiro de 2007, através do Programa Observatório de Educação, foi aprovado o projeto intitulado "Web-PIDE — Uma Plataforma aberta para Integração e avaliação de Dados Educacionais na Web" [72] de parceria entre a UFMS e UFSCAR e com duração de quatro anos.

O projeto tem como objetivo produzir um sistema computacional para integrar e disponibilizar os dados educacionais do INEP, por meio de uma linguagem comum e padronizada de marcação intitulada Linguagem de Marcação de Dados Educacionais (LIDE), como acontece com a plataforma Lattes [39] do CNPq. Além disso, a plataforma também objetiva facilitar a consulta aos dados de avaliação educacional gerando hipóteses a serem investigadas através de ferramentas de visualização.

O desenvolvimento de um eficiente sistema nacional de informação educacional tem orientado a atuação do governo federal no que se refere à sua função supletiva, voltada para a superação das desigualdades regionais. Com os instrumentos criados, o MEC pode estruturar programas destinados, especificamente, a suprir deficiências do sistema educa-

cional existente. Porém, observa-se que além de ter o sistema de informação é necessário ter estratégias, padrões e mecanismos para facilitar a exportação dos dados educacionais para sua integração com outros sistemas de informação. Por exemplo, a UFMS possui um sistema de avaliação institucional específico e para usar os dados do ENEM, primeiramente, deve solicitar ao INEP liberação de acesso aos dados, analisar o padrão dos dados, decidir como importar e, por fim, verificar quais dados podem ser utilizados para análise. Este processo é demorado e pode consumir muito custo de desenvolvimento computacional, podendo inviabilizar o objetivo final que é reutilizar os dados gerados e integrá-los com dados locais da universidade para uma análise e uma avaliação institucional.

Assim, o presente quadro motiva estudos e pesquisas em tecnologia e educação para tornar fácil o reuso dos dados educacionais disponíveis pelo INEP. Uma vez integrados às bases institucionais das universidades, é possível traçar um quadro abrangente da situação educacional do país e fornecer subsídios indispensáveis para o aprofundamento de análises e pesquisas críticas que possam enriquecer o debate sobre os rumos da educação brasileira.

O projeto é caracterizado como **Núcleo em Rede** no Programa Observatório da Educação, pois é composto por docentes orientadores e estudantes de mestrado (quatro para cada período de dois anos, de 2007-2008 e 2009-2010) dos programas de pós-graduação *stricto sensu* da Faculdade de Computação da Universidade Federal de Mato Grosso do Sul e do Departamento de Computação da Universidade Federal de São Carlos. Além disso, participaram do projeto, como colaboradores, alunos de graduação da UFMS e UFSCar. Ao todo o projeto contou, até o presente momento, com a colaboração de mais de 20 pessoas — alunos de mestrado, alunos de graduação, docentes colaboradores, docentes orientadores e coordenadores do projeto.

A seguir, são descritas as principais bases de dados do INEP que sustentam os sistemas de avaliação do MEC.

Bases de Dados do INEP

Na Figura A.1 é apresentada a estrutura dos sistemas de avaliação do INEP e as 11 principais bases de dados geradas.

No ensino básico (que compreende o ensino regular e o ensino médio), as principais bases de dados produzidas pelo INEP são SAEB, Prova Brasil e ENEM, que são geradas a partir de seus respectivos sistemas de avaliação. Os dois primeiros são exames complementares que compõem o Sistema de Avaliação da Educação Básica (SAEB) e o último é um exame que avalia os estudantes egressantes do ensino médio, utilizado, principalmente, por estudantes que almejam ingressar no ensino superior.

Já no ensino superior, compondo o Sistema Nacional de Avaliação do Ensino Superior (SINAES), a principal base de dados produzida pelo INEP é o Exame Nacional de Desempenho de Estudantes (ENADE) que agrega a Avaliação dos Cursos de Graduação e a Avaliação das Instituições de Educação Superior.

A seguir, são descritos cada um destes sistemas de avaliação e suas respectivas bases de dados.

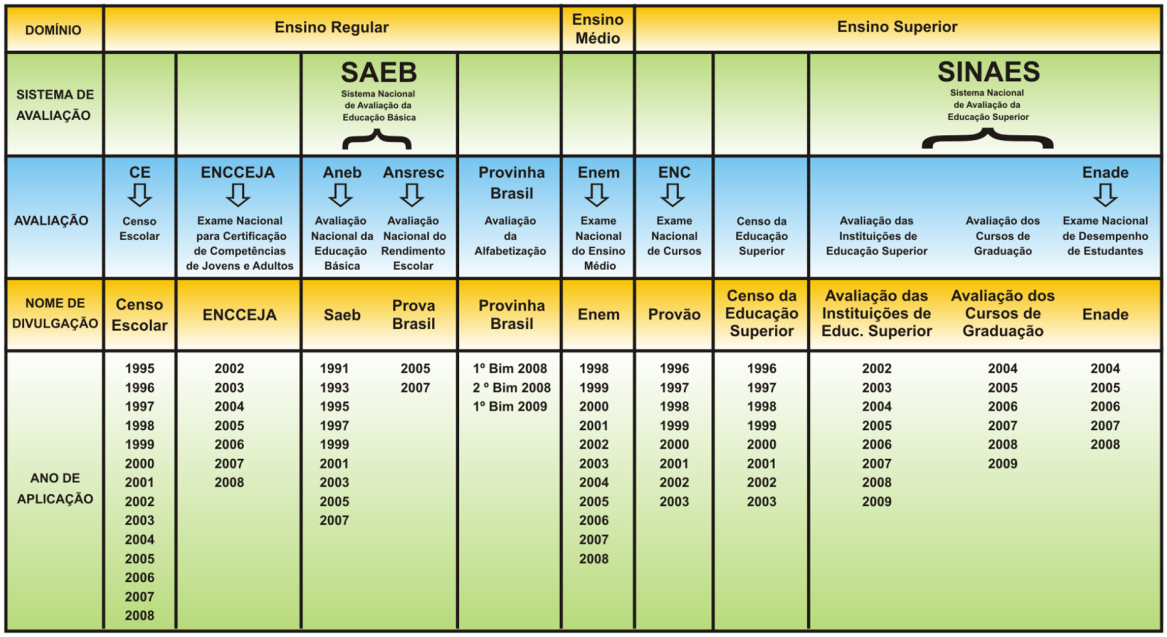


Figura A.1: Sistemas de Avaliação e Bases de Dados do INEP. [30]

Saeb

O Saeb [9] foi a primeira iniciativa brasileira, em escala nacional, para se conhecer o sistema educacional brasileiro em profundidade. Ele começou a ser desenvolvido no final dos anos 80 e foi aplicado pela primeira vez em 1990.

Em 1995, o Saeb passou por uma reestruturação metodológica que possibilita a comparação dos desempenhos ao longo dos anos. Desde a sua primeira avaliação, fornece dados sobre a qualidade dos sistemas educacionais do Brasil como um todo, das regiões geográficas e das unidades federadas.

O Saeb é realizado a cada dois anos e avalia, através de exame de proficiência em Matemática e em Língua Portuguesa, uma amostra representativa dos alunos regularmente matriculados nas $4^{\underline{a}}$ e $8^{\underline{a}}$ séries do ensino fundamental e $3^{\underline{o}}$ ano do ensino médio, de escolas públicas e privadas, localizadas em área urbana ou rural. Além de coletar dados sobre a qualidade da educação no País, o SAEB procura conhecer as condições internas e externas que interferem no processo de ensino e aprendizagem, por meio da aplicação de questionários de contexto respondidos por alunos, professores e diretores, e por meio da coleta de informações sobre as condições físicas da escola e dos recursos de que ela dispõe.

Prova Brasil

A Prova Brasil [9] foi criada e aplicada em 2005, a partir da necessidade de se tornar a avaliação mais detalhada, em complemento à avaliação já feita pelo SAEB, sendo novamente aplicada em 2007. Ela é censitária e por esta razão, expande o alcance dos resultados, porque oferece dados não apenas para o Brasil e unidades da Federação, mas também

para cada município e escola participante. Ela avalia todos os estudantes da rede pública urbana de ensino, de 4ª e 8ª séries do ensino fundamental, excluindo assim os alunos da rede pública rural de ensino e os alunos do 3º ano do Ensino Médio que são avaliados apenas pelo SAEB.

ENEM

Criado em 1998, o Exame Nacional do Ensino Médio (ENEM) [9] tem o objetivo de avaliar o desempenho do estudante ao fim da escolaridade básica. Podem participar do exame alunos que estão concluindo ou que já concluíram o ensino médio em anos anteriores.

O ENEM é utilizado como critério de seleção para os estudantes que pretendem concorrer a uma bolsa no Programa Universidade para Todos (ProUni). Além disso, centenas de instituições de ensino superior já usam o resultado do exame como critério de seleção para o ingresso no ensino superior, seja complementando ou substituindo o vestibular institucional.

ENADE

No período de 1996 a 2003, foi aplicado aos formandos de graduação o Exame Nacional de Cursos (ENC-Provão) [9], com o objetivo de avaliar os cursos de graduação da Educação Superior, no que tange aos resultados do processo de ensino-aprendizagem.

A partir de 2004, o INEP extinguiu o ENC-Provão e passou a aplicar o ENADE, que tem como objetivo avaliar o desempenho dos estudantes com relação aos conteúdos programáticos previstos nas diretrizes curriculares dos cursos de graduação, o desenvolvimento de competências e habilidades necessárias ao aprofundamento da formação geral e profissional, e o nível de atualização dos estudantes com relação à realidade brasileira e mundial.

Arquitetura da Plataforma Web-PIDE

A arquitetura da plataforma Web-PIDE é ilustrada na Figura A.2, sendo organizada em três camadas: dados, negócio e interface. Para cada base de dados do INEP, por exemplo, ENADE, ENEM e SAEB é desenvolvida uma LIDE específica. Por meio das LIDEs objetiva-se facilitar o armazenamento, recuperação e consulta dos dados relativos às várias bases de dados do INEP. O armazenamento e recuperação de dados é realizado utilizando a tecnologia *Data Warehousing[22]* que se preocupa em integrar e consolidar as informações de fontes heterogêneas e fontes externas, sumarizando, filtrando e limpando estes dados, preparando-os para análise e suporte à tomada de decisão utilizando ferramentas de visualização além de hipóteses gráficas e textuais para isso. Sendo possível utilizar ferramentas próprias da plataforma Web-PIDE ou outras ferramentas que deverão ser integradas à plataforma.

As principais características que a plataforma Web-PIDE proporcionará são:

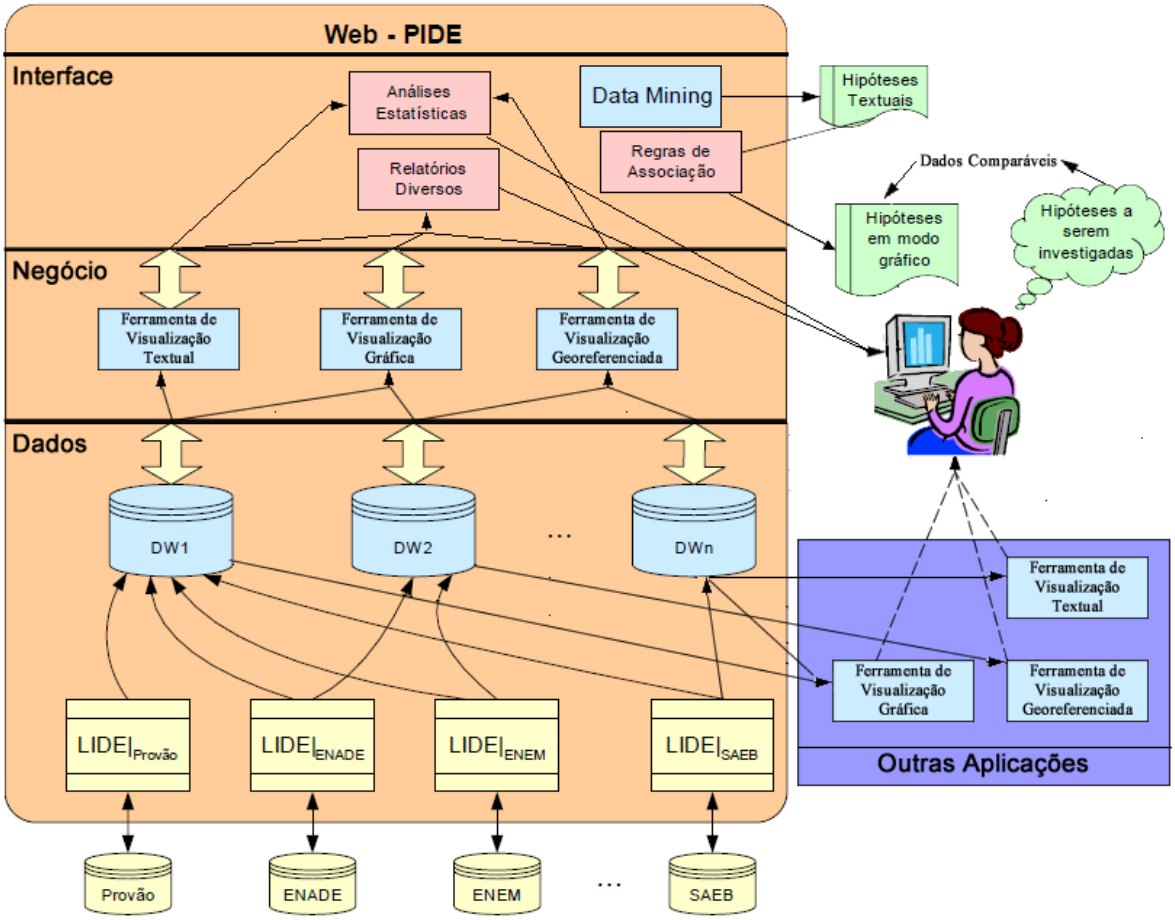


Figura A.2: Arquitetura da plataforma Web-PIDE. Adaptada do Projeto Web-PIDE [72]

- Extração de dados educacionais de fontes heterogêneas (existentes (do INEP, por exemplo) ou externas (das universidades públicas));
- Transformação e integração dos dados antes de seu armazenamento na plataforma;
- Visualização dos dados em diferentes níveis. Os dados do DW podem ou não ser extraídos para um nível mais específico, os Data Marts (DM), e a partir destes para um banco de dados individual; e
- Utilização de ferramentas voltadas para acesso com diferentes níveis de apresentação.

Pesquisas Desenvolvidas

Diversas pesquisas foram realizadas pelos membros do projeto sendo que algumas já renderam trabalhos de iniciação científica, graduaçao e mestrado além de artigos científicos

submetidos e aceitos em eventos nacionais e internacionais. A seguir, são apresentados os principais trabalhos e produtos resultantes das pesquisas realizadas e concluídas. Vale ressaltar que outras pesquisas estão em execução no projeto mas serão apresentadas aqui somente as pesquisas finalizadas.

Portal Web-PIDE

O Portal Web-PIDE[79], disponível no endereço http://webpide.ledes.net/, foi produzido inicialmente pelo aluno Jackson Dias Savitraz e continua sendo desenvolvido por outros membros da equipe do projeto, sob supervisão do Prof. Dr. Marcelo Augusto Santos Turine - UFMS. Ele é composto por uma área pública (informativa) e uma área restrita (administrativa), como pode ser visto na Figura A.3 e está sendo construído de modo a se tornar uma plataforma on-line para integrar e disponibilizar informações educacionais fornecidas pelo INEP, pelos membros do projeto e pela comunidade em geral interessada em avaliação educacional.

Figura A.3: Área Informativa (à esquerda) e Área Administrativa (à direita) do Portal Web-PIDE [42].

Área Informativa

A área informativa do portal foi implementada com tecnologia PHP e banco de dados PostgreSQL. Além disso, utilizou-se o controlador de versões SubVersion, e ferramentas colaborativas Wiki e Gforge para gerenciar o desenvolvimento do portal.

Esta área tem como objetivo divulgar informações de cunho aberto e de interesse público, onde os interessados podem acessar diversas funcionalidades para avaliação de dados educacionais. Ela é composta de seções como: projetos e publicações, notícias e arquivos. Na seção de projetos e publicações estão disponíveis os projetos de mestrado e

graduação relacionados ao projeto Web-PIDE. Enquanto que a seção de notícias agrupa e disponibiliza informações pertinentes às avaliações do INEP além de novidades sobre o andamento do projeto Web-PIDE.

A página inicial do portal possui também um componente visual no formato de um mapa, que permite a comparação do desempenho escolar de estudantes. Além de uma área restrita, onde usuários do portal têm acesso à área administrativa do portal, e links de acesso rápido para aplicações utilizadas no projeto, por exemplo, a Wiki do projeto Web-PIDE, como podem ser vistos na Figura A.3.

Área Administrativa

Esta área foi instanciada utilizando-se o Titan [13] [12] [49], que é um framework para instanciação de *Content Management Systems* (CMS), ou seja, gerenciadores de conteúdo para sistemas Web. Por meio deste framework é possível realizar a instanciação de gerenciadores de conteúdo de forma rápida e fácil, utilizando o reuso de componentes.

Na área administrativa é possível controlar basicamente todo o conteúdo da área informativa do portal. Nela é possível, por exemplo: gerenciar usuários, permissões e grupos; gerenciar notícias e agendas; disponibilizar artigos e dissertações de mestrado para *download*, entre outras funcionalidades.

Para que um usuário tenha acesso a área administrativa, é necessária a realização de um pré-cadastro acessando o *link* "não sou cadastrado" presente na página inicial do portal. O sistema possui quatro grupos de usuários pré-definidos: administradores, alunos, gestores, e orientadores; e após a realização e efetivação do cadastro, o usuário recebe privilégios dos grupos que ele estiver vinculado. Sendo que alunos podem disponibilizar conteúdos como notícias e agenda, orientadores podem disponibilizar artigos, dissertações de mestrado e trabalhos de graduação, gestores possuem acesso limitado e administradores possuem acesso total.

Ferramenta DEAR

O INEP disponibilizou todos os microdados de cada avaliação educacional para o projeto, sendo cada um deles composto por: um arquivo texto puro em formato ASCII (também chamado de microdado) que contém os dados coletados pela avaliação; um arquivo SAS (também chamado de dicionário de variáveis) que contém a descrição padronizada de cada uma das variáveis presentes no microdado; um arquivo PDF com a descrição detalhada de cada uma das variáveis presentes no microdado.

Dessa forma, com o intuito de permitir que os microdados sejam carregados em relações do PostgreSQL foi desenvolvido pelo aluno Thiago Luís Lopes Siqueira, sob orientação do Prof. Dr. Ricardo Rodrigues Ciferri - DC/UFSCar, a ferramenta intitulada "*Data Extractor ASCII to Relational* (DEAR)" [65]. Desenvolvida em Java, mas sem interface Web, a ferramenta possui duas etapas de execução. A primeira etapa é responsável por gerar o dicionário de variáveis (caso este ainda não esteja definido) e armazená-lo em uma relação do PostgreSQL, conforme ilustra a Figura A.4. Nesta etapa é importante utilizar, como apoio à definição do dicionário, os arquivos PDF e SAS que descrevem detalhamente

cada uma das variáveis presentes no microdado que será importado posteriormente. Foi necessária essa etapa pois o dicionário de variáveis disponibilizado pelo MEC continha diversas inconsistências com os dados brutos.

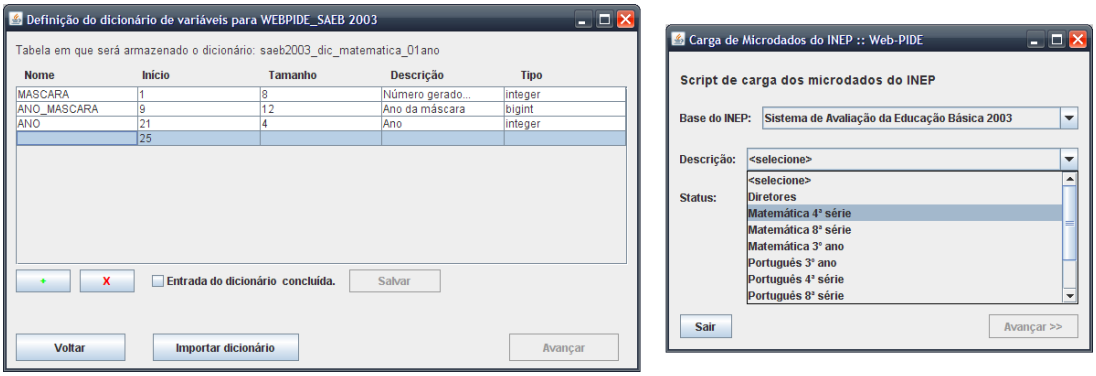


Figura A.4: Definição do dicionário de variáveis (à esquerda) e carga do microdado em relações do PostgreSQL (à direita) utilizando a ferramenta DEAR [42].

Conforme ilustra a Figura A.4, na etapa seguinte ocorre efetivamente a carga do microdado no PostgreSQL. Baseando-se em uma configuração pré-definida em relações do PostgreSQL e no dicionário de varíaveis definido anteriormente, a ferramenta solicita qual avaliação, ano e tipo que deseja importar e realiza a carga do microdado.

Para ilustrar todo o processo de carga de microdados do INEP em relações do PostgreSQL por meio da ferramenta DEAR tem-se a Figura A.5.

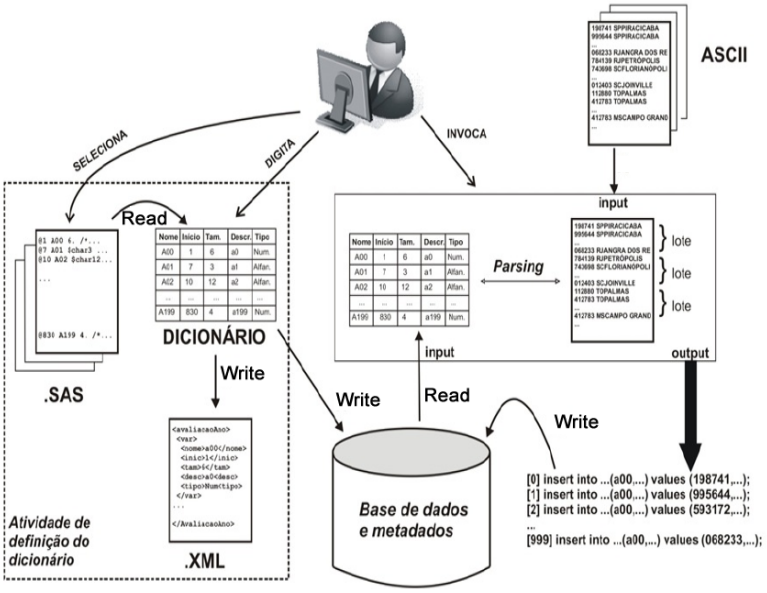


Figura A.5: Processo completo de carga dos microdados do INEP em relações do PostgreSQL na plataforma Web-PIDE [65].

O auxílio que esta ferramenta proporciona ao andamento do projeto é muito importante. Como o foco do projeto está na educação básica então já foram importados,

utilizando tal ferramenta, todos os microdados do CEB, SAEB e ENEM para relações do PostgreSQL e os demais microdados também serão importados utilizando-se tal ferramenta, brevemente.

O aluno, Thiago Luís Lopes Siqueira, responsável pela criação da ferramenta DEAR, submeteu e apresentou um artigo intitulado "Projeto, Construção e Manutenção de Data Warehouses para Auxiliar o Planejamento de Políticas Públicas de Educação" [] na XVI *Jornadas de Jóvenes Investigadores* ocorrido em Montevideo, Uruguai, 2008. Pela produção do artigo ele ainda recebendo o prêmio de "*Mejor Trabajo del área Evaluación Institucional, Planeamiento Estratégico y Gestión Universitária en las XVI Jornadas de Jóvenes Investigadores, Asociación de Universidades Grupo Montevideo* do ano de 2008.

Ele também realizou pesquisas sobre *Data Warehouse* Geográfico tendo concluído sua dissertação de mestrado intitulada "SB-INDEX: Um Índice Espacial baseado em Bitmap para Data Warehouse Geográfico" [64] em 2009. Além disso, ele submeteu e apresentou, no mesmo ano, o artigo intitulado "*A spatial bitmap-based index for geographical data warehouses*"[41] em um evento internacional. A dissertação de mestrado produzida pelo aluno foi classificada dentre as 11 melhores do ano de 2009 pelo XXII Concurso de Teses e Dissertações da Sociedade Brasileira de Computação.

A pesquisa realizada pelo aluno em sua dissertação de mestrado poderá contribuir, efetivamente, para a plataforma Web-PIDE já que o índice proposto pelo aluno poderá ser utilizado para agilizar as consultas nos DWs produzidos pela equipe do projeto.

Ferramenta VIDEWeb

Após o processo de carga dos microdados do SAEB para relações do PostgreSQL, surgiu a necessidade de gerar hipóteses a serem investigadas sobre o SAEB, a fim de entender melhor como está organizada tal base. Para isso foi desenvolvida uma aplicação Web intitulada "VIDEWeb — Um ambiente para Visualização de Dados Educacionais na plataforma Web-PIDE" [42], pelos alunos Mario Cesar da Cunha Machado, Maxwell Sampaio dos Santos e Rodrigo Pinto Gondim, todos da UFMS, sob orientação do Prof. Dr. Marcelo Augusto Santos Turine e auxiliado pelo aluno Thiago Luís Lopes Siqueira - DC-UFSCar.

A metodologia adotada para a geração da ferramenta foi, primeiramente, elaborar possíveis hipóteses a serem investigadas sobre os dados do SAEB, para isso foi gerada uma lista de perguntas pertinentes aos dados e direcionadas aos interesses da equipe do projeto para uma avaliação dos resultados. Após esse processo, foi necessária a geração de uma micro base de dados do SAEB com apenas informações necessárias para responder tais perguntas, dado que o volume de dados era extremamente grande e isso poderia tornar a aplicação ineficiente. Por fim, com uso de tecnologia Java e JPivot[1][36], foi construída a ferramenta que permite a geração de gráficos 2D a partir de consultas realizadas na micro base do SAEB, sendo ela disponibilizada no endereço http://webpide.ledes.net:8080/videweb/.

[1]JPivot é uma biblioteca em linguagem JSP customizada que renderiza tabela OLAP e permite que usuários executem navegações típicas em OLAP como *slice and dice*, *drill down* e *roll up*.

A contribuição deste trabalho para o projeto foi significante pois, boa parte da equipe do projeto pôde vivenciar um processo de carga de microdados em um SGBD além de conhecer um pouco mais sobre a base do SAEB, pois muitas inconsistências entre os microdados e os arquivos PDF e SAS foram encontradas e registradas em relatórios. Além disso, este trabalho serviu como base para a geração do DW do SAEB pelos alunos da UFMS de Coxim em parceria com os alunos da UFSCar.

Ferramentas: SEV-Tool e ONTOP-Tool

Para auxiliar na análise das bases do INEP e definição da LIDE (especialmente para o SAEB), além de permitir a construção de uma ontologia no contexto das avaliações do INEP, foram desenvolvidos, pela aluna Elis Cristina Montoro Hernandes, sob orientação da Profª Dra. Sandra Camargo Pinto Ferraz Fabbri, dois processos, P-LIDE e ONTOP, e duas ferramentas intituladas, SEV-Tool e ONTOP-Tool [30], respectivamente.

A motivação para a criação das ferramentas deu-se pelo fato que quando uma determinada questão do formulário do SAEB fosse adicionada ao DW era necessário analisar todos os questionários aplicados durante realização do SAEB para identificar se ocorreu tal questão e como ela foi elaborada, pois muitas vezes a mesma questão não foi escrita com a mesma sintaxe e alguns termos foram utilizados com significados diferentes durante os anos de aplicação do SAEB.

Com isso ficou evidente que o apoio computacional deveria ser parcial, com o objetivo de facilitar a decisão humana e não de automatizar o processo por completo, logo, decidiu-se utilizar uma técnica de visualização para permitir que o usuário analisasse todos os dados da base e, além de realizar a padronização sintática das questões, também pudesse abstrair rapidamente novas informações sobre os diversos questionários e conhecer melhor a estrutura das avaliações.

O Processo para Definição da Linguagem para Integração de Dados Educacionais (P-LIDE) e a SEV-Tool compõem um apoio para o tratamento dos dados sobre as avaliações e pode ser aplicado a todas as bases do INEP. Já o ONTO*logy Process* (ONTOP) e a ONTO-Tool compõem um apoio para a conceitualização de ontologias, independentemente do domínio ao qual ela se refere. Sendo que o uso da visualização como recurso para tratamento desses dados facilitou substancialmente a realização das tarefas propostas por ambos os processos.

Como resultado deste trabalho obteve-se a criação da LIDE do SAEB que está sendo utilizada na plataforma Web-PIDE e a geração de uma lista de termos que compõe um glossário que dará suporte à criação de uma ontologia para as avaliações do INEP. Além disso, as ferramentas irão apoiar a definição da LIDE para as outras bases do INEP e ajudar a equipe do projeto no entendimento das estruturas das avaliações realizadas pelo INEP. Como as ferramentas criadas não possuem interface Web, apenas *desktop*, então elas não poderão ser integradas (permitir o acesso de usuários do portal a ferramenta) na plataforma Web-PIDE, servindo, principalmente, como suporte à equipe do projeto no desenvolvimento da plataforma.

Data Webhouse do SAEB e ferramenta de consulta ao dados

O Data Webhouse[21] do SAEB foi construído pelo aluno Dinísio de Machado Leite - UFMS/Coxim, sob orientação da Profª Ma. Leila Lisiane Rossi - UFMS/Coxim, em parceria com os outros bolsistas e professores do projeto.

A metodologia adotada para criação do DW SAEB [40] foi, primeiramente, preparar a base de dados do SAEB, que havia sido carregada utilizando a ferramenta DEAR como suporte, e a partir dela, utilizando a ferramenta JUDE[2][37], criar o cubo e as consultas OLAP[3][51].

Na Figura A.6 é possível observar o cubo SAEB que é composto pela tabela `Fatos` centralizada contendo as medidas necessárias para a realização das consultas OLAP, além das tabelas `idade`, `ano_aplicacao`, `sexo`, `cor`, `dependencia_adm`, `perfil_socioeconomico`, `aluno`, `localizacao` e `disciplina`, que representam as dimensões do cubo.

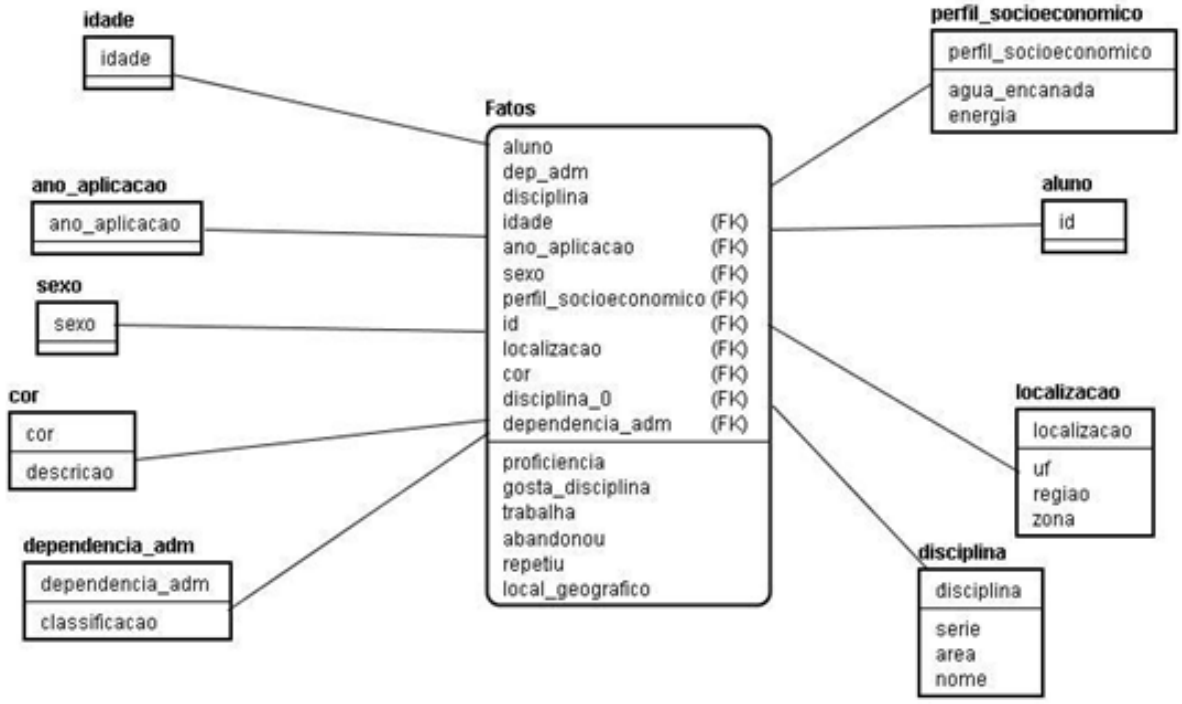


Figura A.6: Cubo SAEB gerado utilizando JUDE/Java [40].

Com o DW finalizado e utilizando tecnologias Java, Mondrian[4][45] e Jpivot foi desenvolvido uma ferramenta Web para consulta aos dados do DW na forma de tabelas e

[2]JUDE é uma ferramenta para projeto de software que suporta UML, Diagrama de Entidade-Relacionamento, Fluxograma, CRUD, Diagrama de Fluxo de Dados, entres outros.

[3]OLAP é um acrônimo para *On Line Analytical Processing*. OLAP executa análises multidimensionais de dados de negócio e fornece com aptidão cálculos complexos, análise de tendências e modelagem sofisticada de dados.

[4]Mondrian é um Servidor OLAP (*On-Line Analytical Processing*) escrito em Java. Ele permite analizar interativamente grandes conjuntos de dados em bases de dados SQL sem a necessidade de especificar consultas SQL.

gráficos. O endereço de acesso a ferramenta é o http://200.192.127.129:8180/webpide.

Como resultado da contribuição deste trabalho para o projeto, observou-se que a criação de um *Data Webhouse* é uma tarefa extremamente complexa considerando, principalmente, bases com grandes volumes de dados e a necessidade do conhecimento de várias áreas da computação como a Engenharia de Software, Banco de Dados, Análise e Projeto de Software além do domínio de ferramentas e tecnologias para a disponibilização dos resultados na Web. Além disso, este trabalho serviu como base para a criação dos *Data Webhouses* das outras bases do INEP, como é o caso do trabalho que será apresentado a seguir, pois o processo de criação do DW envolveu muita análise dos dados e diversas inconsistências foram encontradas, sendo elas registradas em relatórios técnicos.

Data Webhouse do CEB e ferramenta de consulta ao dados

O DW do CEB foi construído inicialmente pelo aluno Dinísio de Machado Leite e finalizado pelo aluno Fernando Maia da Mota, sob orientação da Profª Ma. Leila Lisiane Rossi, em parceria com os bolsistas e professores do projeto.

A metodologia adotada para criação do DW CEB [47] foi, primeiramente, criar um modelo hierárquico para a base de dados do CEB usando a ferramenta Cmap Tools[5][16] para um melhor entendimento da estrutura dos dados, conforme ilustra a Figura A.7.

Em seguida, a partir de um relatório de inconsistências da base de dados do CEB e por meio do dicionário de dados da base foi criado o cubo e as consultas OLAP, usando como suporte a ferramenta JUDE e a modelagem multidimensional estrela na definição do cubo, conforme ilustra a Figura A.7. Nela é possível observar que o cubo é composto pela tabela `fato_ensino_escola` centralizada contendo as medidas necessárias para a realização das consultas OLAP, além das tabelas `pre_escolar`, `ens_fundamental`, `geografia`, `tempo`, `escola` e `ens_medio`.

Por fim, utilizando tecnologias Java, Mondrian e Pentaho *Business Intelligence Server*[56], foi criada uma ferramenta Web intitulada "Pentaho/Web-PIDE" e disponibilizada no endereço http://webpide.ledes.net:8180/pentaho para consulta aos dados do DW do CEB, sendo possível a geração de relatórios e visualização dos resultados em tabelas e gráficos, com a possibilidade de armazenar tais resultados para posterior acesso, sem necessidade de nova consulta ao DW.

Este trabalho contribuiu com o andamento do projeto já que agora estão criados os *Data Webhouses* do SAEB e CEB além de ferramentas para consultá-los, faltando agora a integração destas ferramentas na plataforma Web-PIDE, que é um dos objetivos da proposta do presente trabalho de mestrado. Além disso, foram disponibilizados relatórios técnicos de inconsistências das bases SAEB e CEB que ajudarão na geração dos *Data Webhouses* das outras bases do INEP.

Os *Data Webhouses* do SAEB e CEB criados não representam efetivamente todo os dados do CEB e SAEB pois eles foram criados para responder a um conjunto restrito de

[5]Cmap Tools é uma ferramenta que permite aos usuários construir, navegar, compartilhar e criticar modelos de conhecimento representados por mapas conceituais.

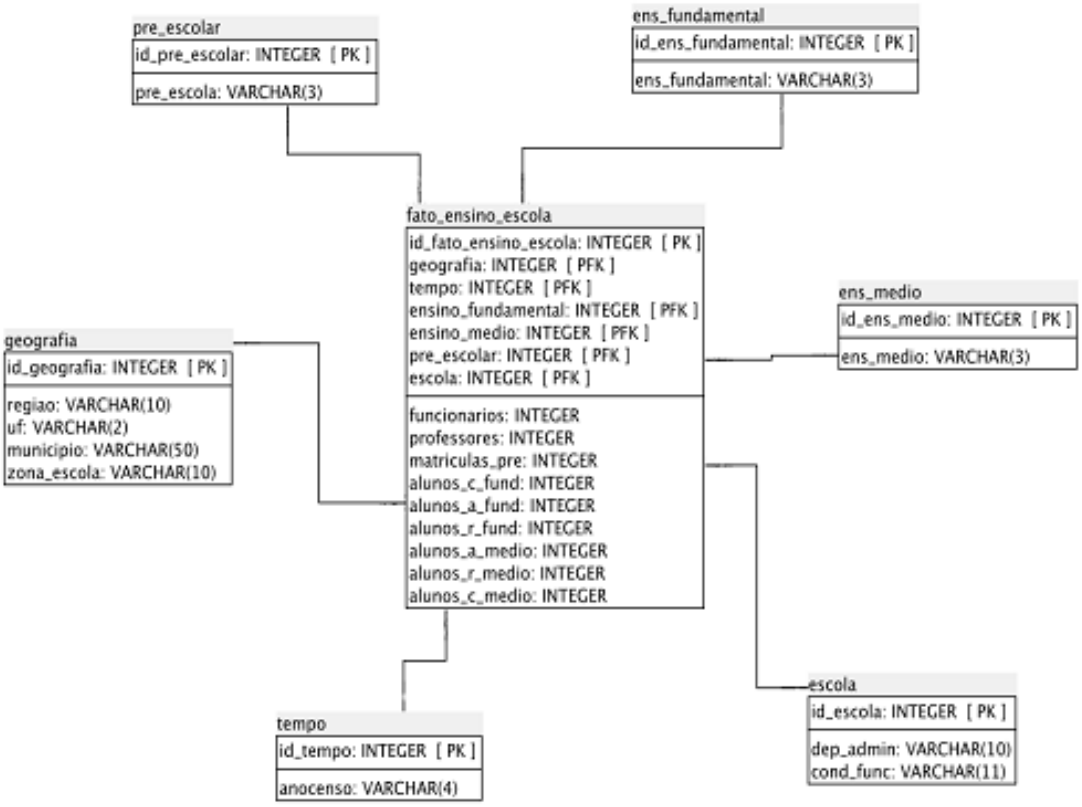


Figura A.7: Cubo CEB gerado utilizando JUDE/Java [47].

questões sobre os dados. Portanto, eles devem ser considerados na verdade *Data Marts* que podem ser evoluídos ou associados com outros *Data Marts* para somente então serem considerados *Data Webhouses*.

Ferramenta Amb-PIDE

Como até o momento o INEP não possui nenhuma ferramenta para visualização, em formato georeferenciado, dos indicadores educacionais brasileiros, logo foi desenvolvida a aplicação intitulada "Ambiente de Integração de Dados — Amb-PIDE" [62]. Ela foi criada pelo aluno Jackson Dias Savitraz - UFMS sob orientação do Prof. Dr. Marcelo Augusto Santos Turine e disponibilizada no endereço http://webpide.ledes.net/amb-pide/.

Para construir tal ferramenta foi necessária a criação de um *Web Service* para realizar a integração dos dados educacionais, mantidos no repositório do Projeto WebPIDE, com a ferramenta propriamente dita. Neste *Web Service* foram implementados três métodos: *getDatabases* - que retorna uma listagem, em formato XML, das bases de dados disponíveis para consultas; *getColumns* - que retorna as informações disponíveis sobre os indicadores disponíveis em uma determinada base de dados; e *getData* - que retorna a listagem dos indicadores da base de dados selecionada que correspondem aos critérios, quando especificados.

Utilizando os métodos descritos anteriormente e consultas aos *Data Webhouses* da

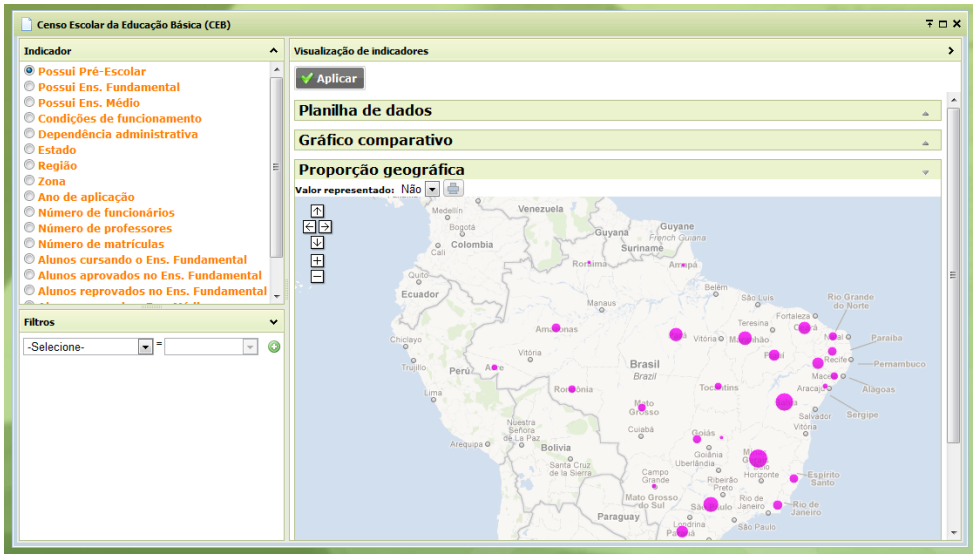


Figura A.8: Interface Web da ferramenta Amb-PIDE. Adaptada da dissertação do Jackson Dias Savitraz [62]

bases do INEP, o *Web Service* consegue gerar dados estruturados, em formato XML, que podem ser utilizados por qualquer aplicação Web que utilize o *Web Service* e siga o protocolo definido para a execução dele. Na Figura A.8 pode-se observar, a partir da execução da aplicação AmbPIDE, que após escolha de um indicador e a aplicação de filtros é possível obter resultados em formatos tabulares, gráficos e georeferenciados dos indicadores educacionais de determinada avaliação do INEP, assim como também, exportar tais resultados em planilhas XLS.

Com o desenvolvimento desta ferramenta foi possível realizar estudos comparativos, em determinadas hipóteses, executados com duas bases de dados do INEP, SAEB e CEB. Com a inclusão das demais avaliações (ENEM, ENADE, ...) no repositório do Web-PIDE e a geração dos *datawarehouses* destas bases será possível ampliar ainda mais este horizonte de estudos comparativos cruzando informações do Ensino Básico com o Ensino Médio, por exemplo.

Ferramenta de Consulta aos Dados do CEB utilizando o padrão *Linked Open Data*

A pesquisa realizada pelo aluno Fernando Maia da Mota da UFMS-Coxim, orientado pela Profª. Ma. Karen Kiomi Nakasato da UFMS-Coxim e coorientado pelo Prof. Dr. Marcelo Augusto Santos Turine produziu um artigo intitulado "Uma Estratégia para Publicação dos Dados da Base CEB-INEP no Padrão Linked Open Data" [46].

Para concepção da estratégia proposta no trabalho, partiu-se do conhecimento sobre o CEB adquirido durante o desenvolvimento de um trabalho de iniciação científica pelo próprio aluno e apoiado pela pesquisa de mestrado realizada pela aluna Elis Cristina Montoro Hernandes que demonstrou a falta de padrão no que se refere as questões formuladas para o censo ao longo dos anos. Esse cenário ocasiona grandes dificuldades ao se tentar

reutilizar as mesmas estruturas de pesquisa em anos diferentes, uma vez que as questões sofrem grandes alterações ao longo dos anos. Desta forma o padrão *Linked Open Data* se mostra como uma possibilidade real para a solução deste problema, uma vez que este padrão se caracteriza pela publicação de dados utilizando conceitos de Web semântica através de triplas RDF e ontologias que descrevem clara e objetivamente os dados.

Dessa forma, foi concebida a estratégia para a publicação de dados que parte da carga dos microdados do INEP para relações do PostgreSQL, em seguida essas relações foram avaliadas para criação de um modelo lógico relacional normalizado de banco de dados, que verificou ser pré-requisito de entrada das ferramentas STDTRIP e TRIPLIFY, que foram contruídas para dar suporte ao processo de publicação de dados no padrão *Linked Open Data*. Sendo que a ferramenta STDTRIP desenvolvida na PUC-Rio visa a criação e reuso de ontologias e a ferramenta TRIPLIFY que, de fato, implementa a triplicação dos dados.

Para publicação das triplas RDF do CEB são necessárias várias etapas e a Figura A.9 ilustra e descreve estas etapas que ocorrem de forma sequencial.

A contribruição deste trabalho para o projeto rendeu a apresentação do artigo no IV Congresso Internacional de Software Livre e Governo Eletrônico (CONSEGI) em 2011. Além disso, foi desenvolvimento uma ferramenta para consulta das triplas RDF da base do CEB e um processo que poderá ser aplicado as outras bases do INEP que compõem a plataforma Web-PIDE.

Comparação de Algoritmos Paralelos para a Extração de Regras de Associação

O aluno Marcos Alvez Mariano da UFMS-Campo Grande sob orientação do Prof. Dr. Henrique Mongelli da UFMS-Campo Grande realizou, em seu trabalho de mestrado, um estudo comparativo entre três algoritmos de mineração de dados Apriori, Eclat e FP-Growth para determinar qual deles é o mais eficiente para a tarefa de extração de regras de associação considerando a execução com diferentes quantidades de dados, assim como também a execução com 1 até 32 processadores em paralelo.

Nos experimentos realizados, concluiu-se que, dentre os algoritmos paralelos Apriori, Eclat e FP-Growth, o algoritmo Apriori foi o que exigiu maiores tempos de execução para qualquer volume de dados submetido em um processamento comparativo. Porém, o algoritmo Apriori foi o que apresentou uma melhora mais acentuada no desempenho, conforme aumenta-se o número de processadores, quando é requerido um processamento com grandes quantidades de dados.

O algoritmo Eclat, por não utilizar comunicação entre os processadores envolvidos, foi o algoritmo que apresentou uma maior redução no tempo de processamento, conforme aumenta-se o número de processadores, quando utilizados volumes de dados considerados pequenos. Porém, também foi o algoritmo que obteve menor desempenho nos experimentos realizados com grandes quantidades de dados.

Já o algoritmo FP-Growth não apresentou reduções de tempo de processamento, com pequenas quantidades de dados, quando comparados aos tempos do algoritmo Eclat.

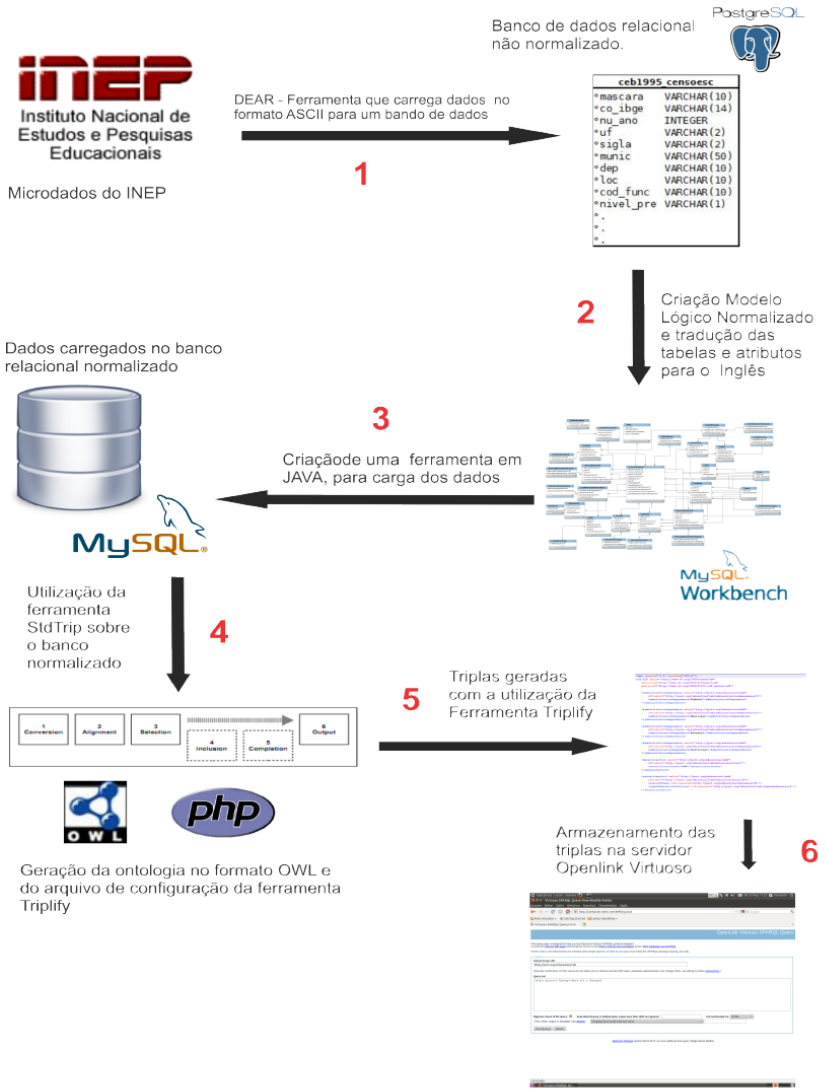


Figura A.9: Etapas para Publicação dos Dados no Padrão *Linked Open Data* [46].

Porém, dentre os algoritmos, foi o que obteve o menor tempo de processamento quando executado com grandes quantidades de dados. Com relação a ganhos de desempenho, o algoritmo FP-Growth apresentou resultados próximos aos obtidos pelo algoritmo Apriori com grandes volumes de dados.

Como contribuição ao projeto Web-PIDE este trabalho sugeriu que o algoritmo Apriori poderia ser utilizado na criação de uma aplicação que permita a extração de regras de associação utilizando os dados das bases do INEP, que por acaso são de grande volume. Com isto, desde que a execução da aplicação ocorra em paralelo e com muitos processadores, regras de associação poderiam ser extraídas destes dados de forma mais eficiente do que com os outros algoritmos paralelos de mineração de dados Eclat e FP-Growth.

Ferramenta para Gestão de Planejamento Educacional

A aluna Fernanda Aparecida Rocha da Silva da UFSCar sob orientação da Profa. Dra. Sandra Camargo Pinto Ferraz Fabbri desenvolveu, em seu trabalho de mestrado [20], uma ferramenta baseada em serviços para dar suporte à atividade de planejamento e controle, uma vez que para atingir as metas de desempenho definidas pelo governo brasileiro, com base nos dados das avaliações do INEP, o gestor educacional deve traçar um plano de ação para sua unidade. Ferramenta esta que foi desenvolvida especialmente para a plataforma Web-PIDE.

Além disso, ela definiu também uma estratégia para identificação e determinação de serviços reutilizáveis tanto na evolução da plataforma Web-PIDE como de outras aplicações. Sendo que esta estratégia foi aplicada na plataforma Web-PIDE e os serviços gerados, para compor a plataforma, forneceram suporte ao processo de planejamento em diferentes níveis, ou seja, no nível estratégico, tático e operacional da criação do plano de ação.

Para agregar os serviços gerados alguns módulos foram desenvolvidos e cada um deles são descritos abaixo:

- **Gestão Estratégica**: permite o cadastro do planejamento, que produzirá um mapa estratégico, uma previsão de custo, além do cronograma, em alto nível, das atividades vinculadas a um plano de ação;
- **Indicadores do INEP**: apresenta um gráfico com IDEB (Índice de Desenvolvimento da Educação Básica), além de links que redirecional o usuário para portais do INEP relacionados a consultas sobre avaliações como o SAEB e a Prova Brasil;
- **Consulta às Bases do INEP**: disponibiliza dois tipos de consultas. O primeiro tipo consiste em um conjunto de palavras-chave que é utilizado para pesquisar nas LIDEs se existem questões das avaliações do INEP relacionadas ao conjuto de palavras-chave utilizado. Como resultado uma lista de questões referentes ao conjunto de palavras-chave pesquisado é retornado e ao escolher uma das questões o sistema exibe um gráfico com os resultados. No segundo tipo de consulta, o sistema permite a escolhe de qual base do INEP e avaliação serão utilizadas, além de escolher qual tipo de gráfico será gerado. Em seguida, após uma busca utilizando uma ou mais palavras-chave o sistema retorna uma lista de questões relacionadas. Ao escolher uma questão o sistema gera o gráfico corresponde aos resultados obtidos.
- **Histórico**: apresenta um histórico dos planejamentos já concluídos, de forma que o usuário possa pesquisar sobre a documentação desses planejamentos, ou seja, mapas estratégicos utilizados, planos de ação, avaliações das estratégias aplicadas, atividades desenvolvidas para atingir os objetivos do planejamento, dificuldades encontradas para desenvolver alguma atividade, etc.
- **Relatórios Diversos**: gera relatórios sobre o desempenho das instituições de ensino nas avaliações do INEP. Podendo gerar relatórios de (1) estágio de desempenho, variando de muito crítico até excelente, em uma determinada série e ano; (2) nota

média obtida por uma determinada série ao longo dos anos de aplicação das avaliações do INEP; (3) desempenho de uma determinada turma em relação ao número de acerts e erros em cada questão da avaliação de uma determinada disciplina e; (4) comparação entre duas ou mais instituições de ensino em uma mesma região, em relação ao desempenho em um determinada disciplina.

- **Gestão Estratégica**: acessa ao cadastro, edição, validação do planejamento, além de pesquisar sobre os planos de ação em andamento. Os planejamentos são definidos com base no *Balanced Scorecard* (BSC) com opções de analisar pendências no plano estratégico, analisar o cronograma de execução dos planos, monitorar indicadores definidos no plano estratégico e analisar os indicadores de desempenho segundo as perpectivas do *Balanced Scorecard* (BSC).

Ferramenta *WPS-Cartographer*

O aluno Vinícius Ramos Toledo Ferraz da UFSCar sob orientação da Profa. Dra. Marilde Terezinha Prado Santos realizou, em seu trabalho de mestrado [27], um estudo investigativo quanto à criação de Ontologias Cartográficas que auxiliem usuários de Sistemas de Informação Geográfica na escolha da linguagem cartográfica que compõe um mapa temático geográfico. Dessa forma, evita-se que usuários leigos em cartografia tomem decisões equivocadas, gerando mapas ineficientes ou até mesmo mentirosos.

Com isto, o aluno definiu uma metodologia composta por uma ontologia cartográfica, intitulada *OWL-Cartography*, uma heurística de raciocínio baseada na ontologia proposta e um serviço Web denominado *WPS-Cartographer*. Para avaliar a medologia foi realizado um estudo de caso seguido de análise estatística em um cenário típico de Projeto Cartográfico Temático.

Como contribuição deste trabalho para o projeto Web-PIDE verificou que o uso de uma ontologia cartográfica para auxiliar na escolha da linguagem cartográfica que compõe um mapa temático é importante e a ontologia *OWL-Cartography* poderia ser utilizada na ferramenta Amb-PIDE facilitando ainda mais o processo de geração dos mapas temáticos pelos usuários dela. Ou então, a ferramenta *WPS-Cartographer* poderia ser adaptada para o domínio da plataforma Web-PIDE para permitir a geração de mapas temáticos utilizando dados das bases do INEP.

Resultados Alcançados e Esperados do Projeto

Apesar do grande volume de dados que as avaliações do INEP produzem e da pouca padronização dos dados produzidos, pôde-se alcançar resultados satisfatórios no decorrer da execução do projeto como a construção dos *Data Webhouses* do SAEB e CEB (que por enquanto representam apenas *Data Marts* que podem ser evoluídos para *Data Warehouses*) e das ferramentas para visualização tabular, gráfica e georeferenciada de consultas realizadas sobre os DW.

A plataforma Web-PIDE facilitará o acesso aos dados do INEP e a comunidade e os gestores poderão ter um local único para gerenciar o volume de dados produzidos pelas avaliações do INEP.

Para que seja alcançado este objetivo é necessário realizar a integração das ferramentas produzidas pela equipe do projeto na plataforma Web-PIDE, sendo que o processo proposto neste trabalho apoiará esta tarefa. A Tabela A.1 representa uma análise destas ferramentas quanto aos critérios listados a seguir para realizar tal integração.

- **Qual é o tipo de plataforma da ferramenta**: (Desktop) indicando que a ferramenta foi desenvolvida para ser executada por meio do Desktop; (Web) indicando que a ferramenta foi desenvolvida para ser executada por meio da Web;
- **A ferramenta possui acesso restrito**: este critério indica se a ferramenta possui funcionalidades que precisam de autenticação para serem executadas ou não;
- **Quais tecnologias foram utilizadas no desenvolvimento da ferramenta**: este critério indica quais principais tecnologias foram utilizadas na construção da ferramenta tais como: linguagem de programação, banco de dados e demais.
- **A ferramenta está integrada na plataforma Web-PIDE**: este critério indica se a ferramenta já está integrada na plataforma Web-PIDE ou não.

Na Tabela A.1 é possível observar que ferramentas do tipo *Desktop* não estão integradas na plataforma Web-PIDE já que a plataforma é Web e inviabiliza tal integração até o momento (existem trabalhos futuros para possibilitar a integração de ferramentas *desktop* à plataforma). Por outro lado, algumas ferramentas do tipo *Web* já estão integradas na plataforma, como por exemplo, Portal Web-PIDE, VIDEWeb e Amb-PIDE. Com isso, somente as ferramentas que possuem acesso restrito e são do tipo *Web* ainda não estão integradas na plataforma Web-PIDE, sendo elas fortes candidatas a utilizarem o processo PIBSAWL proposto neste trabalho.

Considerações Finais

Neste capítulo foi apresentado o projeto Web-PIDE, os trabalhos já finalizados no projeto e os aplicações desenvolvidas. Cada uma destas aplicações foram analisadas baseado em quatro critérios para serem utilizadas como estudos de caso do presente trabalho e a aplicação escolhida foi a Pentaho/Web-PIDE.

A Pentaho/Web-PIDE foi escolhida por ser uma das aplicações do projeto Web-PIDE que possui acesso restrito, suporte à tecnologia LDAP e que não está integrada à plataforma Web-PIDE.

As outras ferramentas de acesso restrito do Web-PIDE (possíveis candidatas à integração) e que ainda não estão integradas à plataforma Web-PIDE poderão ser facilmente integradas à plataforma pelos seus desenvolvedores utilizando o PIBSAWL e o AIA, já

Aplicação	**Tipo de Plataforma**	**Possui Acesso Restrito**	**Tecnologias Utilizadas no Desenvolvimento**	**Integrada à Plataforma Web-PIDE**
Portal Web-PIDE	Web	Sim	PHP e XML	Sim
DEAR	Desktop	Não	Java	Não
VIDEWeb	Web	Não	Java e JPivot	Sim
SEV-Tool	Desktop	Não	Java	Não
ONTOP-Tool	Desktop	Não	Java	Não
Consulta ao Data Webhouse do SAEB utilizando Mondrian	Web	Não	Java, JPivot e Mondrian	Não
Consulta ao Data Webhouse do CEB utilizando Pentaho B.I. Server	Web	Não	Java e Pentaho Business Intelligence Server	Não
Amb-PIDE	Web	Sim	PHP e XML	Sim
Consulta aos Dados do CEB utilizando o padrão Linked Open Data	Web	Não	Java e Virtuoso Universal Server	Sim
Gestão de Planejamentos Educacionais	Web	Sim	Java	Não
WPS-Cartographer	Web	Não	XML, JSON[6]	Não

Tabela A.1: Análise das ferramentas produzidas pela equipe do projeto Web-PIDE quanto aos critérios para integrá-las na plataforma Web-PIDE.

que estes possuem o domínio necessário para realizar adaptações nestas ferramentas e o processo PIBSAWL exige tais adaptações e conhecimento básico das tecnologias de desenvolvimento utilizadas na aplicação.

Referências Bibliográficas

[1] Alkouz, A., and El-Seoud, S. A. Web services based authentication system for e-learning. *Proceedings of the International Journal of Computing & Information Sciences* (2007), 74–78.

[2] Amaral, B. S. Ldap: Centralização e disponibilização de informações. Disponível em http://www.ricardoterra.com.br/publications/students/2010_amaral.pdf [05/2011], Dezembro 2010.

[3] Andrade, L. F., and Fiadeiro, J. L. Coordination technologies for web-services. In *OOPSLA: Workshop on Object-Oriented Web Services* (Tampa, Florida, EUA, 2001).

[4] Andrew, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., and Weerawarana, S. Business process execution language for web services, 2003. Disponível em http://www.ibm.com/developerworks/library/specification/ws-bpel/ [04/2010].

[5] Arakaki, A. A. Gerência de projetos de software livre no framework safe. Master's thesis, Universidade Federal de Mato Grosso do Sul, Campo Grande, MS, Brasil, 2008.

[6] Arnold, K., Scheifler, R. W., Waldo, J., Wollrath, A., Scheifler, R., and O'Sullivan, B. *The Jini Specification*. Addison-Wesley Pub, 1999.

[7] Bastos, H. A. Siga brasil: Tecnologia da informação a serviço da eficiência, transparência e controle social do gasto público. In *Senatus* (Brasília, DF, Brasil, 2009), vol. 7, Secretaria de Informação e Documentaçao/Senado Federal, pp. 87–91.

[8] Bonamino, A., and Franco, C. Avaliação e política educacional: O processo de institucionalização do saeb. *Cadernos de Pesquisa 108* (1999), 101–132.

[9] Brasil. Plano de desenvolvimento da educação (pde). Tech. rep., Ministério Público e Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira, Brasília, DF, Brasil, 2008.

[10] Burbeck, S. The evolution of web applications into service-oriented components with web services. Disponível em

http://www.ibm.com/developerworks/webservices/library/ws-tao/ [04/2010], 2000.

[11] Capes. Fundação coordenação de aperfeiçoamento de pessoal de nível superior - capes. Disponível em http://www.capes.gov.br/ [11/2009].

[12] Carromeu, C. Instanciação de sistemas de informação para educação a distância com o framework titan. Disponível em http://wiki.ledes.net/images/Titan_EaD_2009.pdf [03/2011], 2009.

[13] Carromeu, C., Beirigo, M. G., Salomão, O. M., and Ferencz, R. Z. Titan platform: Uma plataforma de desenvolvimento de webapps utilizando o titan framework. Disponível em http://wiki.ledes.net/images/Titan_Platform_Monografia_2010.pdf [03/2011], 2010.

[14] Carromeu, C., and Turine, M. A. S. Um framework de aplicações web de apoio a gestão de fomento. In *Workshop de Teses e Dissertações do XI Simpósio Brasileiro de Sistemas Multimídia e Web - WebMedia* (Poços de Caldas, MG, Brasil, 2005).

[15] Cerqueira, C. A., and Rigotti, J. I. R. As bases de dados do inep e os indicadores educacionais: conceitos e aplicações. In *International Union for Scientific Study of Population - IUSSP* (Salvador, BA, Brasil, 2001).

[16] CmapTools. Cmap tools - knowledge modelling kit. Disponível em http://cmap.ihmc.us/ [01/2010].

[17] CUMMINS, F. A. *Integração de Sistema*. Campus, 2002.

[18] Cunha, M. X. C., Júnior, M. F. S., and Dornelas, J. S. O uso da arquitetura soa como estratégia de integração de sistemas de informação em uma instituição pública de ensino. *Simpósio de Excelência em Gestão e Tecnologia (SEGeT)* (2008).

[19] Curbera, F., Nagy, W. A., and Weerawarana, S. Web services: Why and how. In *OOPSLA: Workshop on Object-Oriented Web Services* (Tampa, Florida, EUA, 2001).

[20] da Silva, F. A. R. Identificação e determinação de serviços para compor a plataforma web-pide. Master's thesis, Universidade Federal de São Carlos, 2011.

[21] de Holanda, L. M. C., Guedes, I. A., da Rocha, R. C., and dos Santos, D. M. B. Data webhouse: A evolução do data warehouse para web e suas contribuições para o aperfeiçoamento do relacionamento com clientes. In *Simpósio de Excelência em Gestão e Tecnologia - SEGet* (Rio de Janeiro, RJ, Brasil, 2005).

[22] DW. Data warehouse. Disponível em http://en.wikipedia.org/wiki/Datawarehouse [03/2010].

[23] Engels, G., Hess, A., Humm, B., Juwig, O., Lohmann, M., Richter, J.-P., Voss, M., and Willkomm, J. A method for engineering a true service-oriented architecture. *10th International Conference on Enterprise Information Systems* (2008).

[24] Erl, T. *Service-Oriented Architecture: Concepts , Technology and Design*. Prentice Hall, 2005.

[25] Erl, T. *SOA: princípios de design de serviços*. Prentice Hall Brasil, 2008.

[26] Fayçal, H., Habiba, D., and Hakima, M. Integrating legacy systems in a soa using an agent based approach for information system agility. In *Conference International on Machine and Web Intelligence (ICMWI)* (2010).

[27] Ferraz, V. R. T. Simbolização de mapas temáticos utilizando uma ontologia cartográfica. Master's thesis, Universidade Federal de São Carlos, 2011.

[28] Ferreira, M., Ramos, G., Bernardo, L., Silva, R., and Maior, B. Manual de instalação e utilização da plataforma pentaho de business intelligence, 2010. Disponível em http://wiki.softwarelivre.org/pub/PentahoBrasil/Documentos/Pentaho_3_5.pdf [07/2010].

[29] Han, H., and Tokuda, T. A method for integration ofweb applications based on information extraction. In *Eighth International Conference on Web Engineering* (2008).

[30] Hernandes, E. C. M. Um processo automatizado para tratamento de dados e conceitualização de ontologias com apoio de visualização. Master's thesis, Universidade Fedederal de São Carlos, São Carlos, SP, Brasil, 2009.

[31] Html. Hyper text markup language (html) 4.01 specification. Disponível em http://www.w3.org/TR/REC-html40/ [12/2009].

[32] Ierusalimschy, R., de Figueiredo, L., , and Celes, W. Lua 5.1, reference manual. Disponível em http://www.lua.org/ [03/2010].

[33] Inep. Instituto nacional de estudos e pesquisas educacionais anísio teixeira. Disponível em http://www.inep.gov.br/ [10/2009].

[34] Jannuzzi, P. M. *Indicadores Sociais no Brasil: Conceitos, Fontes de Dados e Aplicações*. Alínea, 2001.

[35] JavaBeans. Enterprise javabeans specification version 2.0. Disponível em http://java.sun.com/products/ejb/docs.html [03/2010].

[36] JPivot. Jpivot. Disponível em http://jpivot.sourceforge.net/ [01/2010].

[37] Jude. Jude. Disponível em http://jude.change-vision.com/ [01/2010].

[38] Kruchten, P. *The Rational Unified Process: An Introduction*, second ed. ed. Addison-Wesley, 2000.

[39] Lattes. Plataforma lattes do cnpq. Disponível em http://lattes.cnpq.br/ [01/2010].

[40] Leite, D. M., and Rossi, L. L. Estudo e avaliação das técnicas e das ferramentas para o projeto de data warehouse do saeb/inep. Tech. rep., Universidade Federal de Mato Grosso do Sul, Coxim, MS, Brasil, 2010. Disponível em http:/webpide.ledes.net/ [07/2010].

[41] Lopes Siqueira, T. L., Ciferri, R. R., Times, V. C., and de Aguiar Ciferri, C. D. A spatial bitmap-based index for geographical data warehouses. In *Proceedings of the 2009 ACM symposium on Applied Computing* (New York, NY, USA, 2009), SAC '09, ACM, pp. 1336–1342.

[42] Machado, M. C. C., Santos, M. S., and Gondim, R. P. Videweb: Um ambiente para visualização de dados educacionais na plataforma web-pide. Disponível em http://webpide.ledes.net/ [03/2009], 2007.

[43] Martin, D., Burstein, M., Lassila, O., Hendler, J., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., and Sycara, K. Owl-s: Semantic markup for web services, 2004. Disponível em http://www.w3.org/Submission/OWL-S/ [04/2011].

[44] Mec. Ministério da educação. Disponível em http://www.mec.gov.br/ [10/2009].

[45] Mondrian. Pentaho analysis services (mondrian). Disponível em http://mondrian.pentaho.org/ [01/2010].

[46] Mota, F. M. Uma estratégia para publicação dos dados da base ceb-inep no padrão linked open data. In *IV Congresso Internacional de Software Livre e Governo Eletrônico* (2011).

[47] Mota, F. M., and Rossi, L. L. Estudo e avaliação das técnicas e das ferramentas para o projeto de data warehouse do ceb/inep. Tech. rep., Universidade Federal de Mato Grosso do Sul, Coxim, MS, Brasil, 2010. Disponível em http:/webpide.ledes.net/ [07/2010].

[48] Muller, J. L., Buchner, A., and Muller, P. A framework for the requirements analysis of service-oriented workflows. In *Proceedings of the Third International Conference on Next Generation Web Services Practices* (Seoul, South Korea, 2007), pp. 104 – 109.

[49] Nacer, J. F. R. Titan architect: Um wizard para instanciação de gerenciadores de conteúdo utilizando o titan framework. Disponível em http://wiki.ledes.net/images/Titan_Architect_Monografia_2008.pdf [03/2011], 2009.

[50] Observatório. Programa observatório da educação - capes/inep. Disponível em http://www.capes.gov.br/bolsas/especiais/observatorio_educacao.html [10/2009].

[51] Olap. On line analytical processing. Disponível em http://www.olap.com/ [01/2010].

[52] OpenLDAP. Openldap - community developed ldap software. Disponível em http://www.openldap.org/ [04/2010].

[53] Osgi. Osgi service platform specification 4. Disponível em http://www.osgi.org/Specifications/HomePage [03/2010].

[54] Peltz, C. Web services orchestration and choreography. *Journal Computer 36*, 10 (2003), 46–52.

[55] Peltz, C. Web services orchestration and choreography. *Computer 36*, 10 (oct. 2003), 46 – 52.

[56] Pentaho. Pentaho business intelligence server. Disponível em http://sourceforge.net/projects/pentaho/ [01/2010].

[57] Php. Php hypertext preprocessor. Disponível em http://www.php.net/ [12/2009].

[58] PostgresSQL. Manual do postgresql 8.1.11. Disponível em http://www.postgresql.org/files/documentation/pdf/8.1/postgresql-8.1-A4.pdf [12/2010].

[59] Quartel, D. A. C., Pires, L. F., van Sinderen, M. J., Franken, H. M., and Vissers, C. A. On the role of basic design concepts in behaviour structuring. *Journal Computer Networks and ISDN Systems 29* (1997), 413–436.

[60] Reussner, R., and Hasselbring, W. *Handbuch der Software-Architektur*. dpunkt Verlag, 2006.

[61] Sarmento, W. W. F. Integração de um ambiente virtual de aprendizagem com aplicações móveis de suporte a educação a distância. Master's thesis, Universidade Federal do Ceará, 2007.

[62] Savistraz, J. D. Uma abordagem de integração e exploração visual de dados educacionais na plataforma webpide. Master's thesis, Universidade Federal de Mato Grosso do Sul, Campo Grande, MS, Brasil, 2010.

[63] Sigfap. Manual de usuário do sigfap, 2011. Disponível em http://fundect.ledes.net/?section=sigfap&itemId=4 [03/2011].

[64] Siqueira, T. L. L. Sb-index: Um Índice espacial baseado em bitmap para data wharehouse geográfico. Master's thesis, Universidade Federal de São Carlos/SP, 2009.

[65] SIQUEIRA, T. L. L., CIFERRI, R. R., and SANTOS, M. T. P. Projeto, construção e manutenção de data warehouses para auxiliar o planejamento de políticas públicas de educação. In *XVI Jornadas de Jóvenes Investigadores* (Montevideo, Uruguay, 2008), pp. p. 1016–1025.

[66] Smith, D. Migration of legacy assets to service-oriented architecture environments. In *Companion to the proceedings of the 29th International Conference on Software Engineering* (Washington, DC, USA, 2007), ICSE COMPANION '07, IEEE Computer Society, pp. 174–175.

[67] Stojanovic, Z., and Dahanayake, A. *Service-oriented Software System Engineering Challenges And Practices*. IGI Publishing, Hershey, PA, USA, 2005.

[68] Stratton, D. The omg corba trader service. Tech. rep., School of Information Technology and Mathematical Sciences, University of Ballarat, Australia, 1998.

[69] Thomas, O., Leyking, K., and Dreifus, F. Using process models for the design of service-oriented architectures: Methodology and e-commerce case study. *Hawaii International Conference on System Sciences 0* (2008), 109. International conferences;Process modelling;Service-oriented architectures;System sciences;.

[70] Tim Berners-Lee, J. H., and Lassila, O. The semantic web. *Scientific American Magazine* (2001).

[71] Trigo, C. H. *OpenLDAP: Uma abordagem integrada*. Novatec, São Paulo, SP, Brasil, 2007.

[72] Turine, M. Web-pide: Uma plataforma aberta de integração e avaliação de dados educacionais. In *Programa Observatório de Educação* (2007).

[73] Turine, M. A. S., Carromeu, C., da Silva, M. A. I., and Cagnin, M. I. Gestão pública flexível e ágil por meio do sigfap. In *ComCiência* (2011), Agencia de Notícias para a Difusão da Ciência e Tecnologia.

[74] Uddi. Uddi version 3.0 specification. Disponível em http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm [03/2010].

[75] Uml. Unified modeling language. Disponível em http://www.uml.org/ [12/2009].

[76] Uri. Uris, urls, and urns: Clarifications and recommendations 1.0. Disponível em http://www.w3.org/TR/uri-clarification/ [03/2010].

[77] Vissers, C. A., and Logrippo, L. The importance of the service concept in the design of data communications protocols. In *Proceedings of the IFIP WG6.1 Fifth International Conference on Protocol Specification, Testing and Verification* (Amsterdam, The Netherlands, The Netherlands, 1985), North-Holland Publishing Co., pp. 3–17.

[78] Wang, X., Hu, S., Haq, E., and Garton, H. Integrating legacy systems within the service-oriented architecture. In *Power Engineering Society General Meeting* (2007).

[79] Web-PIDE, P. Portal de acesso a plataforma web-pide, 2011. Disponível em http://webpide.ledes.net/ [07/2011].

[80] Wsdl. Web services description language (wsdl) 1.1. Disponível em http://www.w3.org/TR/wsdl [03/2010].

[81] Xml. Extensible markup language (xml) 1.0 specification. Disponível em http://www.w3.org/TR/REC-xml [03/2010].

[82] Zhang, B., Bao, L., Zhou, R., Hu, S., and Chen, P. A black-box strategy to migrate gui-based legacy systems toweb services. *IEEE International Symposium on Service-Oriented System Engineering* (2008).