



**UNIVERSIDADE FEDERAL DA FRONTEIRA SUL
CAMPUS DE CHAPECÓ
CURSO DE CIÊNCIA DA COMPUTAÇÃO**

DAYAN ROBERTO WEBER

**MODELO PARA INTEGRAÇÃO DE SISTEMA DE GESTÃO ACADÊMICA COM
AMBIENTE VIRTUAL DE APRENDIZAGEM MOODLE**

**CHAPECÓ
2021**

DAYAN ROBERTO WEBER

**MODELO PARA INTEGRAÇÃO DE SISTEMA DE GESTÃO ACADÊMICA COM
AMBIENTE VIRTUAL DE APRENDIZAGEM MOODLE**

Trabalho de conclusão de curso apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal da Fronteira Sul.
Orientador: Prof. Dr. Claunir Pavan

CHAPECÓ
2021

Weber, Dayan Roberto

MODELO PARA INTEGRAÇÃO DE SISTEMA DE GESTÃO
ACADÊMICA COM AMBIENTE VIRTUAL DE APRENDIZAGEM
MOODLE / Dayan Roberto Weber. – 2021.

49 f.: il.

Orientador: Prof. Dr. Claunir Pavan.

Trabalho de conclusão de curso (graduação) – Universidade Federal
da Fronteira Sul, curso de Ciência da Computação, Chapecó, SC, 2021.

1. Integração entre SGA e Moodle. 2. Modelos de integração.
3. Web services. I. Pavan, Prof. Dr. Claunir, orientador. II. Universi-
dade Federal da Fronteira Sul. III. Título.

DAYAN ROBERTO WEBER


**MODELO PARA INTEGRAÇÃO DE SISTEMA DE GESTÃO ACADÊMICA COM
AMBIENTE VIRTUAL DE APRENDIZAGEM MOODLE**

Trabalho de conclusão de curso apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal da Fronteira Sul.

Orientador: Prof. Dr. Claunir Pavan


Este trabalho de conclusão de curso foi defendido e aprovado pela banca avaliadora em 14/05/2021.

BANCA AVALIADORA



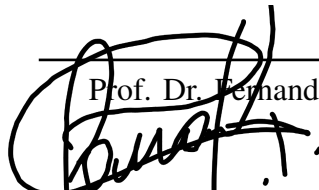
Prof. Dr. Claunir Pavan – UFFS

p/



Prof. Dr. Fernando Bevilacqua – UFFS

p/



Prof. Dr. Emilio Wuerges - UFFS

RESUMO

É constante a evolução tecnológica dos sistemas de informação, inclusive no contexto acadêmico de ensino, ao passo que as instituições de ensino superior estão aderindo as tecnologias em vista a necessidade de atender a demandas cada vez maiores, informatizando processos administrativos, a fim de prestar melhores serviços, o que favorece a proliferação do uso de ambientes virtuais de aprendizagem (AVA) em conjunto ao uso do sistema de gestão acadêmica (SGA). Entretanto, nem sempre estão disponíveis sistemas integrados que atendam totalmente as demandas institucionais, fazendo necessário uma estratégia de integração capaz de estabelecer uma comunicação eficaz. Diante disso, com base na literatura pesquisada, que dispõe de modelos de integradores aptos a proporcionar uma integração flexível e adaptável, é proposto um modelo conceitual de integração entre SGA genérico e Moodle, que utiliza da arquitetura *web services* para implementação, trazendo uma metodologia de criação do integrador através dos recursos, funcionalidades e do conjunto de APIs disponíveis na plataforma Moodle.

Palavras-chave: Integração entre SGA e Moodle. Modelos de integração. Web services.

ABSTRACT

The technological evolution of information systems is constant, including in the academic context, as the universities are adhering to technologies in order to attend the need of increasing demands, computerizing administrative processes to be able to provide better services, which favors the proliferation use of learning management system (LMS) together with the use of academic management system (AMS). However, integrated systems that fully meet institutional demands are not always available, requiring an integration strategy capable of establishing effective communication. Therefore, based on the researched literature, which has models of integrators able to provide a flexible and adaptable integration, is proposed a conceptual integration model between generic AMS and Moodle, witch utilize web services architecture for implementation, producing a integrator creation methodology through resources, functionalities and the group of APIs available in the Moodle environment.

Keywords: Integration Moodle and AMS. Integration model. Web services.

LISTA DE ILUSTRAÇÕES

Figura 1 – Arquitetura <i>web service</i>	19
Figura 2 – Como os serviços se relacionam	22
Figura 3 – Fases da utilização de <i>web services</i>	23
Figura 4 – Modelo de integração Moodle-SGA	24
Figura 5 – Modelo de integração	25
Figura 6 – Habilitar web service e protocolos	32
Figura 7 – Criação dos serviços	32
Figura 8 – Adicionar funções	33
Figura 9 – Criação novo papel	34
Figura 10 – Criação dos tokens	35
Figura 11 – Modelo de integração proposto	40
Figura 12 – Inscrição do usuário	43

SUMÁRIO

1	INTRODUÇÃO	11
1.1	TEMA E METODOLOGIA	12
2	MOTIVAÇÃO	13
3	OBJETIVOS	15
3.1	OBJETIVO GERAL	15
3.2	OBJETIVOS ESPECÍFICOS	15
4	REFERENCIAL TEÓRICO	17
4.1	INTEGRAÇÃO	17
4.2	SISTEMA DE GESTÃO ACADÊMICA	18
4.3	AMBIENTE VIRTUAL DE APRENDIZAGEM	18
4.4	MÉTODOS DE INTEGRAÇÃO	20
4.4.1	Banco a banco	20
4.4.2	Troca de dados eletrônica	20
4.4.3	<i>Application Programming Interface (API)</i>	20
4.4.4	Ferramenta intermediadora	21
4.5	ARQUITETURA ORIENTADA A SERVIÇOS	21
4.6	WEB SERVICES	22
4.7	TRABALHOS RELACIONADOS	23
5	DESENVOLVIMENTO	27
6	RESULTADO	31
6.1	MODELO DE INTEGRAÇÃO PROPOSTO	39
7	CONSIDERAÇÕES FINAIS	45
	REFERÊNCIAS	47

1 INTRODUÇÃO

É constante a evolução tecnológica dos sistemas de informação e comunicação (TIC), inclusive dentro do contexto das instituições de ensino superior, pois tendo em vista demandas cada vez maiores e exigências de docentes e estudantes, as instituições acabam por informatizar os processos e serviços administrativos. Desta forma, todo percurso acadêmico dos estudantes como também dos docentes é mantido em registros dentro de sistemas (8).

Tais sistemas de informação e comunicação desempenham um papel fundamental nas organizações e instituições de ensino, pela capacidade de fornecer infraestrutura de informação adequada, que gera melhorias na eficiência operacional e administrativa. Também segundo Silva (10), as instituições de ensino superior vêm investindo esforços e recursos para prestar bons serviços e a se manterem atualizadas, fazendo uso de sistemas de informação. Inclusive como cita (3), a utilização de TIC é um "fator determinante para obtenção de vantagem competitiva, o aumento da produtividade e essencialmente para melhoria do resultado que se espera alcançar".

Desta forma, é notável a importância da utilização de ferramentas que possibilitem a gestão e gerenciamento mais efetivo de processos e recursos da instituição, bem como a integração entre os sistemas, para benefício da instituição (10).

É nesse contexto que as instituições fazem uso de sistemas de gestão acadêmica (SGA), para gerenciar o percurso do acadêmico e docentes, em conjunto ao uso de ambientes virtuais de aprendizagem (AVA), como o Moodle, afim de melhorar a qualidade do ensino prestado e auxílio na aprendizagem dos acadêmicos (3).

Para (8) a adequação tecnológica de integração de sistemas de informação (SI) tornou-se um desafio cada vez mais complexo, devido a demanda de flexibilidade, adaptabilidade, implementação, manutenção e gestão. Inclusive deve ser analisado do ponto de vista estratégico e economista da organização, integrar sistemas que partilham a mesma informação é um caminho inevitável a seguir.

Quando analisados SGA e AVA, há muita informação que deve ser compartilhada entre eles, visto que no SGA estão as informações dos estudantes e docentes de forma mais completa, enquanto o AVA precisa apenas de uma porção dessa informação. Essa partilha de informações entre os sistemas é o que obriga a integração entre eles, para automatizar, agilizar e simplificar as tarefas de gestão (8).

Entretanto, a integração entre sistemas não é uma tarefa simples, segundo Oliveira, Correa e Fonseca (9) existe uma série de impedimentos ao integrar sistemas, dificuldades ao tratar transações e processos, principalmente quando estes atravessam a fronteira da empresa. Como o autor cita, um deles é que a maioria dos fornecedores só permite integração entre seus próprios sistemas e aplicações.

1.1 TEMA E METODOLOGIA

Este trabalho tem como proposta o estudo e caracterização de técnicas e abordagens de integração entre sistema de gestão acadêmica genérico e Moodle, para desenvolver um modelo conceitual de integrador de SGAs e AVA Moodle.

Com a finalidade de cumprir o propósito do trabalho a metodologia começa com a caracterização de técnicas disponíveis para integração entre sistemas observando aquela que se prova a mais dinâmica e capaz de cumprir o objetivo do trabalho. Após isso é descrito quais elementos precisam de integração entre os sistemas, suas características e requisitos para integração. Sabendo quais elementos precisam de integração é descrito como a integração deve ocorrer, quais funcionalidades serão usadas e por quais motivos, adicionadas as especificações de seus parâmetros, particularidades e atributos a prover a integração, simulando em ambiente Moodle a execução das tarefas a prover a integração.

2 MOTIVAÇÃO

Dentro do contexto educacional SGAs e AVAs são essenciais para a gestão e funcionamento dos serviços e recursos de qualquer instituição de ensino, entretanto, mesmo tendo conhecimento da importância dessas ferramentas, por vezes não existe qualquer comunicação entre elas e quando há, são muito limitadas, resolvendo apenas algumas demandas específicas sem haver flexibilidade de comunicação (4).

O SGA é essencial para gestão administrativa e operacional da instituição, pelo fato de ser o sistema encarregado pelos registros dos acadêmicos e docentes; nome, endereço, dados pessoais. Pelas informações curriculares; cursos, disciplinas e notas. Dados financeiros; mensalidades e serviços. Além de outras informações auxiliares no fluxo de trabalho da instituição. Naturalmente é observável que sistemas como esse tem rápido crescimento, devido as necessidades impostas pelo mercado e a oferta de vagas. Desta maneira, é possível dizer que o SGA é indispensável para qualquer instituição de ensino (5).

O Moodle está em constante adaptação às novas tecnologias aderidas ao contexto de ensino, afim de proporcionar aos estudantes novos recursos para auxílio a aprendizagem, novas funcionalidades foram disponibilizadas através do lançamento de novas versões. Por exemplo calendário mais completo, painéis e notificações personalizadas, adesão de recursos de vídeo, imagem e áudio, como formas de resposta a questionamentos, suporte a conteúdo interativo como HP5 (7).

O Moodle é o sistema com grande demanda de integração, pois os dados primeiro são cadastrados no sistema de gerenciamento da instituição, após isso é preciso enviá-los ao Moodle, à criar no ambiente virtual os cursos e disciplinas, vinculando alunos e professores com elas entre outros serviços.

Portanto, uma correta integração entre SGA e Moodle é necessária porque traz uma série de benefícios de gestão, gerenciamento e aprendizagem, que pode melhorar a qualidade do ensino e serviços prestados pela instituição. Considerando que há disponível no mercado um conjunto diverso de SGAs, orientados para distintas IES (públicas e privadas), incluindo algumas com desenvolvimento pela própria IES, um modelo específico de integração com o Moodle é impraticável. Assim, torna-se um serviço de valor a apresentação de um modelo conceitual de integração para que as IES utilizem como referência para as suas necessidades.

3 OBJETIVOS

3.1 OBJETIVO GERAL

- Descrever uma aplicação modelo de integrador de sistema de gestão acadêmica genérico e Moodle.

3.2 OBJETIVOS ESPECÍFICOS

- Caracterizar técnicas e abordagens de integração entre sistemas;
- Identificar os processos e serviços para integração entre qualquer SGA e Moodle;
- Desenvolver modelo conceitual de integrador.

4 REFERENCIAL TEÓRICO

4.1 INTEGRAÇÃO

A integração pode ser basicamente definida como a sincronização entre sistemas, reunindo duas ou mais soluções em apenas uma, garantindo que a informação seja registrada em ambos sistemas, afim de permanecer disponível para consulta em qualquer um dos sistemas integrados, comumente integrando três tipos de informações, serviços, dados e processos (9), existindo quatro métodos principais de integração, banco a banco, troca de dados eletrônica, *Application Programming Interface* (API) e ferramenta intermediadora (11).

Com o crescimento do volume de dados trafegando pelos sistemas, implantar corretamente a integração fornece mais velocidade e confiabilidade nos processos da organização, inclusive, este volume de dados é melhor utilizado quando integrado, pois os dados gerados por um sistema podem ser aproveitados por outro, potencializando o uso da inteligência de negócio e evitando o retrabalho de inserir dados repetidos em vários sistemas (11).

Por outro lado, a limitação de integração, ou falta dela, gera vários problemas operacionais na instituição, tornando serviços ineficientes ou redundantes (4), também é observado por (5) alguns problemas gerados pela falta de integração:

- Perda de Competitividade;
- Elevação dos custos e aumento dos riscos de exposição a erros;
- Capacidade limitada de inovação e pró-atividade da empresa;
- Dificuldade na evolução e no aprimoramento dos processos de negócios.

Os problemas decorrentes da limitação de integração entre sistemas demonstram a necessidade da utilização de ambientes integrados, afim de melhorar a qualidade dos serviços prestados, inclusive segundo (5) a integração é essencial para uma boa gerência de negócios. O mesmo autor cita alguns benefícios de ambientes integrados:

- Redução dos gastos operacionais evitando que o mesmo trabalho seja feito duas ou mais vezes, o que favorece a automação de atividades;
- Crescimento na capacidade da instituição em adaptar seus processos;
- Torna viável a expansão da instituição de forma consistente;
- Viabiliza novos modelos de negócios;
- Maior agilidade e capacidade em atender as regulamentações do mercado.

A integração proporciona algumas vantagens quando implantada, como ganho em desempenho ao automatizar e integrar processos, o que é essencial para conseguir atender a toda demanda. Diminui os custos pois elimina deficiências operacionais e, processos automatizados

são mais rápidos e eficientes, ganhando em tempo, gerando economia. O retrabalho é reduzido pois as atividades e comunicações ficam registradas num sistema amplo e eficiente, facilitando a consulta às informações armazenadas. Por último, a integração melhora a comunicação com os clientes, fornecendo as informações com mais acessibilidade e rapidez (11).

4.2 SISTEMA DE GESTÃO ACADÊMICA

Sistema de gestão acadêmica (SGA) é uma ferramenta utilizada para gerenciamento dos processos correspondentes a gestão acadêmica, presente em instituições de ensino superior (5).

No SGA estão contidas todas as informações pertencentes a vida acadêmica do estudante e do professor (5), com isso ele é capaz de consolidar as informações relevantes para gestão da instituição, e isso por sua vez, é fundamental para o gerenciamento de demandas cada vez maiores, face ao crescimento da oferta de vagas (4).

A transparência das informações disponíveis no SGA é importante para gestão e controle mais eficiente do processo administrativo, possibilitando planejar as ações de maneiras mais simples e competentes, diminuindo os custos ao otimizar o uso dos recursos (10).

Mas para (8) por conta de várias razões de ordem econômica e visão estratégica das instituições, os sistemas de gestão acadêmica foram crescendo muitas vezes de maneira independente, sem muitas preocupações para integração entre sistemas.

4.3 AMBIENTE VIRTUAL DE APRENDIZAGEM

Ambientes Virtuais de Aprendizagem (AVA) ou em inglês conhecido como *Learning Management System* (LMS) é um *software* que utiliza de princípios pedagógicos e uma filosofia social construtiva para ensino através de ferramentas online (5).

O Moodle (*Modular Object-Oriented Dynamic Learning Environment*) é um *software* desenvolvido em *open source* de apoio a aprendizagem, implementando um sistema de gerenciamento de cursos - *Course Management System* (CMS) (5). Ele implementa uma plataforma de administração de atividades educacionais projetada para criação de ambientes virtuais online voltados para a aprendizagem colaborativa, possibilitando um sistema aos educadores, administradores e alunos com qualidades como robustez, segurança e integração (6).

Como mencionado por (8) a plataforma Moodle foi projetada para ser flexível, versátil e amigável, contendo uma série de recursos que estão em constante atualização, permitindo aos professores compartilhamento de documentos, atribuição de tarefas aos alunos, criação de questionários *online*, que inclusive possam ser respondidos *offline*, fóruns para discussão, envio de vídeos e áudios, e integração com o aplicativo Moodle para dispositivos móveis, chamado de Moodle *Mobile* (6).

A plataforma do Moodle, segundo (8), é caracterizada por uma arquitetura desenvolvida na organização em *plugins* e módulos, conforme expressa a figura 1, que possibilita a criação

de novos módulos de maneira independente, e uma vez agregados ao sistema principal, ficam disponíveis para utilização de acordo com a preferência dos usuários.

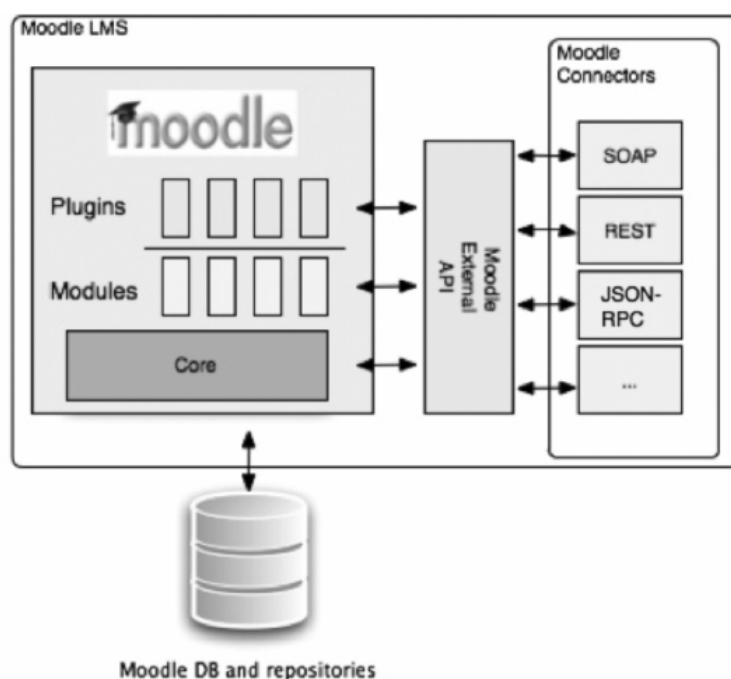


Figura 1 – Arquitetura do Moodle

Fonte: (9)

Os *plugins* do Moodle podem ser definidos como um termo genérico que engloba as funcionalidades ou add-ons que podem ser adicionadas ou instalados no sistema, capazes de alterar a aparência do Moodle e criar novos recursos como módulos, questionários, blocos entre outros. Enquanto o módulo é definido por (8) como um subsistema, ou um serviço responsável pelo funcionamento de algum recurso oferecido pelo ambiente virtual, com propósito de adicionar uma nova atividade, a exemplo o fórum ou tarefas do curso. E blocos são estruturas programáveis do Moodle, posicionadas na direita ou esquerda da página inicial do curso de cada usuário, sendo responsáveis pelo agrupamento dos caminhos até os módulos do curso e apresentar as informações aos usuários, como por exemplo o calendário.

Também pode ser observado pela Figura 1 que o Moodle faz o uso de API para estabelecer a comunicação entre os *plugins* e módulos com recursos externos, viabilizados através do uso de protocolos de comunicação como SOAP, REST e JSON-RPC.

O protocolo REST é parte integrante da arquitetura web service que opera como um canal de comunicação entre diferentes computadores ou sistemas via internet. SOAP é um protocolo padrão que utiliza HTTP e XML para comunicação entre diferentes sistemas operacionais como Windows e Linux. E JSON, que significa *JavaScript Object Notation*, é um formato de troca de dados leve e fácil de analisar que independe de linguagem. Os arquivos JSON consistem em coleções de pares de nome, valor e listas ordenadas de valores, que são estruturas de dados usadas pela maioria das linguagens de programação.

O Moodle também possui um aplicativo para dispositivos móveis, que busca apresentar ao acadêmico uma versão *mobile* do site do Moodle, com vários recursos já implementados, trazendo ao acadêmico a praticidade de acesso às funcionalidades do Moodle via dispositivos móveis.

4.4 MÉTODOS DE INTEGRAÇÃO

4.4.1 Banco a banco

Método de integração para sistemas que partilham do mesmo banco de dados para troca de informações. Para isso, são usadas ferramentas de integração, tais como *Extract Transform Load* (ETL) em português Extração Transformação Carregamento, que farão a extração dos dados de um sistema para carregá-los no outro sistema (1).

Esta integração também pode ser realizada através de *scripts* de banco de dados, onde uma consulta efetuada no primeiro sistema gera um conjunto específico de dados, e em seguida, esta informação é inserida diretamente no banco de dados do outro sistema, podendo ser realizada através do uso *triggers*, ao ponto que uma *trigger* é um gatilho que caso acionada dispara uma sequência de comandos de banco de dados e disponibiliza a informação para inserção no outro banco, criando assim os componentes e atribuições no sistema integrado. Entretanto, um problema do método de integração banco a banco é a falta de segurança, pois permite o acesso de outra aplicação a um banco de dados alheio.

4.4.2 Troca de dados eletrônica

O funcionamento deste método de integração depende da exportação e importação de arquivos, onde um sistema exporta e outro importa. Pode ser que esse processo de integração não seja automático, porém sempre haverá maneiras de controle sobre os dados que estão sendo importados, facilitando o controle dos dados (1).

4.4.3 *Application Programming Interface* (API)

Application Programming Interface em português pode ser traduzida por Interface de Programação de Aplicativos. Tal método usa de comunicação em tempo real com ambos sistemas, para lançar as informações alteradas neles, mantendo ambos sistemas atualizados. Este método de integração entre sistemas possui um bom desempenho na comunicação e maior velocidade na troca de informações, inclusive não é necessário a utilização de sistemas intermediadores. (1)

A literatura pesquisada dispõe de trabalhos propondo modelos de integração utilizando API para desenvolvimento, tais modelos foram projetados com objetivo de integrar SGA e

AVA, como (8) e (3). Elaborados de maneira genérica, permitem adaptações para adequação às especificidades e características dos diferentes sistemas de gestão acadêmica.

4.4.4 Ferramenta intermediadora

Este método de integração é baseado no desenvolvimento de uma nova ferramenta integradora para comunicação com ambos sistemas. Este método pode produzir um bom desempenho, visto que otimiza processos, melhorando rotinas e procedimentos acadêmicos, automatizando a integração entre os sistemas. Entretanto, um ponto negativo citado por (1) é o custo de ter um terceiro sistema, além do AVA e SGA, para possibilitar a integração.

4.5 ARQUITETURA ORIENTADA A SERVIÇOS

Do inglês *Service Oriented Architecture* (SOA), trata-se de uma arquitetura de *software* cuja política de desenvolvimento das funcionalidades, ou módulos implementados pelas aplicações, sejam disponibilizadas em forma de serviços, permitindo a reutilização de código por outras aplicações, com objetivo de aprimorar a eficiência, agilidade e produtividade (5).

É muito comum ver em outras arquiteturas de desenvolvimento de *software* a organização dos processos ser tratada separadamente, dividida entre os departamentos, limitando as tarefas pertencentes somente aquele setor. Entretanto em SOA o objetivo é justamente o contrário, é buscado integrar as aplicações, o que proporciona maior flexibilidade para mudanças (2).

Em SOA o serviço é um artifício abstrato que descreve a capacidade de efetuar tarefas com uma determinada funcionalidade (8), composta por um serviço provedor e outro consumidor, o serviço consumidor faz uma requisição ao provedor, que responde com o resultado ao consumidor; o provedor de serviços também pode ser o consumidor (4). Cada serviço pode ser utilizado por outro serviço, baseado no conceito de que os serviços estão cientes da existência um do outro (5).

O relacionamento entre os serviços é apresentado pela Figura 2; o serviço A tem conhecimento da existência do serviço B porque faz a leitura da descrição do serviço, e esta no mínimo irá informar o nome e a localização do serviço. O que representa o relacionamento entre os serviços, ao ponto que o consumidor solicita o serviço que tomou conhecimento e aguarda pelo resultado esperado.

O autor (5) também enfoca na necessidade dos serviços trocarem mensagens para produção de resultados, mas mantendo a estrutura flexível de comunicação entre os serviços proposta por SOA.

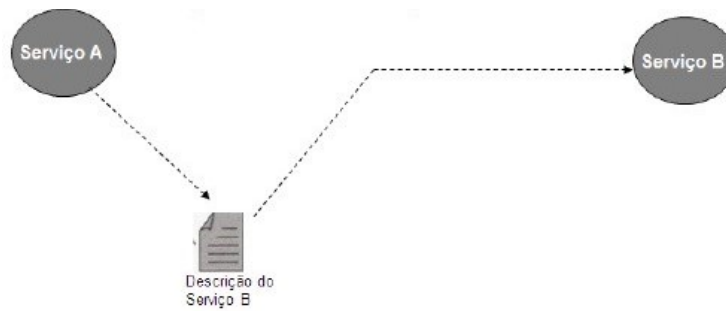


Figura 2 – Como os serviços se relacionam

Fonte: (5)

4.6 WEB SERVICES

Web Services são uma parte integrante de SOA e estão cada vez mais populares, como menciona (4) "estão se popularizando a cada dia por possibilitar que aplicações distintas, possam comunicar-se, independente da linguagem de programação utilizada", ele também destaca o reaproveitamento de código entre os módulos implementados, não importando a linguagem de programação. Por (8) temos que dos aspectos mais importantes na proposta de arquitetura é "separação da implementação do serviço da sua interface".

Web services possuem uma arquitetura independente de plataformas de programação, definidos por (8) como um *software* independente e auto-descritivo disponível na rede, encarregado de executar tarefas e resolver problemas conforme o interesse de quem o requisita.

Desta forma, é possível que o utilizador requisiute o serviço sem estar preocupado com aspectos relacionados ao funcionamento interno do serviço, nem como este irá obter o resultado. O utilizador apenas sabe que deve chamar o serviço e aguardar pelo retorno com o resultado.

A estrutura do *web service* é composta de entradas e saídas de dados descritas através de *Web Service Definition Language* (WSDL), no qual, seu acesso é feito através de protocolos como *Representational State Transfer* (REST) e *Simple Object Access Protocol* (SOAP) que utiliza a estrutura XML para organização das informações e troca de dados (5). O registro do serviço também pode ser feito através do uso do protocolo *Universal Description Discovery and Integration* (UDDI) com infra-estrutura para fixar definições de descrição (*describing*), descoberta (*discovering*) e integrações de serviços (8).

Conforme descrito na Figura 3 o relacionamento da estrutura do *web service*, segundo (8) pode ser organizado em quatro fases;

- I. A descrição do serviço é enviada pelo fornecedor ao serviço informativo, para que este faça a publicação do serviço.
- II. O cliente em conjunto com o serviço informativo procura identificar o melhor serviço para a tarefa requisitada.

- III. O serviço informativo faz o envio do serviço identificado.
- IV. O cliente negocia a utilização do serviço com o fornecedor de serviço

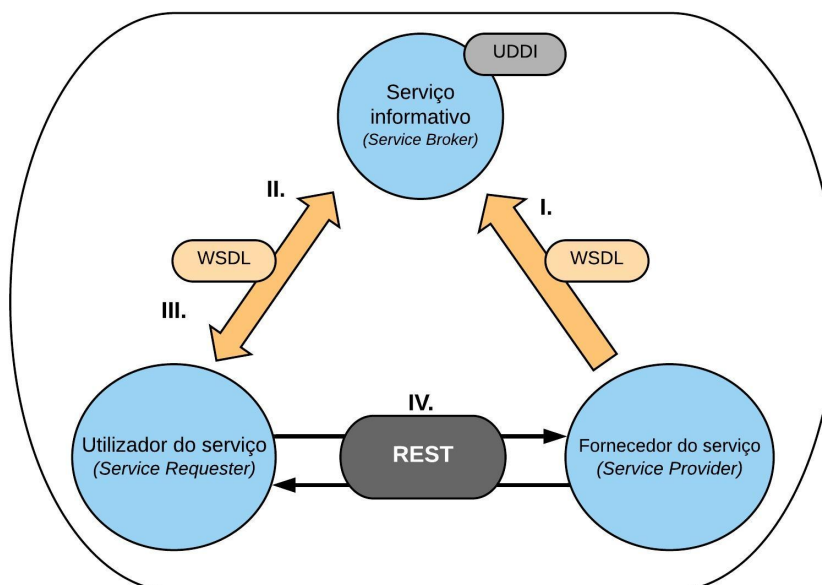


Figura 3 – Fases da utilização de *web services*

Fonte: (8)

Também mencionado por (8), há vários benefícios na utilização de *web services*, dentre eles modularidade, interoperabilidade, extensibilidade, baixos custos de desenvolvimento e reutilização de código.

4.7 TRABALHOS RELACIONADOS

Proposto por (8) um modelo de integração com base em *Web Services* entre SGA e AVA Moodle. Projetado para toda e qualquer instituição de ensino superior, considerando os dados existentes no SGA sobrepostos aos dados no Moodle, por conter dados fidedignos e utilizados como dados oficiais para emissão de certificados e documentos.

O modelo proposto foi implementado utilizando arquitetura orientada a serviços, composto por uma arquitetura de *software* com o princípio de desenvolvimento de aplicações cuja as funcionalidades são implementadas em forma de serviços (8). Por definição os autores optaram por descrever como "um conjunto de serviços que comunicam entre si", definição esta mais simples porém menos confusa.

Como definido pelos autores, na Figura 4 é descrito a configuração do modelo proposto, contendo quatro componentes diferentes.

- Moodle, LMS ou AVA.

- SGA - Sistema de gestão acadêmica.
- Moodle-DFWS - Infra-estrutura de *Web Services do Moodle*.
- WS-SGA - *Web Services* de acesso ao SGA.

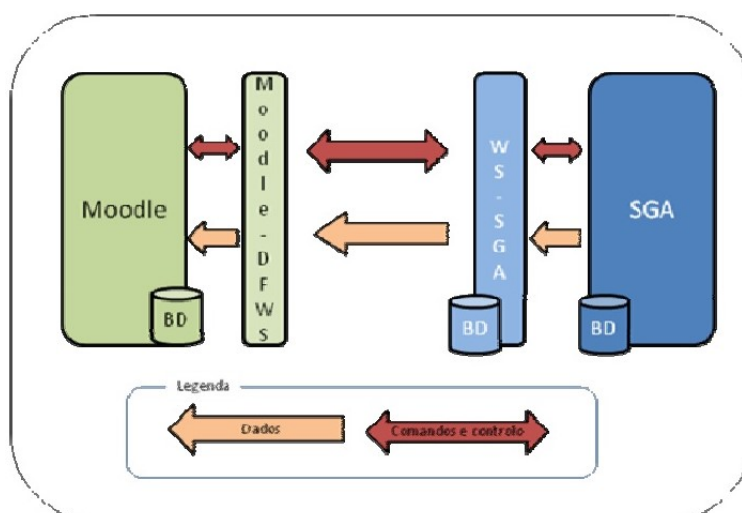


Figura 4 – Modelo de integração Moodle-SGA

Fonte: (8)

Através da Figura 4 podemos observar que o fluxo das informações é emitido do SGA para o Moodle, ou seja, o contrário não acontece, nunca será passado do Moodle para o SGA. Por outro lado, o fluxo dos comandos e controles se dá em todas as direções, tanto o SGA via WS-SGA recebe e envia informações ao Moodle-DFWS como a forma contrária. E também o modelo proposto restringe a comunicação apenas com os respectivos *web service*, não existindo comunicação direta entre SGA e AVA Moodle.

A criação de usuários, estudantes, professores, disciplinas e a associação de estudantes e professores às matérias, é feita de maneira rápida e segura, isso graças ao uso de *web service*, como proposto pelo modelo. Também será possível aumentar a gama de serviços, aumentando as informações compartilhadas na integração.

Também neste âmbito foi projetado um modelo de integração por (4), cuja base é arquitetura orientada a serviços e o uso de *Enterprise Application Integration* (EAI) - Integração de Aplicações Corporativas - que apoia e dá suporte a implementação de ferramentas voltadas para a integração.

O modelo proposto também é adaptável para integração entre qualquer AVA e SGA, usando arquitetura orientada a serviços, mas com acréscimo do desenvolvimento de um *framework* baseado em *web service*, este responsável por fazer acesso direto ao banco de dados, tanto do SGA como do AVA.

Para que o modelo proposto seja adaptável a qualquer SGA e AVA, foi realizado um estudo que identificou requisitos necessários para definir um padrão adaptável, com isso foi adicionado



Figura 5 – Modelo de integração

Fonte: (4)

um *software* chamado de Gerenciador, como visto na Figura 5. *Software* desenvolvido para fazer a gestão das demandas do usuário, controle de todas as funções e recursos da integração, encarregado de configurar os parâmetros necessários para funcionamento do *web service*, como por exemplo, tipo do banco de dados, endereços IP de localização do banco, usuário e senhas de acesso. Ou seja, o usuário faz as configurações da integração através do Gerenciador (4).

Através da Figura 5 ainda é possível observar que o *web service* é responsável de fazer a comunicação e integração entre o SGA e AVA, independente da linguagem de programação dos sistemas, visto que *web service* faz a solicitação de serviços direto ao banco de dados dos sistemas, através de requisições em *Structured Query Language* (SQL). A comunicação com o gerenciador também é feita através de *web service* com o uso do protocolo de comunicação chamado SOAP (4).

A proposta de desenvolvimento do modelo conceitual de integrador entre Moodle e sistema de gestão acadêmico genérico faz uso das abordagens pesquisadas pelo referencial teórico, utilizando para desenvolvimento as técnicas e ferramentas da arquitetura orientada a serviços com uso de *web services*, trazendo luz ao procedimento necessário para desenvolvimento do integrador dentro do sistema Moodle, usufruindo do conjunto de APIs e demais serviços dele.

Para desenvolvimento do modelo conceitual foi proposto a utilização da arquitetura de *software web services* para implementação, combinado com o uso das APIs disponibilizadas pelo Moodle. Uma vez que os trabalhos relacionados também fazem a utilização da arquitetura, tendo em vista a obtenção dos benefícios citados por (8); descritos na seção 4.

5 DESENVOLVIMENTO

Modelo conceitual de integração foi desenvolvido utilizando da arquitetura de Web services, essa já disponibilizada pelo Moodle. Com web service é possível criar uma integração mais próxima de genérica por parte do sistema que integra com Moodle, pois as inserções serão efetuadas por serviços implementados pelo próprio Moodle e o sistema de gestão acadêmica deverá fornecer as informações solicitadas para cada serviço. Dessa maneira, é necessário configurar o SGA com o integrador, e este ficará ouvindo o SGA, pronto para executar as tarefas assim que for acionado pelo sistema, replicando as informações no Moodle, criando a integração.

O Moodle dispõe de uma lista de serviços implementados a executar tarefas únicas, filtrados conforme a necessidade do utilizador, e o papel do integrador proposto é intermediar essa relação entre os web services do Moodle com o SGA genérico.

Sendo assim, o funcionamento do integrador pode ser descrito da seguinte forma; o integrador têm uma série de serviços implementados disponibilizados para uso do SGA, com a descrição dos parâmetros e dados que são necessários para execução de cada serviço, uma vez em posse dos dados o integrador executará a tarefa atualizando os dados no Moodle, seja uma operação de inserção, atualização ou exclusão, retornando ao SGA com a conclusão da execução do serviço, se um retorno positivo ou a informação com o erro para que possa ser tratado.

O modelo proposto foi projetado para criar uma integração capaz de suprir a maior parte da demanda de integração entre SGA da instituição e Moodle, que segundo (9) usualmente são usuário, curso, disciplina, matrícula e nota.

A versão do Moodle escolhida para desenvolvimento do modelo foi a 3.10, sendo essa a versão estável mais nova na data de elaboração do trabalho, e ao instalar e configurar um ambiente Moodle foi possível testar e definir a lista de serviços necessários a estabelecer a integração entre os sistemas.

Neste ambiente Moodle para testes foi criado um web service através da aplicação do Moodle, e adicionado a relação das APIs responsáveis pela execução das tarefas, a cumprir a integração de criação de usuário, cursos e disciplinas, o relacionamento entre usuários e disciplinas, o vínculo das notas e matrículas.

O Moodle permite a criação de um ou vários web services diferentes, ficando a critério da instituição a definição de quantos. Por exemplo, pode ser criado um web service para cada tarefa a ser cumprida, seja criação de usuário, consulta de usuário pelo id e outras. Cada serviço também deve receber as permissões sistêmicas que permitem a execução das tarefas. Por fim deve ser criado um *token* para o serviço, pois é assim que o Moodle organiza os serviços classificando-os de maneira individual; cada serviço recebe um *token* e deve ser chamado utilizando esse *token*.

Foram desenvolvidos testes com a criação de tarefas a fazer uso do web service, que executadas chamam o serviço, observando as configurações necessárias, como a seleção do *token* e parâmetros da função, preenchendo as variáveis conforme o uso, obrigatório ou opcional.

Como exemplo pode ser mencionado o serviço responsável pela criação de usuários, que após criado o serviço, permissões e *token*, é possível fazer a chamada da API *core_user_create_users* do Moodle. A entrada dessa API é um vetor de vetores composta pelo(s) usuário(s) que deseja cadastrar, com os seguintes parâmetros obrigatórios: a identificação do usuário, que é equivalente ao *username*, nome, sobrenome e um e-mail válido e único. Alguns parâmetros tem a obrigatoriedade passível de configuração, ficando a critério da instituição a sua definição, é o caso da senha; é possível criar o usuário via chamada de API obrigando a definição da senha como criá-lo sem senha. Conforme código php abaixo:

```

1<?php
2    //TOKEN
3    $token = '4cd919e408f66c94b69bf0f17e358913';
4    $domainname = 'http://localhost/';
5    // FUNCTION CALLED
6    $functionname = 'core_user_create_users';
7    $restformat = 'json';
8
9    //USERS
10   $user1 = new stdClass();
11   $user1->username = 'userone';
12   $user1->password = '@User1Password1';
13   $user1->firstname = 'User0nefirstname';
14   $user1->lastname = 'User0nelastname';
15   $user1->email = 'user1mail@moodle.com';
16   $user2 = new stdClass();
17   $user2->username = 'usertwo';
18   $user2->password = '@User2Password1';
19   $user2->firstname = 'UserTwofirstname';
20   $user2->lastname = 'UserTwolastname';
21   $user2->email = 'user2mail@moodle.com';
22   $users = array($user1, $user2);
23   $params = array('users' => $users);
24
25   //REST CALL
26   header('Content-Type: text/plain');
27   $serverurl = $domainname . '/webservice/rest/server.php'.
28   '?wstoken=' . $token . '&wsfunction='.$functionname;
29   require_once('./curl.php');
30
31   $curl = new curl;
32   $resp = $curl->post($serverurl . $restformat, $params);
33?>

```

O início do código demonstrado é dado pela definição do *token* do serviço que será executado, este deve ser gerado dentro do sistema do Moodle e utilizado conforme a variação da função de utilizada. Após isso é definido o nome do domínio, que em ambiente de testes foi

usado o próprio *localhost* enquanto em produção será um domínio Moodle pertencente a IES. Após isso é definida a função que será utilizada e o formato REST de retorno, neste caso *json*.

A inserção dos usuários foi realizada de maneira singular, afim de testar a usabilidade da API de criação de usuários. No ambiente de produção deverá ser realizado com um laço que fará leitura dos cadastros encontrados no sistema que está a integrar com Moodle. Alimentando a variável *\$users* na linha 22 com os usuários que deverão ser inseridos no Moodle. Enquanto a linha 23 é definida a organização dos parâmetros como requisitado na descrição do serviço; que no caso desta função é um vetor de usuários com o índice nomeado *users*.

Feito isso é possível fazer a chamada REST, alterando o padrão do *header* para *text/plain*, uma vez que o tratamento da variável *\$serverurl* deve ser mantido como o escrito pelo código. Após isso, na linha 27, é montado o padrão para chamada do web service do Moodle, sendo o nome do domínio, o servidor REST, o *token* correspondente ao serviço que está em uso e o nome da função de execução.

Uma vez montada a estrutura com os dados para fazer a inserção, e também o padrão de comunicação com o web service do Moodle, é utilizada a biblioteca *curl* para comunicação com o servidor; a partir da linha 31. *Curl* está disponível na linguagem utilizada *php*, e permite a conexão com o servidor através de diferentes protocolos como *https*, *ftp*, *http post*, *http put* e outros, estabelecendo o envio e recebimento de mensagens com o servidor. E por último na linha 32 é feita a chamada do método *curl->post* passando como parâmetros os dados do servidor, formato REST e os dados a serem inseridos.

Ainda é possível observar no código *Php* que as APIs do Moodle esperam receber os dados sob um nome de índice específico, que varia para cada função, no caso da função *core_user_create_users* o nome do índice que conterá os valores é *users*. Também é importante mencionar o protocolo para comunicação que foi utilizado, neste caso REST com formato *json* de retorno, porém no Moodle ainda é possível usar outras opções, os protocolos SOAP e XML-RPC.

Uma das características do Moodle é a organização do sistema em módulos, que permite o trabalho dos desenvolvedores de maneira separada de todo o sistema, a implementação fica pertinente àquele módulo que está sendo desenvolvido, assim os dados são recebidos, tratados e executados conforme o objetivo do módulo e retornados a quem os chamou.

Essa característica modular também é visível no tangente ao uso e implementação do web service disponível no Moodle. Visto que há uma relação de várias APIs diferentes implementadas, cada uma com propósitos bem definidos e especificações de parâmetros.

O modelo de integração conceitual desenvolvido também segue a característica do Moodle da organização do sistema ser disposta em módulos. Da forma que os serviços criados foram classificados em módulos conforme a necessidade de integração dos dados, módulos que foram chamados de *core_usuario*, *core_categoria*, *core_curso* e *core_nota*. Cujas composições de cada um deles é dada pelo conjunto de APIs pertinentes às características do usuário, categoria, curso e notas respectivamente.

6 RESULTADO

Neste capítulo será apresentado o modelo conceitual desenvolvido através das pesquisas, estudos e testes realizados dentro de um ambiente de simulação, o modelo apresentado é descrito por uma metodologia de implementação que será responsável pela criação do modelo e configuração dele com o Moodle e SGA, utilizando da arquitetura de web service, protocolos REST e/ou SOAP, descrevendo os componentes envolvidos e informando quais definições são obrigatórias e quais ficam a critério da instituição.

O modelo proposto usa das abordagens apresentadas no referencial teórico para elencar as ferramentas e técnicas que deverão ser usadas para implementação do modelo, descrevendo que a comunicação com o SGA será com uso de *triggers*, e demonstrando a criação do modelo de integrador com a metodologia implementada, que faz uso dos recursos e funcionalidades disponíveis no Moodle, responsáveis pelos processos de alterações no ambiente Moodle da IES.

A estrutura da metodologia para criação do integrador é organizada da seguinte forma:

- Criação do serviço através do Moodle;
- Criação do usuário que fará uso do serviço criado;
- Relação dos serviços necessários para integração;
- Definição das capacidades, regras e permissões fundamentais do usuário;
- Criação do Token do serviço;
- Caracterização dos serviços.

A começar a metodologia de integração, considerando que o ambiente Moodle à integrar está instalado, é necessário dar início às configurações de habilitação do Web Service e APIs, essas configurações são realizadas via aplicação Moodle. Entrando com o usuário administrador deve ser habilitado o uso de web service e o protocolo REST, tais configurações são encontradas na opção de administração do site demonstrado na Figura 6. Fica opcional a escolha do protocolo para uso, pode ser optado por SOAP, REST e XML-RPC, na simulação foi utilizado REST.

Com isso é possível dar início a criação dos serviços, estes segundo o modelo projetado foram estruturados em forma de módulos, organizados conforme as funções de trabalho com os elementos envolvidos para integração, nomeando os módulos de *core_usuario*, *core_categoria*, *core_curso* e *core_nota*. A criação é feita na opção de serviços externos na categoria de web services da seção *plugins*, e deve ser marcada a opção ativo; Figura 7.

Após criados os módulos devem ser adicionadas as funções que farão parte deles, visto que cada módulo criado terá uma relação de APIs (serviços) que serão responsáveis pela integração. A adição é realizada de maneira individual em cada módulo na opção "funções" na lista de serviços externos, conforme demonstrado pela Figura 8. As funções que integram cada módulo estão descritas abaixo:

My Local Moodle

[Painel](#) / [Administração do site](#) / [Plugins](#) / [Web services](#)

Categoria: Administração / Plugins / Web services

[Resumo](#)

Permitir um sistema externo para controlar o Moodle

Os passos seguintes ajudam você a configurar os serviços web do Moodle para permitir que um sistema externo interaja com Moodle. Isso inclui a criação de um método de autenticação do token (chave de segurança).

Etapas	Status	Descrição
1. Habilitar web services	Não	Web services devem ser habilitados em Características avançadas.
2. Ativar protocolos	Nenhum	Ao menos um protocolo deve estar habilitado. Por razões de segurança, apenas protocolos utilizados devem estar habilitados.

Figura 6 – Habilitar web service e protocolos

My Local Moodle

[Painel](#) / [Administração do site](#) / [Plugins](#) / [Web services](#) / [Serviços externos](#) / [Serviço externo](#)

▼ **Serviço externo**

Nome !

Nome breve

☒ Ativado

☐ Apenas usuários autorizados ?

[Mostrar mais ...](#)

Este formulário contém campos obrigatórios marcados com !.

Figura 7 – Criação dos serviços

- **core_usuario:** core_user_get_users, core_user_create_users, core_user_update_users, core_user_delete_users, enrol_manual_enrol_users.
- **core_categoria:** core_course_create_categories, core_course_get_categories, core_course_delete_categories, core_course_update_categories.
- **core_curso:** core_course_create_courses, core_course_get_courses, core_course_delete_courses, core_course_update_courses.

- **core_nota:** core_notes_create_notes, core_notes_delete_notes, core_notes_get_course_notes, core_notes_get_notes, core_notes_update_notes.

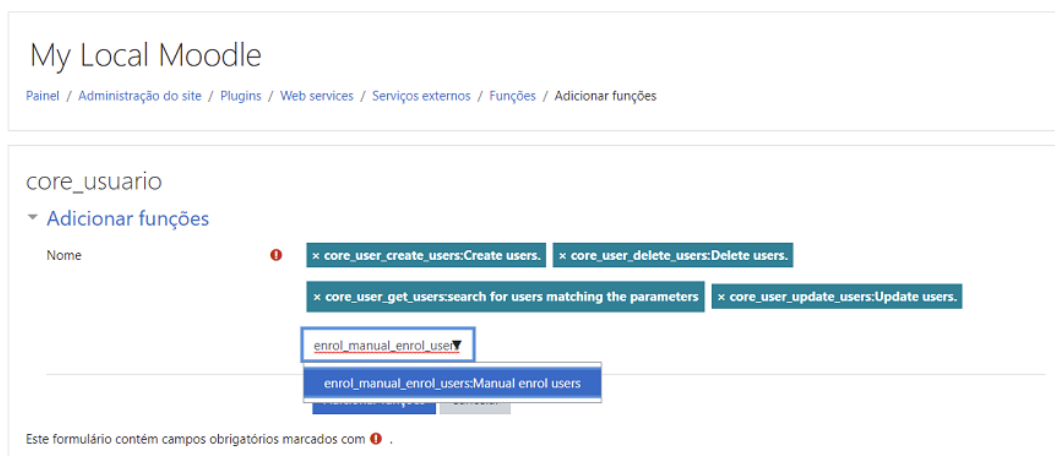


Figura 8 – Adicionar funções

Adicionadas as funções em cada módulo, o próximo passo é criar os usuários que farão uso destes módulos, essa opção também é encontrada dentro do painel de administração do site na seção usuários. É preciso criar quatro usuários, um para cada módulo, assim está caracterizada a criação dos quatro módulos web services no Moodle. Após isso é preciso criar uma nova regra de sistema com intuito de liberar o uso de web service para os usuários criados.

A regra é criada nas opções de usuário em definir papéis, criando uma nova regra que deverá ser marcada como "sistema", pois fará uso das chamadas sistêmicas do Moodle, demonstrado na Figura 9. Na mesma página deverá ser atribuída todas as capacidades requeridas para fazer uso das APIs relacionadas; a lista de capacidades estão descritas abaixo, classificadas por módulo:

- **core_usuario:** moodle/user:create, moodle/user:delete, moodle/user:viewdetails, moodle/user:viewhiddendetails, moodle/course:useremail, moodle/user:update, enrol/manual:enrol, moodle/role:assign;
- **core_categoria:** moodle/category:manage, moodle/category:viewhiddencategories;
- **core_curso:** moodle/course:create, moodle/course:visibility, moodle/course:delete, moodle/course:view, moodle/course:update, moodle/course:viewhiddencourses, moodle/course:changecategory, moodle/course:changefullname, moodle/course:changeshortname, moodle/course:changeidnumber, moodle/course:changesummary;
- **core_nota:** moodle/notes:manage, moodle/notes:view.

My Local Moodle

[Painel](#) / [Administração do site](#) / [Usuários](#) / [Permissões](#) / [Definir papéis](#)

Gerenciar papéis | Permitir atribuição de papel | Permitir sobreposição de papel | Permitir troca de papéis

Adicionando um novo papel?

[Criar este papel](#) [Cancelar](#)

Nome breve ?

Nome completo personalizado ?

Descrição personalizada ?

Arquétipo do papel ?

Tipos de contexto onde esse papel pode ser atribuído

- ☒ Sistema
- ☐ Usuário
- ☐ Categoria
- ☐ Curso
- ☐ Módulo de atividade
- ☐ Bloco

Figura 9 – Criação novo papel

Para concluir a criação do novo papel ainda é preciso permitir a capacidade de uso do protocolo de comunicação, como neste caso foi utilizado REST a capacidade requerida é *webservice/rest:use*, caso seja usado outro protocolo a capacidade requerida é *webservice/soap:use* para SOAP e *webservice/xmlrpc:use* para XML.

O conjunto de serviços propostos requerem que o novo papel criado tenha a permissão de atribuição de papel as funções Estudante e Professor, pois essas são as atribuições necessárias para correta execução das APIs solicitadas pelo projeto, a prover a integração dos sistemas.

As permissões requeridas por cada função são encontradas no gerenciamento de serviços externos ao buscar os serviços criados, deve-se verificar as permissões toda vez que criar um serviço novo. Uma vez feito isso é preciso, na seção de atribuição de papéis globais, atribuir os usuários criados à regra de sistema criada que fará uso de web service, assim está liberado o uso de web service aos usuários.

Para finalizar as configurações a conceder uso do web service por parte do Moodle, deve-se criar os tokens responsáveis por fazer a chamada do serviço, a criação destes é feita na opção de gerenciamento de tokens na categoria web services, deve ser criado um token para cada usuário, e fazer a associação do usuário com o módulo que este fará uso Figura 10.

My Local Moodle

[Painel](#) / [Administração do site](#) / [Plugins](#) / [Web services](#) / [Gerenciar tokens](#)

Gerenciar tokens

Qualquer token pode ser excluído, embora você só possa ver os tokens que você criou.

Token	Nome / Sobrenome	Serviço	Restrição de IP	Válido até	Criador	Operação
251e9590fd8039f629fe2861b9410092	user categoria	core_categoria			Administrador Usuário	Excluir
2a5bc38f7475071125a5fa4e4bd9f270	user curso	core_curso			Administrador Usuário	Excluir
6c1b31b2b6fb93f8c3df19888d499394	user nota	core_nota			Administrador Usuário	Excluir
2444a7ef12f6a341d5818eed0d8e9323	user usuario	core_usuario			Administrador Usuário	Excluir

[Adicionar](#)

Figura 10 – Criação dos tokens

Neste momento também será possível observar se faltou alguma capacidade requerida do serviço criado e associado ao token, caso positivo essa capacidade deverá ser adicionada ao novo papel criado, do contrário a integração não será efetuada com sucesso.

Essa é a metodologia obrigatória para criação e configuração do integrador junto a aplicação Moodle, ficando a critério da instituição a definição do número de módulos, usuários e protocolos que serão utilizados. Mas ainda é preciso caracterizar as funções necessárias para integração, assim abaixo estão descritas as APIs pertinentes a cada módulo e também suas características e parâmetros para execução e funcionamento.

A começar pelo **core_usuario** que é responsável por conter as funções para trabalhar com os usuários, como criação e delete, segue as APIs necessárias:

- **core_user_get_users:**
 - Objetivo: Retornar o usuário buscado pelo id.
 - Formato de entrada: Um vetor de vetores com índice nomeado de "criteria".
 - Parâmetros obrigatórios: São dois parâmetros, o primeiro o critério de busca do id; e-mail, id Moodle, id opcional, username, nome ou sobrenome. O segundo parâmetro é valor do id que será buscado.
 - Demais informações: Não há.
- **core_user_create_users:**
 - Objetivo: Criar usuários no ambiente Moodle.
 - Formato de entrada: vetor de vetores de usuários à criar, com índice nomeado de "users".

- Parâmetros obrigatórios: Por padrão são quatro parâmetros; *username*, primeiro nome, último nome e e-mail.
- Demais informações: A configuração para obrigatoriedade da definição de senha no momento da criação fica a critério da instituição. Ainda a função tem como padrão definir a autenticação como manual e não preencher o *idnumber*, que é utilizado como um id arbitrário vindo da instituição, podendo representar o número da matrícula. Demais informações do usuário são opcionais.
- `core_user_update_users`:
 - Objetivo: Atualizar os dados do usuário no Moodle.
 - Formato de entrada: Vetor de vetores de usuários com nome do índice "users".
 - Parâmetros obrigatórios: Id do(s) usuário(s) que deseja atualizar.
 - Demais informações: Além do parâmetro obrigatório deve ser preenchido somente as informações que deseja alterar.
- `core_user_delete_users`:
 - Objetivo: Deletar o cadastro do usuário.
 - Formato de entrada: Vetor de id dos usuários com nome do índice "userids".
 - Parâmetros obrigatórios: Relação de ids para deletar.
 - Demais informações: Não há.
- `enrol_manual_enrol_users`:
 - Objetivo: Definir papel do usuário e relacionar usuário com o curso.
 - Formato de entrada: Vetor de vetores com nome do índice "enrolments".
 - Parâmetros obrigatórios: Papel do usuário; gerente, criador de curso, professor, moderador, estudante, visitante, usuário autenticado ou usuário autenticado na página inicial. Id do usuário e id do curso.
 - Demais informações: O papel do usuário; gerente, criador de curso, professor, moderador, estudante, visitante, usuário autenticado e usuário autenticado na página inicial são definidos pelos números 1, 2, 3, 4, 5, 6, 7 e 8 respectivamente.

O Moodle organiza os cursos e semestres através de categorias e subcategorias, por isso a importância do **core_categoria** com as funções abaixo:

- `core_course_create_categories`:
 - Objetivo: Criar categorias no Moodle. Pode ser usada para estruturar os semestres, cursos da graduação, pós-graduação entre outras.

- Formato de entrada: Vetor de vetores com nome do índice "categories".
- Parâmetros obrigatórios: Apenas o nome da categoria.
- Demais informações: Como padrão a categoria é criada como uma categoria raiz, para definir a categoria pai que esta deverá ser incluída deve-se informar o id da categoria pai na variável "parent".
- `core_course_get_categories`:
 - Objetivo: Retorna as informações das categorias criadas no ambiente Moodle.
 - Formato de entrada: Um vetor vazio ou um vetor de vetores com nome de índice "criteria".
 - Parâmetros obrigatórios: Ao utilizar a opção "criteria" são dois parâmetros obrigatórios; o critério para busca, como id. E o segundo parâmetro o valor do critério. Caso a entrada seja um vetor vazio a função não tem parâmetros.
 - Demais informações: Pode ser definido o retorno das categorias filhas, por padrão todas as sub-categorias são retornadas.
- `core_course_delete_categories`:
 - Objetivo: Deletar a categoria informada.
 - Formato de entrada: Vetor de vetores com nome de índice "categories".
 - Parâmetros obrigatórios: Id da categoria que deve ser excluída.
 - Demais informações: É possível transferir as sub-categorias para um novo pai ao informar o id dessa categoria pai. Também é possível excluir todas as sub-categorias, por padrão as sub-categorias serão transferidas para o categoria pai informada.
- `core_course_update_categories`:
 - Objetivo: Atualizar a categoria informada.
 - Formato de entrada: Vetor de vetores com nome de índice "categories".
 - Parâmetros obrigatórios: Id da categoria a atualizar.
 - Demais informações: Demais valores deverão ser preenchidos conforme a necessidade de atualização. Também é possível definir o formato de descrição; o valor padrão é 1 para HTML, 0 para Moodle, 2 para Plain ou 4 para Markdown.

As disciplinas de cada curso e semestre estão relacionados nas funções pertinentes ao módulo **core_curso**:

- `core_course_create_courses`:
 - Objetivo: Criar curso.

- Formato de entrada: Vetor de vetores com nome de índice "courses".
- Parâmetros obrigatórios: nome completo, nome curto, id da categoria que o curso pertence.
- Demais informações: A função cria o curso com várias informações como padrão, deve-se atentar a todas as funcionalidades para criar o curso com os padrões desejados.
- `core_course_get_courses`:
 - Objetivo: Retorna os cursos disponíveis no Moodle conforme critério de busca.
 - Formato de entrada: Vetor vazio ou vetor de vetores com nome do índice "options".
 - Parâmetros obrigatórios: Caso optar pelo opção "options" deve informar qual o id do curso que deve ser buscado.
 - Demais informações: Ao optar pelo vetor vazio a função retornará todos os cursos disponíveis.
- `core_course_delete_courses`:
 - Objetivo: Deletar curso.
 - Formato de entrada: Vetor com nome de índice "courseids"
 - Parâmetros obrigatórios: Id do curso que deve ser excluído.
 - Demais informações: Não há.
- `core_course_update_courses`:
 - Objetivo: Atualizar o curso informado.
 - Formato de entrada: Vetor de vetores com nome de índice "courses".
 - Parâmetros obrigatórios: Id do curso que será atualizado.
 - Demais informações: Não há.

E por último o módulo para lidar com as notas, o **core_nota**:

- `core_notes_create_notes`
 - Objetivo: Atribuir notas à usuários.
 - Formato de entrada: Vetor de vetores com nome de índice "notes".
 - Parâmetros obrigatórios: Id do usuário que irá receber a nota, estado de publicação; *personal*, *course* ou *site*. Id do curso que a nota será atribuída, texto da mensagem.
 - Demais informações: Por padrão o texto da mensagem está como HTML.
- `core_notes_delete_notes`:

- Objetivo: Deletar nota
- Formato de entrada: Vetor com nome de índice "notes".
- Parâmetros obrigatórios: Id da nota a ser deletada.
- Demais informações: Não há.
- `core_notes_get_course_notes`:
 - Objetivo: Retorna todas as notas em curso e usuário específico.
 - Formato de entrada: Vetor para curso com nome de índice "courseid" e vetor para usuário com nome de índice "userid".
 - Parâmetros obrigatórios: Id do curso e id do usuário.
 - Demais informações: Não há.
- `core_notes_get_notes`:
 - Objetivo: Retorna as notas.
 - Formato de entrada: Vetor para notas com nome de índice "notes".
 - Parâmetros obrigatórios: Id da nota a ser buscada.
 - Demais informações: Não há.
- `core_notes_update_notes`:
 - Objetivo: Atualizar nota
 - Formato de entrada: Vetor de vetores com nome de índice "notes".
 - Parâmetros obrigatórios: Id da nota, estado de publicação, texto da mensagem.
 - Demais informações: Por padrão o texto da mensagem está como HTML.

6.1 MODELO DE INTEGRAÇÃO PROPOSTO

O modelo de integração é proposto para uso com SGA genérico à integrar com Moodle, utilizando as APIs do Moodle para criação da integração, ao ponto que os dados solicitados ao SGA foram caracterizados em cada descrição dos serviços, a saber quais são os dados obrigatórios e opcionais, e a organização deles.

Para comunicação do integrador com o SGA da instituição é proposto uma abordagem com o uso de *triggers*, capturando o momento que a base de dados do SGA for modificada e que tipo alteração foi realizada, como inserção, update e delete, assim a *trigger* implementada pelo lado do SGA conterà as informações e dados a fornecer ao integrador para replicá-los no ambiente Moodle.

Com a implementação de um terceiro sistema intermediário, responsável por estabelecer a integração entre o SGA genérico e Moodle, é válido pontuar a atenção necessária ao tangente

da segurança do conjunto de dados e informações que estarão circulando pelo Modelo de integrador, e uma vez que este será desenvolvido e executado apenas dentro do sistema interno da instituição, a segurança deste será pelo menos igual à segurança praticada pelo SGA.

O funcionamento do modelo de integração é visto pela Figura 11, no qual em centro está o integrador, que primeiramente está a ouvir o SGA genérico, e este conhece a lista de serviços disponíveis com sua descrição de requisitos e funcionalidade, para que quando uma demanda seja criada, esta chame o serviço do integrador, enviando os dados; cuja comunicação é descrita com o uso de *triggers*. Quando o integrador receber os dados caracterizados pela demanda de integração devida, é selecionado o módulo responsável pela execução do serviço e a função pertinente à demanda, os dados serão tratados observando os parâmetros do serviço, próximo passo é via protocolo REST ou SOAP fazer a chamada da API do Moodle. Após a execução é recebido o retorno da API e este deverá ser repassado ao SGA, a saber se a chamada foi efetuada com sucesso ou não.

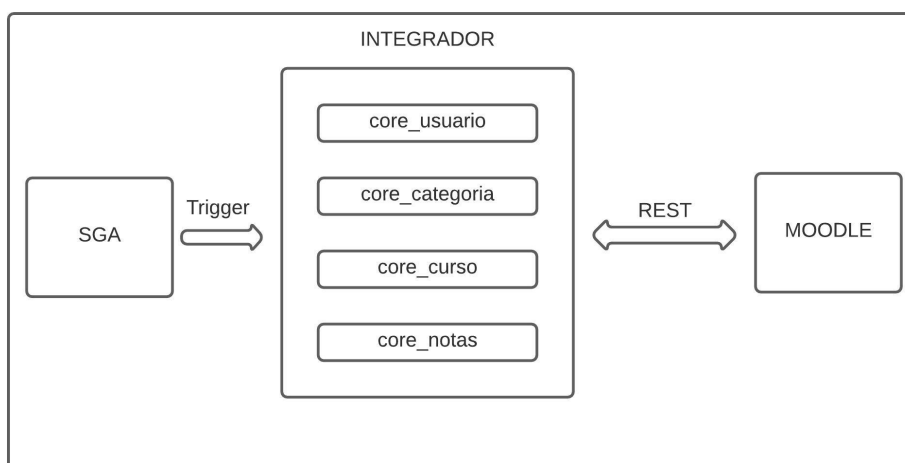


Figura 11 – Modelo de integração proposto

O funcionamento do integrador pode ser visto no exemplo abaixo, executado em um ambiente local para testes, afim de provar o funcionamento do conjunto de serviços, que fazem uso do web service do Moodle, propostos pelo trabalho. É preciso mencionar que em um ambiente real existem diferenças na implementação do integrador, pois informações quanto ao domínio da IES, a maneira de coleta dos dados do SGA, elaboração das *triggers*, organização do código contendo laços para leitura dos dados e de execução junto à API do Moodle.

Para exemplificar foi utilizado o código demonstrado no Capítulo 5 que fez a criação do usuário "userone", com a execução do serviço pertencente ao módulo *core_usuario*, fazendo uso da função *core_user_create_users*. A seguir foi realizada a criação de novos cursos no banco local do Moodle, usando o serviço do módulo *core_curso* e função *core_course_create_courses*, inserindo na categoria padrão do Moodle o novo curso "Curso teste 1" e, por último, foi definido o papel do usuário "userone" como "estudante" então em seguida estabelecido o relacionamento

entre o estudante e o curso criado, isso pela execução da função *enrol_manual_enrol_users* também do módulo *core_usuario*.

Os exemplos dos códigos necessários para cumprimento das tarefas estão descritos a seguir, começando pela implementação do serviço *core_course_create_courses*, responsável pela criação de novos cursos:

create_course.php

```

1<?php
2  //TOKEN
3  $token = '536ca4d8f3b6c4f2d00f455adbd3bdaf';
4  $domainname = 'http://localhost/moodle';
5  //FUNCTION CALLED
6  $functionname = 'core_course_create_courses';
7  $restformat = 'json';
8
9  ///COURSE
10 $course = new stdClass();
11 $course->fullname = "Curso_teste1";
12 $course->shortname = "teste1";
13 $course->categoryid = 1;
14
15 $courses = array( $course);
16 $params = array('courses' => $courses);
17
18 //REST CALL
19 header('Content-Type: text/plain');
20 $serverurl = $domainname . '/webservice/rest/server.php' .
21 '?wstoken=' . $token . '&wsfunction='.$functionname;
22 require_once('./curl.php');
23
24 $curl = new curl;
25 $restformat = ($restformat == 'json')?'&moodlewsrestformat=' .
26 $restformat:'';
27 $resp = $curl->post($serverurl . $restformat, $params);
28?>

```

A começar a codificação é definido o *token* de acordo com o serviço de seleção e o nome do domínio, neste caso o ambiente local *localhost*. Após isso nas linhas 6 e 7 é feita a atribuição da API que será utilizada e o formato *json* para retorno.

A partir da linha 10 a linha 16 são definidos as opções de criação do curso, preenchendo neste caso os itens considerados obrigatórios pela API do Moodle; a classe *stdClass*, nome completo do curso, nome curto e o id da categoria que o curso pertence, esta foi mantida a categoria padrão do Moodle chamada "Miscelânea". Então é estruturado os dados de entrada como ordena os parâmetros da API; um vetor de vetores com nome de índice "*courses*", nas linhas 15 e 16.

A seguir é feita a chamada REST, intervalo da linha 19 a linha 27; foi observado que o padrão de utilização do protocolo para comunicação com o web service do Moodle é o mesmo, como descrito no Capítulo 5, adicionando a verificação da linha 25 cujo propósito é adicionar compatibilidade com outras versões do Moodle.

E por fim utilizando a biblioteca *curl*, disponível na linguagem PHP, para fazer conexão com o servidor e assim realizar a chamada via *\$curl->post* passando os parâmetros como descrito pela API. O retorno da execução da função é recebido na variável *\$resp* para saber se o serviço foi executado corretamente.

A seguir a implementação do serviço de execução da função *enrol_manual_enrol_users*:
enrol_user.php

```

1<?php
2  //TOKEN
3  $token = '4cd919e408f66c94b69bf0f17e358913';
4  $domainname = 'http://localhost/moodle';
5  //FUNCTION CALLED
6  $functionname = 'enrol_manual_enrol_users';
7  $restformat = 'json';
8
9  ///ENROLMENT
10 $enrolment = new stdClass();
11 $enrolment->roleid = 5;
12 $enrolment->userid = 2;
13 $enrolment->courseid = 4;
14 $enrolments = array( $enrolment);
15 $params = array('enrolments' => $enrolments);
16
17 //REST CALL
18 header('Content-Type: text/plain');
19 $serverurl = $domainname . '/webservice/rest/server.php' .
    '?wstoken=' . $token . '&wsfunction=' . $functionname;
20 require_once('./curl.php');
21
22 $curl = new curl;
23 $restformat = ($restformat == 'json')?'&moodlewsrestformat=' .
    $restformat:'';
24 $resp = $curl->post($serverurl . $restformat, $params);
25?>

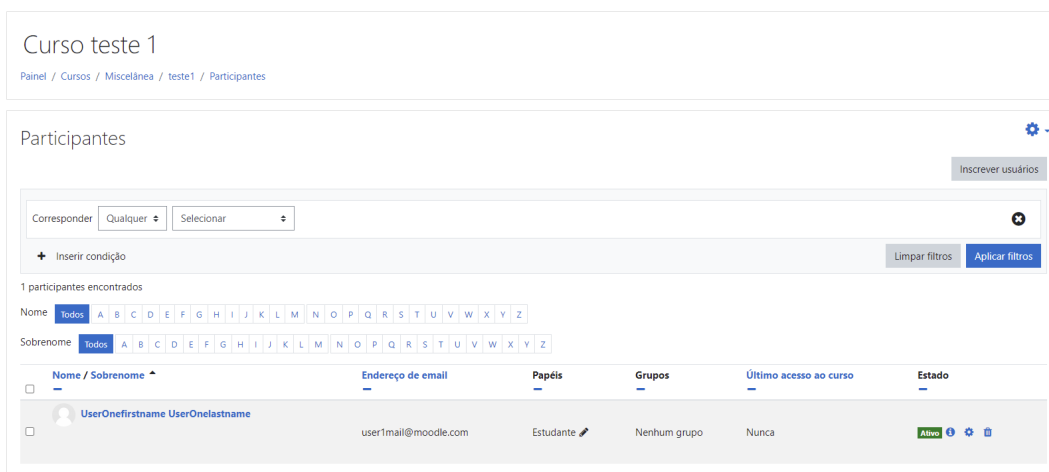
```

O padrão de desenvolvimento e execução do serviço *enrol_user.php* é semelhante ao *create_course.php*, com diferenças na seleção do *token* e definição da função nas linhas 3 e 6 respectivamente, e também na organização central do serviço, entre as linhas 10 e 15. Pois o serviço *enrol_user.php* primeiro define qual será o papel do usuário em questão, na linha 11; cuja opção selecionada é 5 para defini-lo como estudante. Na linha 12 é feita a seleção do usuário pelo id e na linha 13 a seleção do curso, também pelo id, que o usuário será matriculado.

Após isso é organizado a estrutura do serviço de acordo com os parâmetros da API; um vetor de vetores com nome de índice "*enrolments*", nas linhas 14 e 15.

O restante do desenvolvimento do código, das linhas 18 a 25 é o mesmo, pois o padrão para uso do protocolo REST, conexão com o servidor e parâmetro para chamada da função é mantido o mesmo.

A Figura 12 foi capturada do ambiente de testes após executados os serviços acima mencionados, demonstrando como resultado da execução o estudante "user1" inscrito no curso "Curso teste1".



Curso teste 1

Painel / Cursos / Miscelânea / teste1 / Participantes

Participantes

Inscriver usuários

Corresponder Qualquer Seleccionar

+ Inserir condição

Limpar filtros Aplicar filtros

1 participantes encontrados

Nome Todos A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Sobrenome Todos A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

	Nome / Sobrenome	Endereço de email	Papéis	Grupos	Último acesso ao curso	Estado
<input type="checkbox"/>	UserOnefirstname UserOnelastname	user1mail@moodle.com	Estudante	Nenhum grupo	Nunca	Ativo

Figura 12 – Inscrição do usuário

7 CONSIDERAÇÕES FINAIS

A demanda por integração entre sistemas consolidados pertencentes as instituições de ensino tem aumentado significativamente, processo caracterizado principalmente pela busca de inovação tecnológica e melhora na qualidade dos serviços prestados, utilizando para isso ambientes virtuais de aprendizagem. Assim no meio acadêmico tornou-se relevante fazer que sistemas de gestão acadêmica possam se comunicar de maneira eficiente e dinâmica com ambientes virtuais de aprendizagem. Ambos sistemas tem fundamental importância na prestação de ensino da instituição, porém, em geral, não existe comunicação entre eles, ou quando existe esta é bem restrita.

Foi observado pelo trabalho que a arquitetura de Web Service se mostrou a mais adequada para cumprir o objetivo de integração proposto pelo modelo desenvolvido, a integrar ambos sistemas de maneira eficiente e dinâmica, pois com a metodologia de criação é possível fazer uso do recursos operados pelo Moodle, visto que o ambiente implementa o web service com uma lista de APIs que promovem a integração das principais demandas.

Quanto ao sistema que integra com o Moodle, o SGA genérico, este precisa fornecer os dados de acordo com os parâmetros caracterizados por cada função, e dentro das suas vantagens é válido destacar duas; primeiro para integração é preciso desenvolver as *triggers* a fornecer à informação ao serviço, e este fará a tarefa de integrá-lo ao Moodle. A segunda é oportuno mencionar que futuras alterações no Moodle, quanto as novas funcionalidades que podem ser implementadas e atualizações de versionamento, serão mais facilmente tratadas pelo modelo de integração, uma vez que a integração não é restrita à poucas demandas específicas, e também as inserções não são feitas diretamente no sistema do Moodle, tendo em vista que as operações de integração serão realizadas via APIs desenvolvidas pelo ambiente.

Como trabalhos futuros, é sugerido desenvolver o integrador seguindo a metodologia proposta pelo modelo de integração, desenvolver também as *triggers* para comunicação entre o SGA e o integrador, a criar um ambiente de integração entre SGA e Moodle, provendo as demandas de integração até conseguir colocá-las em desenvolvimento na instituição, visando todos benefícios citados, na esfera da administração e acadêmica.

REFERÊNCIAS

- 1 CONEXOS. **Integração entre sistemas: conhecendo os tipos mais utilizados.** 2017. Disponível em: <<https://blog.conexos.com.br/tipos-de-integracao-entre-sistemas/>>. (acesso: 10.10.2019).
- 2 DEVMEDIA. **vantagens e desvantagens de soa.** 2013. Disponível em: <<https://www.devmedia.com.br/vantagens-e-desvantagens-de-soa/27437>>. (acesso: 10.25.2019).
- 3 EULÁLIO, Athos Denis. Modelo de integração entre Ambientes Virtuais de Aprendizagem e Sistemas de Gestão Acadêmica baseado em Arquitetura Orientada a Serviços e Computação em nuvem, 2016.
- 4 FURTADO, Ulisses de Melo; LIMA, Rommel Wladimir de. Modelo de integração adaptável entre ambientes virtuais de aprendizagem e sistemas de gestão acadêmica baseado em arquitetura orientada a serviços. **Em rede, Revista de Educação a Distância**, 2015, 2015.
- 5 LUCENA, Maxwell de Paiva. IM-SIGA: PLATAFORMA DE INTEGRAÇÃO MOODLE-SISTEMA DE GESTÃO ACADÊMICA BASEADA EM WEBSERVICE, 2013.
- 6 MOODLE. **About Moodle.** 2018. Disponível em: <https://docs.moodle.org/37/en/About_Moodle>. (acesso: 28.10.2019).
- 7 _____. **Features Moodle.** 2019. Disponível em: <<https://docs.moodle.org/37/en/Features>>. (acesso: 03.11.2019).
- 8 MOURA, Ricardo; BERNARDINO, Jorge. Um modelo para a integração de serviços: Moodle e Sistemas de Gestão Acadêmica. **RISTI, Revista Ibérica de Sistemas e Tecnologias de Informação, Porto, v. 5, p. 31-44, 2010**, 2010.
- 9 OLIVEIRA, José Martins Junior; CORREA, Luan Pereira do Nascimento; FONSECA, Luís Carlos Costa. Integração do Moodle com Sistema de Gestão Acadêmica, 2018.
- 10 SILVA, César Augusto Barreto da. ARQUITETURA EMPRESARIAL: UM ESTUDO DE CASO SOBRE A INTEGRAÇÃO ENTRE A PLATAFORMA MOODLE E O SIGAA NA UFRN, 2012.
- 11 WITTEL. **Entenda os desafios de integração entre sistemas de comunicação.** 2017. Disponível em: <<https://blog.wittel.com/integracao-entre-sistemas/>>. (acesso: 28.10.2019).