

Avaliação de ferramentas para implantação automática de Business Intelligence¹

José Ismael dos Santos Falcone, Rodrigo de Almeida Vilar de Miranda

Departamento Ciências Exatas (DCX) – Universidade Federal da Paraíba
(UFPB) Rua da Mangueira, s/n, Companhia de Tecidos Rio Tinto CEP
58297-000 - Rio Tinto - PB - Brasil

{ jose.ismael, rodrigovilar } @dcx.ufpb.br

Abstract. *The advance of business intelligence supports business competitiveness, but there is still a major disadvantage for small businesses that do not have the resources to invest in this type of system to support decision making. Deploying business intelligence is costly because it depends on an expert in the field. In this paper, the Scaffolding technique is analyzed in order to automate the deployment of a business intelligence solution through code generation, reducing its cost. The main results of this work are: a survey of objective criteria to evaluate business intelligence tools capable of being integrated with code generators; the definition of an architecture for the solution; and the creation of a proof of concept to validate the architecture.*

Resumo. O avanço da *business intelligence* dá suporte à competitividade empresarial, contudo ainda há uma grande desvantagem para as pequenas empresas que não têm tantos recursos para investir nesse tipo de sistema para suporte à tomada de decisão. A implantação de *business intelligence* tem custo elevado, pois depende de um especialista na área. Neste trabalho, a técnica de *Scaffolding* é analisada a fim de automatizar a implantação de uma solução de *business intelligence* através da geração de código, reduzindo o seu custo. Os principais resultados deste trabalho são: um levantamento de critérios objetivos para avaliar ferramentas de business intelligence capazes de serem integradas a geradores de código; a definição de uma arquitetura para a solução; e a criação de uma prova de conceito para validar a arquitetura.

1. Introdução

O conceito de *Business Intelligence* (BI) é reconhecido hoje como um dos mais importantes para ajudar as organizações no suporte à tomada de decisão. Por meio do BI, organizações podem utilizar os seus dados para montar estratégias e ter *insights* mais inteligentes, fazendo uma análise mais focada nos objetivos, e reduzindo tempo para tomada de decisões, o que pode ser vital para o negócio (Primak, 2008).

Nesse sentido, é possível analisar fatos ocorridos objetivamente e por diversas óticas, como frequência do fato, quem estava envolvido, quando ocorreu, onde

¹ Trabalho de Conclusão de Curso (TCC) na modalidade Artigo apresentado como parte dos pré-requisitos para a obtenção do título de Bacharel em Sistemas de Informação pelo curso de Bacharelado em Sistemas de Informação do Centro de Ciências Aplicadas e Educação (CCAIE), Campus IV da Universidade Federal da Paraíba, sob a orientação do professor Rodrigo de Almeida Vilar de Miranda.

aconteceu. Segundo Primak (2008), implantar sistemas de BI em organizações privadas ou públicas pode ser uma fonte de informações valiosas para uma organização.

Para criar uma solução de BI, primeiramente é necessário extrair e processar os dados do negócio. Posteriormente é preciso analisar e visualizar os dados criando assim cubos *OLAP (On-Line Analytical Processing)* para fazer cruzamentos de grandes volume de dados em tempo de execução. Também podem ser criados *Dashboards* para analisar os indicadores chaves de desempenho (do inglês *KPI - Key Performance Indicators*) que geralmente são as metas do negócio.

Existem várias ferramentas de BI no mercado. Algumas proprietárias como o Oracle Business Intelligence Enterprise Edition (OBIEE²) ou Power BI da Microsoft³, Pentaho⁴ e Qlik⁵, que têm versões de testes ou limitações em sua versão gratuita. Por outro lado, algumas ferramentas, como o Pentaho Community Edition, são gratuitas ou são disponibilizadas através de código aberto. De uma maneira geral, as ferramentas seguem os mesmos conceitos para transformar dados em informação.

Todavia ferramentas de BI não são tão simples pois demandam um custo de implantação normalmente elevado, dependendo das necessidades do negócio e do volume de dados armazenado. Sendo necessário um especialista para implantar e configurar sistemas de BI. Dessa maneira, utilizar BI pode ser inviável para organizações de pequeno porte.

Segundo Popovic e Jaklic (2010), o uso de BI tem sido privilégio de grandes empresas, mesmo que a adoção dessa técnica esteja em ascensão na América Latina. Nessa perspectiva, percebe-se a necessidade de **popularizar, baratear e desenvolver mais rápido sistemas de BI.**

De acordo com Vilar (2017), o barateamento no desenvolvimento de software já ocorreu em diversos tipos de sistemas, utilizando técnicas modernas de Engenharia de Software. Por exemplo, o desenvolvimento de sistemas web demandava especialistas e era caro, mas segundo Magno com a técnica de Scaffolding os programadores especializados em web foram substituídos por geradores automatizados de código, que recebem metadados e geram o código inicial de aplicações web. Ruby on Rails⁶, jHipster, Grails⁷ são exemplos de geradores de código para sistemas web, que reduzem o esforço de desenvolvimento e habilitam programadores mais inexperientes a entregar produtos para seus clientes.

²<https://www.oracle.com/middleware/technologies/bi-enterprise-edition.html>

³<https://powerbi.microsoft.com/pt-br/get-started/>

⁴<https://community.hitachivantara.com/s/pentaho>

⁵<https://www.qlik.com/pt-br/try-or-buy/download-qlikview>

⁶<https://rubyonrails.org/>

⁷<https://grails.org/>

Diante desse contexto, a questão de pesquisa deste trabalho pode ser definida como: **É possível evoluir um gerador de código, semelhante a *Scaffolding*, para automatizar a implementação de sistemas de BI?**

O objetivo geral desta pesquisa é projetar e também avaliar a viabilidade de uma arquitetura de software para implementar uma solução de BI associada a geradores de código.

Os objetivos específicos deste trabalho são:

- Levantamento de critérios objetivos e ferramentas para que possam ser integradas a geradores de código a fim de automatizar a implantação de BI;
- Projeto de uma arquitetura para uma solução de BI automática com a ferramenta escolhida no passo anterior;
- Implementação e validação de uma prova de conceito da arquitetura proposta.

O restante do trabalho está estruturado nas seguintes seções: fundamentação teórica, metodologia, desenvolvimento e conclusão.

2. Fundamentação teórica

Esta seção apresenta conceitos que serviram de base para análise e proposta de solução neste trabalho. Inicialmente uma introdução ao conceito de *Business Intelligence* e depois uma explanação sobre o uso de metadados para abstrair conceitos de software e geradores de código baseados na técnica de *Scaffolding*.

2.1 Business Intelligence

De acordo com o que ensina Primak (2008), por volta da década de 80, surgiu no Gartner Group o termo *Business Intelligence*, alusivo ao processo de coleta, análise, compartilhamento e monitoração de dados, gerando informações para suporte a tomadas de decisões. Nos dias atuais, corporações de pequeno, médio e grande porte precisam de auxílio, através de BI, para as mais diferentes situações como: otimizar o trabalho da organização, reduzir custos e eliminar a duplicação de tarefas. Dessa maneira, permitindo análises preditivas e uma visão sistêmica da empresa, contribuindo assim para elaboração de estratégias.

Primak também afirma que há muitas opiniões que nem sempre são unânimes, já que consultores e analistas dizem que para implementar ferramentas de análise, precisa-se de um repositório específico para reunir os dados já transformados em informações. Esse repositório não precisa ser necessariamente um Data Warehouse,

mas sim algo menos complexo como, por exemplo, um Data Mart, um banco de dados desenhado de forma personalizada para assuntos ou áreas específicas.

Conforme Piton (2017), BI ajuda a responder fatos ocorridos na organização, e para responder esses fatos de maneira mais rápida, é utilizado o método 5w3h, que é uma ferramenta da área de administração que serve basicamente para fazer um *checklist* com perguntas chaves que os *key users* estão mais acostumados a ouvir para identificação do fato. Nesse sentido, o método é baseado em oito perguntas, utilizadas para implementar soluções. Com isso conseguimos entender **qual** fato aconteceu, **quem** estava envolvido, **onde** aconteceu, **quando** ocorreu, **por que** aconteceu, **como** aconteceu, **quanto custou** e a **frequência**.

Com essas perguntas é possível projetar uma solução de BI, identificando várias situações que possam estar ligadas na organização, como a relação de vendas de um produto com um determinado vendedor, ou as notas de alunos e a associação com uma determinada disciplina. Nesse contexto, é possível criar vários cenários e analisar os fatos sobre diversas óticas. Com isso, criando uma visão sistêmica e baseada nas necessidades reais do cliente gerando um melhor entendimento do fato que aconteceu.

2.2 Metadados

Segundo Niso (2004), metadados são informações que explicam, localizam ou facilitam a manipulação de uma fonte de informação. Pode se dizer que metadados são dados que representam dados. Através desse conceito, é possível identificar objetos através de seus elementos como nome, título e ISBN, que representa por exemplo o objeto Livro, ou a organização estrutural de um objeto, e até mesmo dados que envolvem permissões de acesso e segurança.

Os metadados podem ser utilizados no código-fonte de um software para torná-lo genérico, pois elementos do software podem ser parametrizados, sendo posteriormente processados por um gerador de código. Dessa forma, em vez de implementar vários componentes de software, como controladores, telas, serviços e repositórios de dados, o programador precisa apenas lidar com metadados, por exemplo: um cadastro de clientes é composto por nome, CPF e data de nascimento (FRANKY; PAVLICH-MARISCAL, 2012).

Assim sendo, pode-se, em teoria, combinar os metadados de um domínio de problema às perguntas (5w3h) que podem ser respondidas por uma solução de BI a fim de gerar o código de BI automaticamente.

2.3 Scaffolding

A técnica de *Scaffolding* foi popularizada pela plataforma *Ruby on Rails* e permite que se implemente uma aplicação CRUD web funcional em minutos, simplesmente mapeando metadados de objetos, banco de dados e interfaces (ADAMUS et al., 2010).

Nesse sentido, o termo *Scaffolding* é empregado para programas que utilizam a geração automática de código fonte baseado em *templates*. Esses *templates* são modelos de código-fonte de projetos bem sucedidos, podendo assim ser considerados padrões para desenvolvimento (FRANKY; PAVLICH-MARISCAL, 2012).

Assim sendo, *Scaffolding* pode ser compreendido como uma forma de automatizar atividades rotineiras. Desse modo, criando automaticamente páginas de listagem, formulários, controladores REST, tabela no banco de dados, etc. A programação se torna mais produtiva e, além disso, há a possibilidade de outros tipos de arquivos serem também gerados por *Scaffolding*, tais como arquivos XML de configuração e folhas de estilo. Em resumo, não é preciso ser um especialista para fazer aplicações web facilmente.

3. Metodologia

Para responder a questão de pesquisa – se é possível evoluir um gerador de código, semelhante a *Scaffolding*, para automatizar a implementação de sistemas de BI – a metodologia utilizada neste trabalho se baseou em *Design Science Research* (DSR), que é a interação de duas atividades, projetar um artefato que melhora algo para os *stakeholders* e investigação empírica sobre o desempenho desse artefato em um contexto. Nesse sentido, artefatos podem ser considerados métodos, técnicas, notações, algoritmos, e o contexto pode ser entendido como projeto, desenvolvimento, manutenção de software (WIERINGA, 2014). Especificamente neste trabalho, se está estudando a relação dos artefatos geradores de código no contexto de *Business Intelligence*.

Nesta pesquisa, serão realizados os seguintes procedimentos:

- i) Estudar *Business Intelligence* e a técnica de *Scaffolding*;
- ii) Propor critérios objetivos para avaliação de ferramentas de BI que possam ser automatizadas e integradas com geradores de código;
- iii) Realizar uma busca sobre ferramentas de BI no Google com termos previamente definidos;

iv) Avaliação de ferramentas encontradas no Google conforme os critérios propostos no item ii) e escolha da ferramenta melhor avaliada;

v) Projetar e implementar uma prova de conceito para uma solução de BI automatizada baseada na ferramenta escolhida no passo iv) e avaliá-la.

A seguir cada um dos procedimentos metodológicos será detalhado.

3.1 Estudar Business Intelligence e a técnica de Scaffolding

Esta parte da pesquisa iniciou através de buscas nos portais de pesquisa Google e Google Scholar utilizando palavras chave como “Business Intelligence”, “BI”, “Business Intelligence open source”, além disso, um curso profissionalizante na suíte Pentaho *Community Edition* durante o período de 2 meses, a fim de entender inicialmente conceitos e funcionalidades.

Logo após a pesquisa sobre BI, começou a fase de investigação sobre a técnica de *Scaffolding*, aplicando as palavras chave como “*Scaffolding*” e “Geração de Código”. Os resultados da pesquisa encontram-se nas Subseções 2.2 e 2.3 onde foram abordados os conceitos e fundamentos.

3.2 Levantamento de critérios objetivos para avaliação das ferramentas

Com o objetivo de pesquisar e analisar as ferramentas que possam ser integradas a um gerador de código, foram definidos, pelo autor deste trabalho, 9 critérios objetivos, com pesos individuais para avaliação das ferramentas de BI. Desse modo, facilitando a escolha da ferramenta para prova de conceito. Os critérios definidos são mostrados na Tabela 1.

A seguir, cada critério será explicado, juntamente com os pesos alocados: 1 para critérios desejáveis mas opcionais; 2 para critérios importantes; e 3 para critérios essenciais ou muito importantes.

Possuir código aberto. Se uma ferramenta de código aberto for selecionada neste trabalho, ela pode ser analisada e adaptada para o gerador de código com maior facilidade. Este fator é muito importante para a prova de conceito a ser realizada, portanto recebeu peso 3.

Ser Gratuita. Uma das motivações gerais da pesquisa é reduzir custos, portanto as ferramentas gratuitas serão priorizadas e este critério possui peso 3.

Possuir arquitetura escalável. Tendo em vista o avanço dos sistemas de informação e grande quantidade de dados nas empresas, possuir uma arquitetura

escalável é importante para evitar lentidão nas funcionalidades para o usuário chave da BI. Assim sendo, a este critério foi atribuído peso 2.

Tabela 1. Critérios objetivos para avaliação das ferramentas.

Id	Descrição do critério	Peso
1	Possui código aberto?	3
2	É Gratuita?	3
3	Possui arquitetura escalável (requisições por segundo)?	2
4	Possui poucas dependências (tamanho em MB) facilitando a montagem de um container?	2
5	Permite integração na interface de um SI existente?	2
6	Permite integração na autenticação de um SI existente?	1
7	Possui boa documentação e comunidade ativa?	2
8	Possui Interface web e API REST?	2
9	Há limitações na conta gratuita?	1

Fonte: autor

Possuir poucas dependências. Para facilitar a geração de código automatizada em um sistema de BI, e implantá-lo em servidor web barato, um dos critérios mais importantes é que a ferramenta possua poucas dependências (medidas aqui como tamanho em megabytes), para quem em poucos passos consiga colocá-lo em execução através de um container. Dessa maneira, este critério foi definido com peso 2.

Permitir integração na interface de um SI existente. Com o intuito de evitar várias etapas até que o usuário consiga visualizar as informações no sistema de BI, o critério de integração na interface de um SI existente foi criado, a fim de que o tomador de decisões tenha uma melhor experiência para conseguir visualizar as informações, em função disso, este critério recebeu peso 2.

Permitir integração na autenticação de um SI existente. Permitir a autenticação integrada ao SI foi um dos critérios definidos, pelo mesmo motivo do item anterior, evitar uma experiência cansativa para que o usuário consiga visualizar as informações geradas com o sistema de BI. Este critério recebeu peso 1 por ser uma extensão do item anterior.

Possuir boa documentação e comunidade ativa. Tendo em vista a criação de novas funcionalidades e evolução da ferramenta de BI, o critério de possuir boa

documentação e uma comunidade ativa, foi motivado para facilitar a vida do desenvolvedor em caso de dúvidas. Ter uma comunidade ativa também é um indicativo dos benefícios da ferramenta e crescente interesse de profissionais. Sendo assim, a este critério foi atribuído peso 2.

Possui Interface web e API REST. Possuir uma interface web gera várias vantagens que podem ser fundamentais para uma sistema que trabalha com dados. O tomador de decisão não precisa se prender ao local de trabalho para conseguir ver as informações do negócio, inclusive o SI não terá necessidade de ser instalado em diversos equipamentos diferentes. Outro fator também é a escalabilidade, caso precise de maior poder de processamento, apenas irá configurar o servidor. Por isso, este critério recebeu peso 2.

Não haver limitações na conta gratuita. Caso uma ferramenta tenha limitações na conta gratuita, pode reduzir a compreensão de informações geradas a partir dos dados, por consequência limitar o poder da solução de BI. Devido ao seu caráter desejável, este critério recebeu peso 1.

3.3 Busca de ferramentas de BI

Os termos “**Business Intelligence free tools**” e “ **Business Intelligence free libs**” foram usados como termos de busca para pesquisa de ferramentas, em 2 etapas no portal de pesquisa Google, onde foram mapeados os 10 primeiros sites para cada termo buscado, excluindo os que estavam no topo da página como anúncios. Em cada site encontrado, foram anotados os nomes das ferramentas citadas. Contabilizado a quantidade de vezes em que as ferramentas apareceram nas duas etapas da pesquisa, pode-se mapear as 10 ferramentas mais citadas.

3.4 Avaliação das ferramentas encontradas

Com o objetivo de escolher a ferramenta mais apropriada para a pesquisa, foi necessário fazer uma investigação na documentação das ferramentas encontradas no procedimento anterior e elaborar um quadro comparativo das 10 ferramentas, utilizando os critérios e pesos definidos na Subseção 3.2.

Para cada ferramenta, foi necessário responder se ela atendia ou não aos critérios. Para fins de legibilidade, o quadro comparativo utilizou cores além do score para cada critério. Portanto, a resposta “Sim” possui a cor verde e score 2; “Não” possui cor vermelha e score 0; e “Não encontrado”, cor amarela e score 1.

A pontuação total é calculada multiplicando o escore da resposta pelo peso da linha (critério). Ao final, é somado o valor total da coluna em cada ferramenta, segundo a fórmula:

$$E_F = \sum_{i=1}^9 (e_i^F \times p_i)$$

onde, E_F representa o Escore geral da ferramenta F, e_i^F significa o Escore da ferramenta F no critério e p_i é o peso do critério i.

3.5 Proposta de uma Arquitetura para Deploy automático de BI

Após a fase de avaliação das ferramentas encontradas, será feita uma proposta de arquitetura para o deploy automático de BI, utilizando os elementos necessários para solução, representados em um diagrama de componentes UML.

3.6 Prova de conceito para Deploy automático de BI

Seguindo a arquitetura definida na Subseção 3.5, será utilizado Metabase para BI e Docker para containerização, integrados a uma aplicação simplificada de notas de alunos em disciplinas.

4. Desenvolvimento

Nesta seção, se explica o desenvolvimento para responder à questão de pesquisa levantada na Introdução. Será mostrado: o resultado e análise das ferramentas encontradas para automação do deploy de BI; a arquitetura proposta para resolver este problema; a prova de conceito implementada com Metabase⁸ e Docker⁹; os exemplos de invocações REST¹⁰/JSON¹¹ para customização de gráficos de BI no Metabase; e, por fim, a generalização das invocações REST através de idiomas.

4.1 Resultado e análise das ferramentas encontradas

Na Tabela 2, estão sendo mostradas as ferramentas mais citadas e a soma da quantidade de vezes que cada uma apareceu nas duas etapas da pesquisa definida na Subseção 3.3.

⁸ <https://www.metabase.com/>

⁹ <https://www.docker.com/>

¹⁰ <https://restfulapi.net/>

¹¹ <https://www.json.org/json-pt.html>

A Tabela 3 representa o quadro comparativo das ferramentas encontradas, que foi montado segundo o procedimento definido na Subseção 3.4.

Tabela 2. Ferramentas de encontradas e quantidade de vezes que apareceram nas pesquisas.

Ferramentas	Contagem total
Microsoft Power BI	11
ReportServer	11
Qlik	10
Eclipse BIRT	10
Tableau	10
Seal Report	8
Metabase	7
KNIME	7
Pentaho	7
SpagoBI	6

Fonte: autor

Tabela 3. Comparação das ferramentas de BI, baseado nos critérios objetivos.

Critérios	Peso	Microsoft Power BI	Report Server	Qlik	Eclipse BIRT	Tableau	Seal Report	Metabase	KNIME	Pentaho	SpagoBI
Possui código aberto?	3	0	2	0	2	0	2	2	2	2	2
É Gratuito?	3	2	2	0	1	0	1	2	2	2	2
Possui arquitetura escalável (requisições por segundo)?	2	2	2	0	2	2	2	1	2	2	2
Possui poucas dependências (tamanho em MB) facilitando a montagem de um container?	2	0	2	0	0	0	0	2	0	0	0
Permite integração na interface de um SI existente?	2	2	1	2	2	0	2	2	2	2	1
Permite integração na autenticação de um SI existente?	1	0	1	1	1	1	1	1	1	1	1
Possui boa D\documentação e comunidade ativa?	2	2	2	2	2	2	1	2	1	2	1
Possui Interface web e API REST?	2	2	1	2	2	2	0	2	1	2	1
Há limitações na conta gratuita?	1	2	2	2	0	2	2	2	2	2	2
PontuaçãoTotal		24	31	15	26	15	22	33	27	31	25

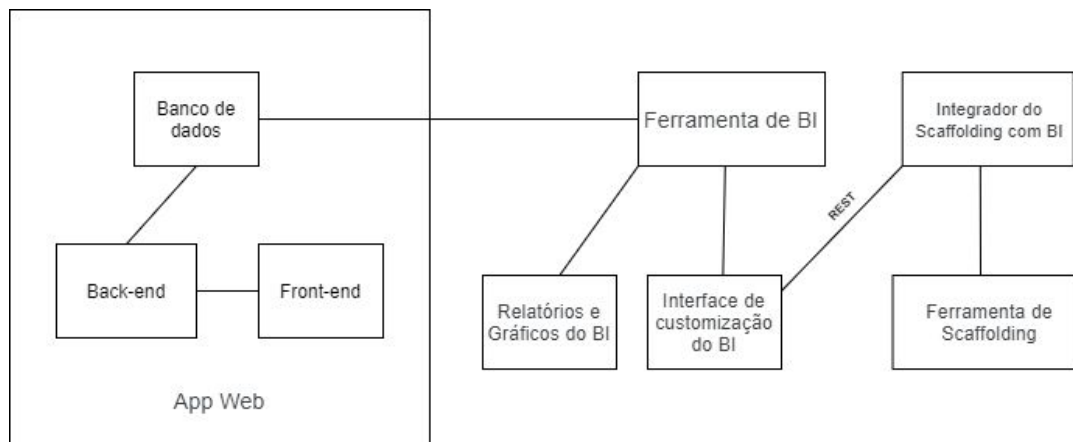
Fonte: autor

Com o resultado, é possível identificar que a ferramenta **Metabase** obteve maior pontuação, com valor resultante de 33 pontos. Através do cálculo da pontuação total é possível escolhê-la como a ferramenta para proposta de arquitetura para solução e para a prova de conceito.

4.2 Arquitetura

O diagrama de componentes UML na Figura 1, mostra os elementos necessários para criação de gráficos e relatórios automatizados. Há uma aplicação web desenvolvida com *front-end*, *back-end* e banco de dados, a ferramenta de BI se conecta ao banco de dados da aplicação. Uma ferramenta de Scaffolding pode fazer uma requisição REST, através de um integrador, a fim de customizar os gráficos ou relatórios de BI a serem disponibilizados para os usuários finais.

Figura 1. Diagrama da arquitetura de software proposta.



Fonte: autor

O App Web é uma aplicação CRUD simplificada para matrícula e cadastro de notas de alunos em disciplinas com o ano da conclusão, a aplicação é desenvolvida em Angular¹² e Spring Boot¹³. Abaixo nas Figuras 2 e 3 é possível ver as telas de listagem e cadastro.

Figura 2. Listagem das matrículas.

¹² <https://angular.io/>

¹³ <https://spring.io/projects/spring-boot>

Matriculas						
Busca Avançada: <input type="text" value="Busca"/>						Show: 10
Id	Aluno	Disciplina	Nota	Ano Conclusao	Ações	
1	Neto Falcone	Geografia	10.0	2019	  	
3	Maria dos Santos	História	9.0	2018	  	
4	Maria Clara	Geografia	9.0	2019	  	
5	Pedro dos Santos	Geografia	10.0	2019	  	
6	João Augusto	Geografia	10.0	2019	  	

Fonte: autor

Figura 3. Formulário de cadastro da aplicação web.

Cadastrar

Aluno:

Disciplina:

Nota:

Ano Conclusao:

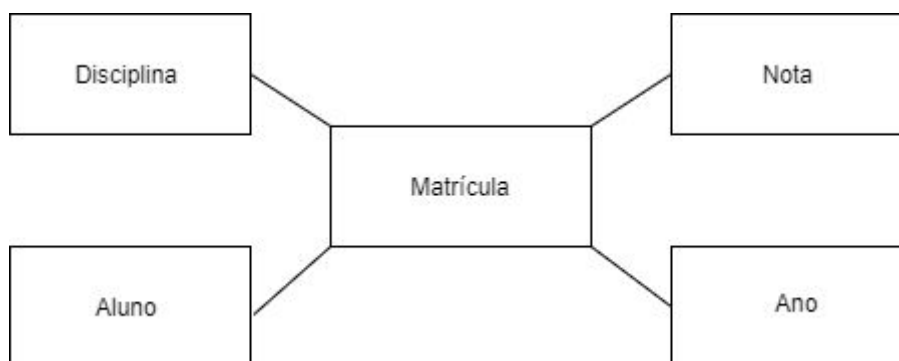
VOLTAR

CADASTRAR

Fonte: autor

Abaixo na Figura 4 é possível observar o conceitos da aplicação web simplificada que foi utilizada neste trabalho, onde a Matrícula é a entidade sendo cadastrada e aluno, disciplina, nota e ano são as propriedades dessa entidade.

Figura 4. Diagrama com entidade e propriedades.



Fonte: autor

4.3 Prova de conceito com Metabase e Docker

A fim de validar o artefato proposto, a arquitetura do software, segundo a metodologia DSR, uma prova de conceito foi implementada com as tecnologias: Angular para o front-end, Spring Boot para o back-end e PostgreSQL para o Banco de dados. Essas são tecnologias comuns na Indústria e foram implementadas conforme as referências encontradas na comunidade de usuários. A parte principal do projeto, com foco na automação do deploy de BI, foi implementada com Metabase, que foi configurado e publicado através de um container Docker.

Com o propósito de gerar gráficos customizados foi utilizada a API do Metabase. Mas antes da invocação dos métodos REST para criação dos gráficos, foi necessário criar uma imagem do Metabase no Docker com o super usuário já configurado, pois as solicitações exigem um *token de sessão*, que pode ser obtido chamando POST /api/session.

Dessa forma, havia uma API REST disponível no container para criar e customizar gráficos e relatórios de BI. Para simular a integração do Scaffolding com o BI, foi utilizada a ferramenta Insomnia, através da qual se submeteu o conteúdo REST que pode ser visto na Subseção 4.5.

4.4 Exemplos de invocações REST/JSON para customização de gráficos no Metabase

A solução desenvolvida possibilita a automatização de procedimentos como: geração de gráficos de média, contagem, valor mínimo, máximo e percentual. Todos com os respectivos atributos e propriedades da entidade, além de gerar o código semelhante ao que é produzido manualmente.

No código abaixo há um exemplo de uma invocação REST chamando POST /api/card/. Com esse JSON é gerado um gráfico de barras mostrando a menor, maior e a nota média de matrículas em uma disciplina por ano.

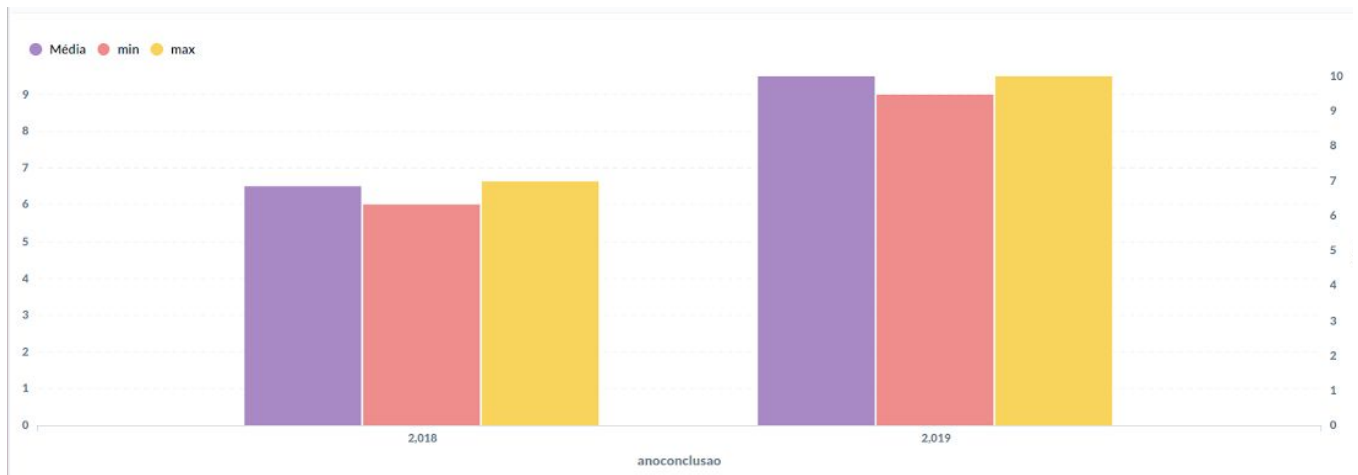
Listagem 1. Exemplo de código JSON para criação de gráfico com a nota média, mínima e máxima.

```
{
  "visualization_settings": {
    "graph.dimensions": [ "anoconclusao" ],
    "graph.metrics": [ "avg", "min", "max" ],
    "graph.show_values": true
  },
  "description": "Descrição - Nota média, mínima e máxima Geografia - tcc",
  "collection_position": null,
  "result_metadata": null,
  "metadata_checksum": null,
  "collection_id": 3,
  "name": "Nota média, mínima e máxima Geografia - tcc",
  "dataset_query": {
    "database": 5,
    "type": "native",
    "native": {
      "query" : "SELECT public.matricula.anoconclusao AS anoconclusao,
                avg(public.matricula.nota) AS avg, min(public.matricula.nota)
                AS min,
                max(public.matricula.nota) AS max
                FROM public.matricula
                WHERE (public.matricula.deleted = FALSE
                AND public.matricula.disciplina = 'Geografia')
                GROUP BY public.matricula.anoconclusao
                ORDER BY public.matricula.anoconclusao ASC"
    }
  },
  "display": "bar"
}
```

Mesmo sendo uma aplicação simplificada, nesse JSON é possível perceber que as partes mais importantes são elementos que identificam o objeto, como a entidade matricula, e suas respectivas propriedades como nota, anoconclusao e disciplina. Além desse exemplo, há outros quatro exemplos que foram criados para calcular contagem

total, percentual de alunos por disciplina, todos exemplos estão em um repositório chamado ResquestsMetabase¹⁴ no Github.

Figura 2. Gráfico de Barras com a nota maior, menor e média de matrículas em uma turma por ano.



Fonte: autor

4.5 Generalização das invocações REST através de idiomas

Para que se possa fazer as inovações REST de maneira generalizada, é necessário criar idiomas com os templates, os elementos do código que foram identificados e dão sentido ao JSON, foram transformados em metadados, como matrícula que foi transformado em entity, e os demais foram transformados em properties como nota, anoconclusao e disciplina. Abaixo segue o idioma que foi criado a partir do template.

Dessa maneira o idioma pode ser facilmente convertido para um template de geração de código.

Listagem 2. Exemplo de idioma para criação de gráfico com a nota média, mínima e máxima.

```
{
  "visualization_settings": {
    "graph.dimensions": [ "{{property}}" ],
    "graph.metrics": [ "avg", "min", "max" ],
    "graph.show_values": true
  },
}
```

¹⁴ <https://github.com/netofalcone/RequestsMetabase>

```

"description": "Descrição - Nota média, mínima e máxima {{property}} - tcc",
"collection_position": null,
"result_metadata": null,
"metadata_checksum": null,
"collection_id": 3,
"name": "Nota média, mínima e máxima {{property}} - tcc",
"dataset_query": {
  "database": 5,
  "type": "native",
  "native": {
    "query" : "SELECT public.{{entity}}.{{property}} AS {{property}},
              avg(public.{{entity}}.{{property}}) AS avg,
              min(public.{{entity}}.{{property}}) AS min,
              max(public.{{entity}}.{{property}}) AS max
FROM public.{{entity}}
WHERE (public.{{entity}}.{{property}} = FALSE
      AND public.{{entity}}.{{property}} = 'property')
GROUP BY public.{{entity}}.{{GroupProperty}}
ORDER BY public.{{entity}}.{{GroupProperty}} ASC"
  }
},
"display": "bar"
}

```

5. Conclusão e trabalhos futuros

Através da arquitetura proposta e da prova de conceito, mostrou-se que é possível integrar um gerador de código, semelhante a Scaffolding, para automatizar a implementação de sistemas de BI.

A ferramenta Metabase foi customizada por uma API REST, pela qual pode ser invocado um comando para criar gráficos e relatórios referentes a um CRUD feito em Scaffolding. É importante notar que a complexidade para compreender e preencher os metadados com as configurações necessárias é baixa.

Como trabalhos futuros, pretendemos utilizar o idioma proposto neste trabalho para criar um template capaz de integrar o gerador de código com o Metabase. Além disso, criar idiomas para tipos mais complexos de CRUDs feitos em Scaffolding. Ex: relacionamentos e dados históricos.

Referências

ADAMUS,R.;KOWALSKI,T.M.;KULIBERDA,K.;WIŚLICKI,J.;BLEJA,M.Tools Supporting generation of "data-intensive"applications for a web environment. Automatyka/Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie, v. 14, p. 951–960, 2010.

DA SILVA, Rafaela Alexandre; SILVA, Fernando Cesar Almeida; GOMES, Carlos Francisco Simões. O uso do Business Intelligence (BI) em sistema de apoio à tomada de decisão estratégica. **Revista GEINTEC-Gestão, Inovação e Tecnologias**, v. 6, n. 1, p. 2780-2798, 2016.

FRANKY, M. ;PAVLICH-MARISCAL, J.Improving implementation of code generators: A regular-expression approach.In: Informatica(CLEI),2012XXXVIIIConferencia Latino americana En. [S.l.: s.n.], 2012. p. 1–10.

NISO. Understanding metadata. Bethesda, MD, 2004. ISBN 1-880124-62-9.

PINTO, L. A.; NUNES, N. S.; MELO, Francisco V. S. Aplicabilidade do balanced scorecard (BSC) na área de tecnologia da informação (TI): um estudo de Caso. VIII SEGET Simpósio de Excelência em Gestão e Tecnologia, 2011

Piton, Rafael. **Data Warehouse – Passo a Passo para Modelar um Data Warehouse**, 2017. Disponível em: <<https://rafaelpiton.com.br/blog/data-warehouse-como-modelar/>>. Acesso em: 01/12/2019.

PRIMAK, Fábio Vinícius. **Decisões com bi (business intelligence)**. Fabio Vinicius Primak, 2008.

MAGNO, Danilo Goulart. Aplicação da Técnica de Scaffolding para a Criação de Sistemas CRUD. 2015.

VILAR, Rodrigo; OLIVEIRA, Delano; ALMEIDA, Hyggo. Rendering patterns for enterprise applications. In: Proceedings of the 20th European Conference on Pattern Languages of Programs. ACM, 2014. p.15 .

WIERINGA, Roel J. Design science methodology for information systems and software engineering. Springer, 2014.