PROPOSTA DE UMA FERRAMENTA DE APOIO PARA GERAÇÃO DE PRINCIPIOS DE SOLUÇÃO

João Marcelo Teixeira e Silva Torres (jmstorres@gmial.com) - PPMEC, UnB

Andréa Cristina dos Santos (andreasantos@unb.br) - PPMEC, UnB

RESUMO

Este trabalho tem por objetivo apresentar uma ferramenta de apoio à etapa de projeto conceitual de produtos mecatrônicos, pelo emprego de sistemas de aprendizado de máquina que se utilizam de raciocínio baseado em casos - RBC. A ferramenta consiste a partir da proposição da modelagem funcional para os produtos mecatrônicos a automatização na geração dos princípios de solução na matriz morfológica com base em casos anteriormente cadastrados. O uso específico do método RBC é motivado pela possibilidade de aproveitamento de princípios de soluções de projetos anteriores em novos problemas. Este aproveitamento, segue regras que não ofereçam restrições ao projeto e nem ao processo criativo, servindo como ferramenta e não como limitador da criatividade. Para recuperação dos dados além da classificação do tipo de função, as funções foram classificadas quanto ao contexto, denominadas de escopo. Ao sugerir soluções para as funções, a ferramenta analisa tanto gramaticalmente quanto contextualmente as condições das funções. A avaliação da ferramenta foi realizada no ambiente acadêmico, no qual as equipes de projetos foram instruídas no uso do aplicativo para desenvolvimento de princípios de solução. A avaliação mostrou resultados promissores em relação ao seu uso, tanto para reuso dos princípios de solução quanto para proposição de novos princípios de solução. O comportamento da ferramenta em relação ao crescimento da base de dados correspondeu ao esperado, com soluções mais completas surgindo com o aumento do número de casos cadastrados. Observa-se que o reuso foi voltado em grande parte à estruturação e análise de similaridade dos casos para cada função definida. A possibilidade de alterar os algoritmos e implementar modelos com melhor funcionalidade ao projeto está presente na arquitetura da ferramenta, deixando o espaço aberto para novas implementações.

.

Palavras chave: projeto conceitual, matriz morfológica, projeto mecatrônico.

Área: Ferramentas e métodos de desenvolvimento de produtos

1. INTRODUÇÃO

A fase de projeto conceitual no processo de projeto de produtos tem por objetivo a partir de um conjunto de especificações geradas, criar diferentes alternativas de soluções. A mesma é caracterizada como a fase de criatividade, na qual é empregado um conjunto de métodos e técnicas para estimular a criação de um maior número de alternativas de solução que possam resolver o problema de projeto. Para tanto se recomenda o uso de métodos sistemáticos tais como o método da síntese funcional e o emprego da matriz morfológica (ROZENFELD, et al. 2006)

No método de Síntese Funcional, a função global é desdobrada, sucessivamente, em funções parciais e elementares, no qual são considerados como processos de transformação de estado e das propriedades de grandezas do tipo de energia, material e sinal. O método da matriz morfológica tem por objetivo encontrar uma nova solução do problema partindo de uma pesquisa sistemática de diferentes combinações de elementos ou parâmetros. As combinações desses elementos servirão de inspiração para geração de novas soluções para os problemas (ROZENFELD, et al. 2006)

A importância da proposição de uma ferramenta para a fase de projeto conceitual se deve a carência de ferramentas computacionais para as fases iniciais do processo de projeto quando comparadas as outras etapas (WANG et al., 2002; CHANDRASEGARAN, 2013; SELL, TAMRI, 2016).

O objetivo deste artigo é apresentar o emprego de tecnologias de aprendizado de máquinas que se utilizam de raciocínio baseado em casos – RBC para proposição de uma ferramenta de apoio ao projeto conceitual de produtos mecatrônicos.

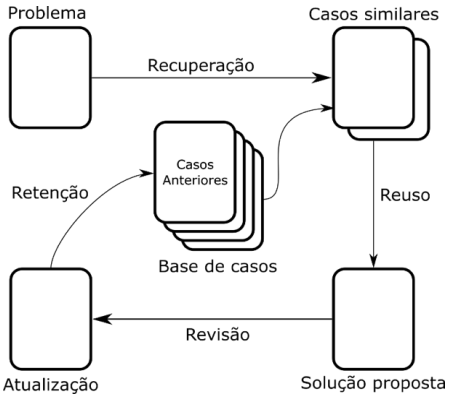
2. METODOLOGIA

Aprendizado de máquina é o treinamento de um modelo a partir de dados que generalizam uma decisão em detrimento de uma medida de desempenho. O treinamento consiste na separação de um conjunto de dados suficientemente grande para que possa ser utilizado em parte para treinar a rede no reconhecimento dos padrões desejados e em parte para validar o treinamento.

O Raciocínio Baseado em Casos (RBC) consiste em uma maneira de processar informações sobre soluções de problemas anteriores, de forma a propor com base nestas informações, novas soluções para um novo problema. Cada caso solucionado pode passar a fazer parte da base de conhecimento para uso em novas soluções. Se baseia em aspectos da cognição humana, utilizando casos anteriores análogos ao problema em questão como conhecimento para a solução deste (RICHTER, AAMDOT, 2005). O ciclo do aprendizado no RBC consiste na passagem pelas seguintes etapas (MANTARAS et al., 2006):

- Recuperação: busca de um conjunto de casos similares, comparados em relação ao um conjunto de características escolhidas para representação da informação nele contida;
- Reuso: os casos da base são então reutilizados para a geração da solução para o novo problema proposto;
- Revisão: avaliação da solução gerada para atestar sua qualidade;
- Retenção: uma solução satisfatória pode então passar a integrar a base como um novo caso.

Figura 1 – Ciclo RBC. Fonte: adaptado Mantaras et al. (2006).



A implementação de um sistema de RBC não segue uma padronização, como os sistemas de inteligência artificial em geral, mas requer adaptação da metodologia acima descrita para solucionar problemas de aprendizado em domínios e ambientes de aplicação específicos (RICHTER, AAMODT, 2005).

3. RESULTADOS

A ferramenta desenvolvida tem por objetivo cadastrar, visualizar e editar "casos", apresentando os dados de projeto de forma familiar aos usuários na forma de interfaces que ilustrem os processos de decomposição funcional e de seleção de princípios de solução para a matriz morfológica. Para cada etapa da matriz morfológica a ferramenta é capaz de propor, através do reuso de informações de projetos anteriores, princípios de solução para o projeto em desenvolvimento.

Para a ferramenta foi utilizado o framework DeepLearning4J para as etapas de aprendizado, para a base de dados o framework Hibernate de Persistência escrito em SQLITE .

3.1. Arquitetura do software

Para simplicidade foi adotada uma arquitetura MVC (Modelo, Visão e Controle) para o projeto, visto que a funcionalidade do software é simplificada para o usuário e sua verdadeira complexidade estará no cálculo das sugestões da matriz morfológica.

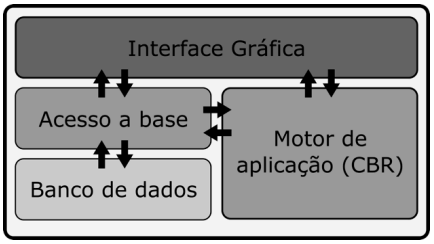
O padrão MVC separa apresentação e interação dos dados do sistema, estruturando o mesmo em três componentes lógicos que interagem entre si. O componente de modelo gerencia os dados do sistema e as operações relacionadas a estes. O componente de visão define e gerencia como os dados são apresentados ao usuário. Por fim o componente de controle gerencia as operações, repassando os comandos do usuário para os dois outros componentes (SOMMERVILLE, 2010).

Neste modelo de divisão da arquitetura temos, para o software desenvolvido:

- Modelo: forma como os dados são armazenados, compreendendo todos os CRUDs e as fórmulas de consulta aos dados e as transações de dados;
- Controle: são as ações efetuadas pela aplicação. Compreende também o motor da aplicação e os algoritmos do RBC;

- Visão: abrange a interface gráfica / apresentação dos dados;

Figura 2 – Arquitetura simplificada.



Dentro desta divisão dos componentes, o projeto completo foi organizado em três módulos para facilitar o gerenciamento do código (CBRMecaControl, CBRMecaModel e CBRMecaView). Além destes três projetos já mencionados, dois projetos auxiliares (fora do escopo de funcionamento do aplicativo final) foram criados para trabalhar os dados gramaticais utilizados para o treinamento do modelo gramatical.

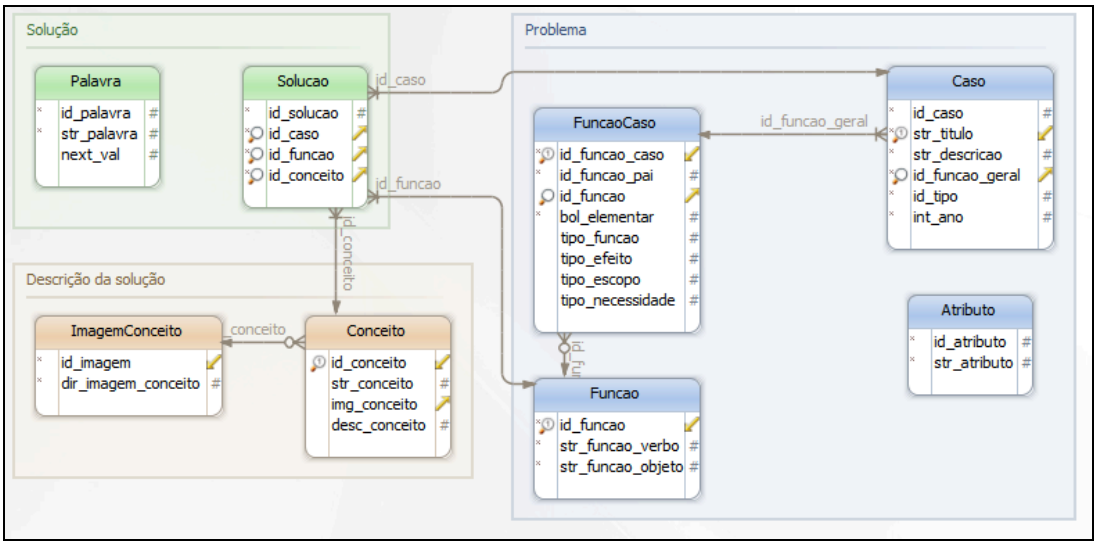
Os três projetos principais se encontram disponíveis para a comunidade em um repositório no serviço GitHub, o mesmo é aberto para interessados no trabalho.O ambiente de desenvolvimento foi montado para a codificação na linguagem de programação Java e para compilação foi utilizado o JDK versão 8 sendo a IDE de escolha o Eclipse Luna, versão 4.4.2r2.

As bibliotecas e dependências necessárias para a compilação do projeto são gerenciadas utilizando o framework Maven já integrado ao Eclipse e para todos os projetos foram criados os arquivos de configuração listando as dependências necessárias devidamente inseridas na raiz do projeto (no arquivo “pom.xml”).

3.2 Modelo

A Figura 3 ilustra a forma como os dados são armazenados. Um caso tem uma estrutura principal que pode descrever o caso como um elemento com título, descrição, um conjunto de funções e para casos solucionados, também uma lista de conceitos associados as funções elementares, cada função possui o seu conjunto de parâmetros, que é utilizado de acordo organização das características como um *feature vector*.

Figura 3 – Modelo de base de dados para armazenamento dos casos



3.3. Controle

As tarefas realizadas neste módulo são as listadas no ciclo do RBC e são elaboradas cada uma como métodos providos pelo motor. Abaixo são explicitadas as ações tomadas para o desenvolvimento de cada uma das etapas do RBC dentro da aplicação.

3.3.1. Recuperação

A maneira de representar as funções dentro do projeto então teve que ser avaliada, e dentre os fatores levados em consideração para sua classificação foram analisados quanto:

- Ao seu efeito dentro dos objetivos de projeto, servindo para amenizar ou corrigir efeitos oriundos da execução de outras funções: regular, preventiva e corretiva (VEGTE; BREEMEN, 2009);
- A sua necessidade, sendo avaliadas pela sua capacidade de servir aos propósitos do usuário: necessária e derivativa (VEGTE; BREEMEN, 2009);
- O tipo de função de produto que esta representa, de acordo com a classificação apresentada em funções técnica e funções interativas (ROZENFELD et al., 2006, p.238).

Para que a função pudesse ser avaliada no contexto de Sistemas Mecatrônicos, foi criado um parâmetro denominado "escopo" para avaliar e diferir qual o contexto disciplinar ao qual a função se adequaria. As disciplinas principais envolvidas neste contexto são descritas por Zhen et al (2014). Algumas abstrações da mecatrônica apontam o sistema mecatrônico como uma junção entre Mecânica, Software e Eletrônica (ADAMSSON, 2005)

Fazendo uma analogia com o conceito apresentado por Adamsson (2005), se define o escopo por cada uma das disciplinas envolvidas no projeto, com a adição de um item denominado "controle", especificando se o contexto possui ou não esta característica.

A avaliação do escopo do contexto foi elaborada em termos das entradas e saídas (energia, material e sinal) e as grandezas das quais trata a função. A Tabela 1 demonstra essa divisão para elucidar o conceito:

Tabela 1 - Detalhamento do escopo disciplinar das funções

Denominação	**Exemplos**	**Descrição**
Informacional / software	Processamento e armazenamento de dados	Envolve transformação de informações em meio computacional, como lógica e algoritmos.
Mecânico	Aplicar força, amortecer impacto, Suportar estrutura.	Envolve a transformação de grandezas Mecânicas como força, massa e pressão, rotação e translação.
Controle	Ajustar direção, manter estabilidade.	Envolve o trabalho de sinal, através de controle. Aplicado em conjunto com as outras três.
Potência	Filtrar sinal, alimentar equipamento, modular frequência.	Envolve a transformação de grandezas elétricas como tensão, corrente e potência.

Com estas análises é efetuada e com maior peso sobre a decisão da solução final uma análise gramatical das frases que compões as funções, (guardadas na base na forma "verbo + objeto") primeiramente avaliadas pelo verbo que as compõe, caso o verbo tenha funções com verbos similares na base, as mesmas passam a uma etapa posterior, onde são buscados dentro do

objeto novamente palavras com significados semelhantes aos que compõem o objeto da função alvo para recuperar um grau de semelhança.

A análise principal efetuada é a análise do vocabulário das funções, que apesar de ser básica foi selecionada por ter certa facilidade de desenvolvimento, com material extenso disponível e suporte nativo do *framework* selecionado para o trabalho. Para este trabalho foram gerados três modelos (também chamados de *Corpus*), cada um com aproximadamente cinco mil palavras treinadas a partir de um *dump* da *Wikipedia* utilizando o formato Word2Vec. O modelo é intercambiável, podendo ser alterado para obter melhores resultados sempre que um mais extenso ou melhor adaptado for compilado.

A caracterização dos dados por estes parâmetros inseridos no cadastro das funções tem como objetivo filtrar funções que sejam gramaticalmente semelhantes, porém com contexto de uso demasiadamente diferenciado, evitando ambiguidade causada pela comparação única dos vocábulos utilizados na primeira comparação. A similaridade entre as funções em relação ao contexto é feita utilizando um algoritmo análogo ao algoritmo do vizinho mais próximo (*Nearest Neighbor*) (MAIN, DILLON, SHIU, 2001)

Resumidamente temos a relação apresentada por Finnie, Sun (2002): o algoritmo típico do vizinho mais próximo é mostrado na Equação 1:

$$\frac{\sum_{i=1}^{n} w_i \; x \; sim(f_i^I, f_i^R)}{\sum_{i=1}^{n} w_i}$$

Equação 1 - Algoritmo do vizinho mais próximo. Fonte: Finnie, Sun (2002)

3.3.2. Reuso

Nesta etapa considera-se o fator humano na escolha das soluções a serem apresentadas. As funções listadas como semelhantes na etapa anterior do processo são mostradas ao usuário para que sejam selecionadas (caso o usuário considere necessário) somente as de maior semelhança. Depois de selecionadas, o sistema buscará na base todas as soluções de todas estas funções e as classifica de acordo com o maior número de utilizações. O reuso de casos recuperados no mesmo contexto do novo caso se foca em dois aspectos (a) as diferenças entre o caso passado e o atual e (b) qual parte do caso pode ser utilizada para o novo caso.

O caso pode ser reutilizado através da cópia integral da solução, tentado aplica-la ao novo caso, ou ter uma etapa de adaptação aonde será feita análise e aplicação das partes cabíveis do caso recuperado ao novo. Ao solucionar as soluções manualmente, o usuário estará adicionando novas experiências na base que serão referência para os próximos projetos feitos utilizando o sistema, ou seja, adaptando a solução do caso recuperado ao problema vigente.

3.3.3. Revisão

Antes de definir quais serão as soluções finais, o usuário poderá revisar as soluções para avaliar se estas estão de acordo com as necessidades de projeto. Além do sistema de sugestão, também foi implementado um sistema simples de cadastro de conceitos que será usado para o cadastro de novos elementos quando o usuário não estiver satisfeito com as soluções criadas, alimentado a base com novas soluções. Para melhor visualização, são guardadas uma imagem e uma descrição do conceito para a composição visual da matriz morfológica.

3.3.4 Reuso

A etapa final é a decisão da realimentação do caso na base de dados para uso futuro. O caso poderá ser guardado na base ou exportado para um arquivo padrão contendo os conceitos utilizados nesta solução e o projeto em si. O sistema idealmente deveria contar com uma base de dados remota aonde os casos seriam imediatamente armazenados a pedido do usuário.

Os algoritmos implementados para cada uma das fases do RBC podem ser intercambiados para melhorar a performance, através do uso da interface *java* que descreve as funcionalidades do ciclo. Idealmente teriam vários algoritmos, no qual pudessem ser intercambiáveis desde que utilizassem o mesmo modelo de base de dados com as funções do módulo de modelo.

3.4 Visão

As funcionalidades do motor e da base de dados foram traduzidas na interface, com as devidas abstrações dos dados para visualização quando necessário. O trabalho do projeto conceitual requer necessita de representação gráfica para edição e modelamento das etapas de decomposição funcional e da matriz morfológica.

O usuário deve primeiro preencher a árvore de funções com as informações de classificação e seus relativos desdobramentos. A função geral não possui classificações, e terá somente os seus desdobramentos sendo necessário que a árvore de funções tenha pelo menos dois níveis de desdobramento para que possa ser considerada válida, ilustrado na Figura 4.

Cada uma das funções adicionada á arvore deve ser preenchida de forma a ter o seu estado alterado, o que é mostrado visualmente no formulário. Caso seja marcada na caixa de seleção como elementar, ela será guardada para uso na matriz morfológica na próxima etapa do processo, Figura 5.

A matriz morfológica é a última etapa do processo e consiste na seleção, para cada uma das funções elementares, das soluções que irão compor a matriz. São oferecidas duas possibilidades para a seleção dos conceitos, uma é a busca e utilização manual de um dos conceitos na base e outra que é o foco principal do trabalho que é a sugestão de conceitos para "solucionar" a função.

As funções selecionadas serão utilizadas para recuperar as suas soluções e compor a solução final para a função da matriz. A solução sugerida pode ser editada, adicionando ou removendo conceitos de acordo com a necessidade do usuário no editor da matriz morfológica.

Figura 4 – Classificações das funções

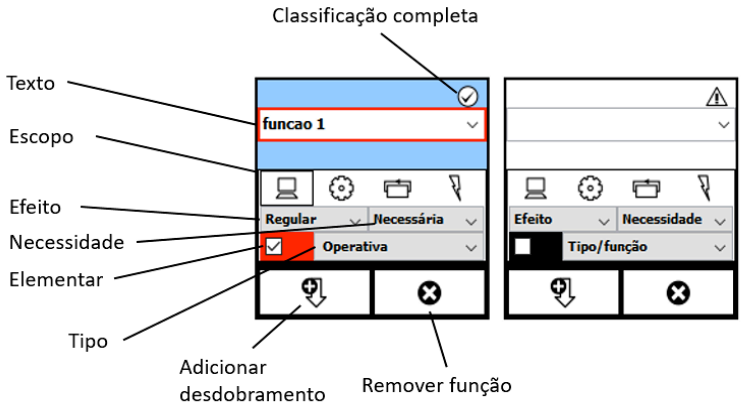
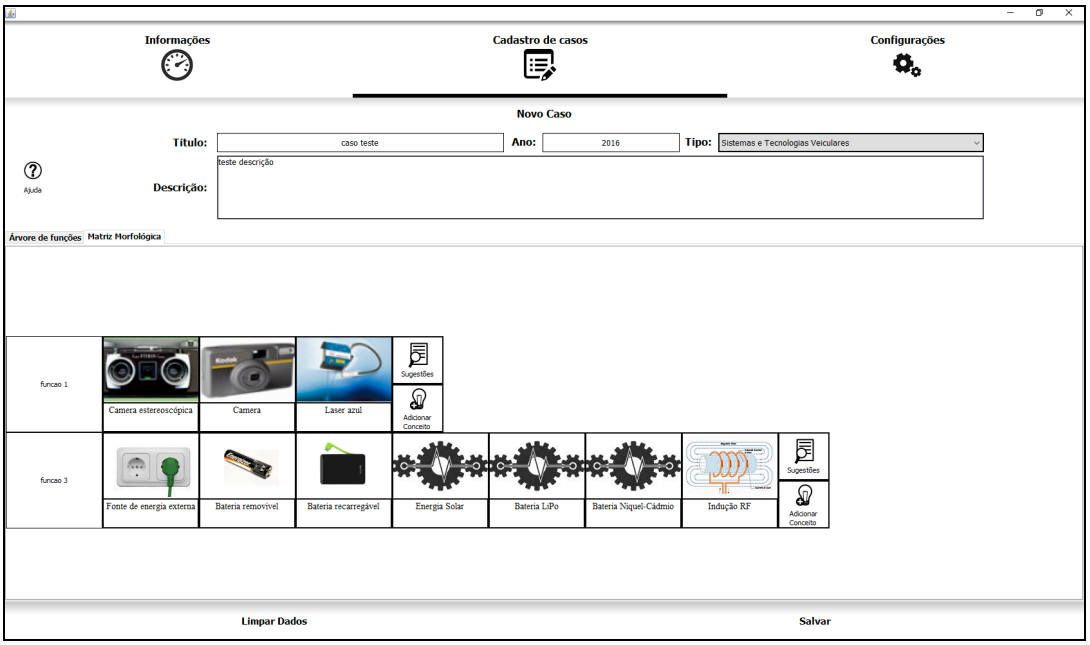


Figura 5 – Editor da matriz morfológica



O resultado final do desenvolvimento para esta funcionalidade pode ser descrito através dos exemplos a seguir. No primeiro, temos uma função bastante comum (utilizada em vários dos projetos) e que ilustra o funcionamento esperado idealmente para a sugestão de funções, Figura 6. Esta função, por ter um conjunto mais variado de casos utilizados tem uma grande quantidade de conceitos para utilização, bem como uma grande quantidade de variações, ou seja, possui um número mais variado de contextos de uso da mesma denominação “Alimentar dispositivo”, aumentando as chances de que soluções válidas sejam apresentadas.

4. AVALIAÇÃO

A avaliação da ferramenta proposta foi baseada nos conceitos validação e verificação descritos em FDA (2002). O planejamento de testes consistiu em unir as etapas de testes de componentes e do sistema em uma única etapa. A verificação e o teste de aceitação como a etapa de validação.

A etapa de validação foi executada com as equipes de alunos do curso de Graduação em Engenharia de Produção da Universidade de Brasília, todos cursando a disciplina de Engenharia de Produto no mesmo semestre. A turma foi dividida em doze equipes, sendo que seis equipes utilizaram o software e seis equipes não utilizaram.

Foi realizado um treinamento para as equipes sobre a utilização da ferramenta. Após a utilização foi apresentou-se um questionário com intuito de avaliar a experiência com a ferramenta. As questões foram elaboradas para cobrir a dificuldade e a eficiência do software em relação a cada uma das funcionalidades previstas. O questionário foi elaborado de acordo com as diretivas apresentadas em (DIX et al., 2012).

A verificação foi feita de forma a avaliar a qualidade do software na forma de testes funcionais para verificar a qualidade da implementação do software e também de forma a avaliar a eficiência do algoritmo como ferramenta de sugestão de conceitos.

Esperava-se que o crescimento da base interferisse positivamente nas sugestões feitas pelo *software* e que quanto mais projetos fossem inseridos melhor seria a qualidade das soluções geradas.

O questionário respondido pelos alunos mostrou que a ferramenta ainda apresenta alguns problemas de utilização, na maior parte relacionados ao modelo selecionado para troca dos dados e interface não intuitiva.

Era esperado que as funções já tivessem algum desempenho. Para três dos seis projetos, as soluções sugeridas foram relevantes na composição da matriz, pois algumas poucas funções não houve sugestão de princípios de solução, devido a carência de princípios de soluções cadastrados na base. Outros, três grupos foi considerado muito relevante, uma das equipes teve o seu projeto quase inteiramente composto de sugestões geradas pela ferramenta.

Observou ainda que comparado com as outras equipes que não utilizaram a ferramenta, as equipes que utilizaram a ferramenta geraram um maior número de princípios de solução e mais alternativas de concepção para os problemas propostos.

As avaliações realizadas mostraram vários dados importantes relacionados tanto à robustez do software quanto a sua utilidade como ferramenta de auxílio ao projeto. Estes dados servirão como ponto de apoio para futuros desenvolvimentos, tanto em novas funcionalidades quanto na melhoria da ferramenta. Ao final desta avaliação, a base de dados de projeto conta com 36 projetos cadastrados.

Figura 6 – Sugestão automatizada dos princípios de solução “Alimentar dispositivo”/Escopo: Energia, Elementar: Técnica (Esquerda) Sugestão automatizada dos princípios de solução “Alimentar dispositivo”/Escopo: Energia, Elementar: Transformação

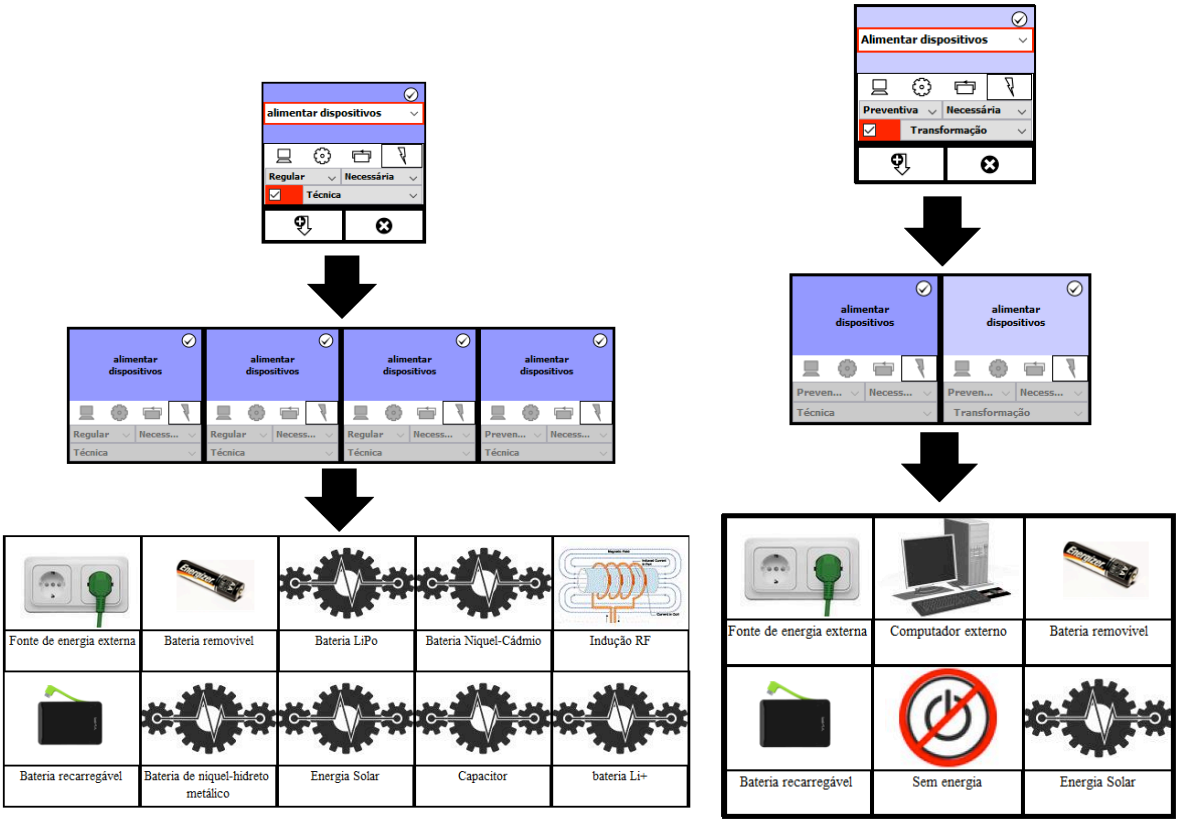


Figura 6 B– Sugestão automatizada dos princípios de solução “Alimentar dispositivo”/Escopo: Energia, Elementar: Transformação

7. CONSIDERAÇÕES FINAIS

O comportamento do sistema em relação ao crescimento da base de dados correspondeu ao esperado, com soluções mais completas surgindo com o aumento do número de casos. A possibilidade de alterar os algoritmos e implementar modelos com melhor funcionalidade ao projeto está presente na arquitetura do projeto. A etapa de testes foi executada com êxito, entre os pontos de melhora avaliados, encontram-se o melhor reuso das soluções através do emprego de técnicas mais avançadas de aprendizado de máquina. Gerou-se a proposição de uma ferramenta de apoio ao ensino da prática de projeto. O maior desafio enfrentado nesta pesquisa foi a busca por informações de projeto que pudessem servir para popular a base de casos, para este fim foram utilizados dados de projetos acadêmicos disponíveis.

REFERÊNCIAS

SELL, R.; TAMRE, M. Mechatronics System Design Process and Methodologies. 2016, (August).

WANG, L. et al. Collaborative conceptual design - State of the art and future trends. CAD Computer Aided Design. v34, n. 13, p.981-996, 2002.

CHANDRASEGARAN, S. et al. The evolution, challenges, and future of knowledge representation in product design systems. Computer-Aided Design. v.45, n.2, p.204-228, 2013.

ROZENFELD, H. et al. Gestão de Desenvolvimento de Produtos: uma referência para a melhoria do processo. São Paulo: Saraiva, 2006. 542 p.

RICHTER, M. M., AAMODT, A. Case-based reasoning foundations. The Knowledge Engineering Review, v. 20, n. 03, p. 203-207, 2005.

MANTARAS, R. et al. Retrieval, reuse, revision and retention in case-based reasoning. The Knowledge Engineering Review, v. 20, n.3, p. 215, 2006.

SOMMERVILLE, I. Software Engineering. Pearson: 2010. 700. p.

AAMODT, A., PLAZA. E. Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI communications, v. 7, n. 1, p. 39-59, 1994.

VEGTE, W. F., BREEMEN, E. J. J. Flowchart-Assisted Function Analysis of Products to Support Teaching of The Exact Sciences. International Conference on Engineering Design, ICED. 2009, 101–112.

ADAMSSON, N. Mechatronics Engineering: New requirements on cross-functional integration. Licentiate Thesi, Departament of Machine Design, Royal Institute of Technology. Stockholm Sweden, 2005.

MAIN, J., DILLON, T. S, SHIU, S. C. K. A tutorial on case based reasoning. Soft computing in case based reasoning. Springer: London, p.1-28, 2001.

FINNIE, G., SUN, Z. Similarity and metrics in case-based reasoning. International journal of intelligent systems, v. 17, n. 3, p. 273-287, 2002.

DIX, A., et al. Human–Computer Interaction. Prentice - Hall, 2012.

ZHENG, C. et al. Mechatronic Design Process: A Survey of Product Data Model. Procedia 24th CIRP Design Conference. v. 21, n.2, p.282–287. 2014.