Uma Ferramenta para Auxiliar na Execução de Processos de Software Modelados em SPEM

Douglas Montanha Giordano[1], **Eduardo Bruning**[2]

[1] Mestrado de Ciência da Computação – Universidade Federal de Santa Maria(UFSM)
Av. Roraima, 1000 – CEP 97105-900 – Santa Maria – RS

douglasmontanhagiordano@gmail.com, eduardo.bruning@gmail.com

Resumo. *Comumente antes do desenvolvimento de um software um processo de software deve ser definido para guiar todo o desenvolvimento do mesmo. Atualmente existem vários tipos de software que oferecem suporte as mais diferentes etapas de um processo de software. A partir de um mapeamento sistemático podemos visualizar em uma escala mais abrangente a necessidade de um software que apoie a execução do processo de software, em um nível que ofereça também um suporte a gestão de todos seus artefatos gerados em cada etapa de sua execução. O objetivo deste trabalho é oferecer uma ferramenta de apoio a execução de processos modelados em SPEM. Para isso construímos uma ferramenta que a partir de um documento no formato SPEM pode gerar um diagrama que possibilita o acompanhamento da execução de cada etapa do processo e também o arquivamento dos artefatos gerados em cada etapa.*

1. Introdução

2. Fundamentação Teórica e Tecnológica

3. Trabalhos Relacionados

Para investigar os trabalhos relacionados a esta pesquisa foi utilizado um mapeamento sistemático da literatura. Ele consiste em três etapas, a definição do protocolo do mapeamento, a execução do protocolo e a análise dos resultados.

3.1. Protocolo do mapeamento

O objetivo do protocolo é estabelecer o estado da arte das ferramentas que dão apoio a execução de processos baseados em SPEM. Baseado nisso, foi definido uma questão de pesquisa para investigar nos trabalhos selecionado, sendo ela:

- Qual a função e objetivo das ferramentas?

O mecanismo de busca escolhido para realizar as buscas foi o Google Scholar [1], levando em consideração que ele indexa a maior parte das bases de dados. A *string* de busca utilizada foi "software process"+"tool" OR "tools" OR "CASE" + "software & systems process engineering metamodel" OR "SPEM". Para filtrar mais a busca, definimos os seguintes critérios de inclusão.

- Artigos completos publicados de 2012 até 2017 e com no mínimo 6 (seis) páginas;
- Artigos que abordam sobre ferramentas de suporte ao SPEM.

[1] Google Scholar http://scholar.google.com

Para retirar alguns trabalhos que não são interessantes para esta pesquisa definimos dois critérios de exclusão.

- Artigos que sejam outros mapeamentos ou revisões sistemáticas;
- Artigos repetidos ou versões resumidas de trabalhos completos.

De acordo com os trabalhos retornados, o modo de classificação definido foi pelo ano e o veículo de publicação.

3.2. Resultado

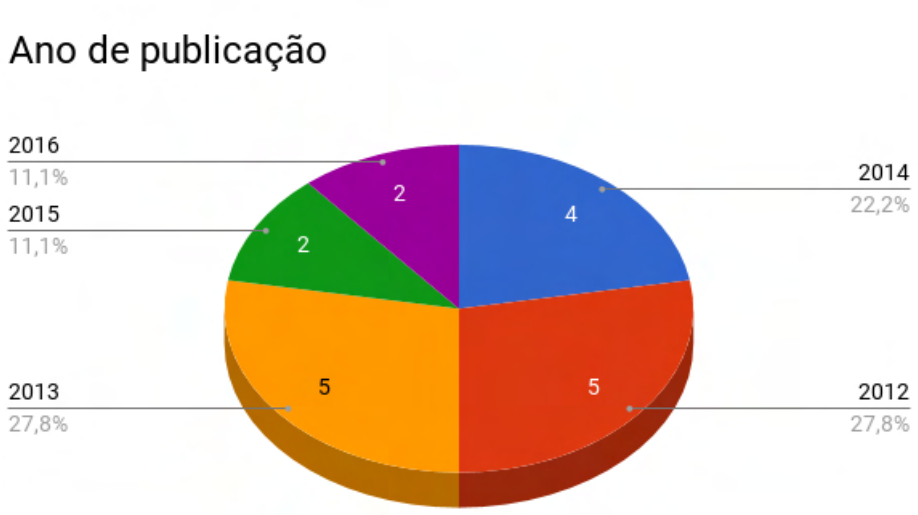
A execução do protocolo retornou 2060 trabalhos, com a aplicação dos critérios de inclusão este número reduziu para 726 e após aplicar os critérios de exclusão 19 trabalhos foram selecionados para leitura e análise mais aprofundada. Dentre os trabalhos selecionados nenhum deles apresentou ferramentas para apoiar a execução de processos, servindo de suporte para guiar e gerenciar as etapas do processo quando ele está sendo executado.

Para dar um panorama geral dos trabalhos que encontrados, são exibidos de acordo com sua classificação, pelos anos de publicação e seus veículos de pesquisa.

3.2.1. Classificação dos resultados

Para visualizar a quantidade de publicações pelos anos, é exibido no gráfico abaixo os trabalhos selecionados de acordo com seu ano de publicação. A grande maioria está entre os anos de 2012 a 2014, sendo a concentração maior no ano de 2012, seguido de 2013 e 2014. Os anos de 2015 e 2016 são equivalentes em relação a quantidade de publicações. Desta forma, nos últimos anos as publicações sobre este assunto apresentam uma queda de quantidade de publicações. Isso é visível apenas observando o gráfico, sendo que não existem trabalhos selecionados do ano de 2017.

Figura 1. Gráfico do ano das publicações



Com a intenção de identificar onde ocorrem a maior parte das publicações dos trabalhos que foram selecionados, eles são classificados de acordo com seu veículo de publicação. Eles são exibidos no gráfico abaixo. A maior quantidade está na fatia chamada de outros, que engloba trabalhos de bases de universidades e de veículos de menor expressão. Esse agrupamento foi realizado para fins de melhorar a exibição dos dados. Dentre os veículos mais conhecidos a IEEE lidera com 4 trabalhos (ou 21,1%) seguida pela Elsevier com 3 trabalhos (ou 15,8%).

Figura 2. Gráfico do veículos das publicações

Após uma exibição geral dos trabalhos, a seguir são exibidas as respostas dos trabalhos para as questões de pesquisa definidas no protocolo.

3.2.2. Qual a função e objetivo das ferramentas?

Kuhrmann et al. (2043) apresentam a ferramenta Process Enactment Tool Framework (PET), que tem como objetivo transformar determinados modelos de processo formal em um formato que as ferramentas de projetos de processos possam interpretar, ou seja, O PET é um instrumento para importar processos com base em um metamodelo e fornecer exportações para um ambiente de projeto específico [Kuhrmann et al. 2014]. Ellner et al. (2012) apresentam uma cadeia de ferramentas integrada baseada em eSPEM e Eclipse. As suportam modelos de processo na modelagem de processos de software baseados em eSPEM e também orienta e apoia a equipe de projeto em trabalhar de acordo com o processo de maneira sensível ao contexto. Elas automatizam o trabalho repetitivo e pesado como verificação de conformidade do processo ou acompanhamento de progresso [Ellner et al. 2012]. Já Hurtado et al.(2013) propõem uma abordagem orientada por modelo para especificação e configuração de linhas de processo de software [Hurtado et al. 2013]. Entrando nas metodologias ágeis Abad, Alipour e Ramsin (2012) propõem um *plug-in* para o EPFC, para permitir aos engenheiros de métodos construir metodologias ágeis através de uma abordagem de PME baseada em montagem. O *plug-in*

fornece facilidades para a especificação das características de um determinado projeto, seleção de componentes de processo ágeis adequados do repositório de métodos e montagem final dos pedaços de método selecionados [Abad et al. 2012].

Três trabalhos abordam sobre implantação de melhorias e avaliação em processos de softwares como Portela et al. (2014) apresentam um conjunto de ferramentas de suporte para a implementação de processos modelados no Metamodelo de Engenharia de Processo de Software (SPEM). Este conjunto de ferramentas, é chamada Spider-PE e tem como objetivo auxiliar as organizações de software na implementação do Modelo de Maturidade de Capacidades para Desenvolvimento (CMMI-DEV) e do Modelo de Referência de Melhoria de Processos de Software para Software (MR-MPS- SW) [Portela et al. 2014]. Hurtado, Bastarrica e Bergel (2013) apresentam uma ferramenta que permite analisar a qualidade dos modelos de processos de software SPEM 2.0. A ferramenta chamada Avispa, identifica uma série de padrões de erro e os destaca em planos diferentes. Uma descrição detalhada dos internos da Avispa é fornecida para mostrar sua estrutura e seus mecanismos de extensibilidade. Além disso, os autores apresentam um mecanismo interativo para definir novos *scripts* de análise e implementar novos padrões e planos [Hurtado Alegría et al. 2013a]. Já se tratando de avaliação, Gazel, Sezer e Tarhan (2012) apresentam uma ferramenta de avaliação de processo de software baseada em ontologia para coleta de dados da avaliação de processos e acompanhar a conformidade dos processos de software com o CMMI como modelo de referência do processo [Gazel et al. 2012].

Muitos trabalhos selecionados abordam sobre adaptação de processos, definindo algumas variáveis para adaptar da melhor maneira o processo para determinado escopo, dentre esses trabalhos, Silvestre, Bastarrica e Ochoa 2014, apresentam uma ferramenta baseada em modelo que permite aos engenheiros de processos gerar automaticamente regras de transformação de adaptação usando uma interface gráfica de usuário [Silvestre et al. 2014]. Hurtado et al (2013) mostram uma abordagem baseada em modelo para a adaptação de processos de software gerando automaticamente processos específicos do projeto com base no processo organizacional e nos contextos do projeto. A proposta é aplicada pelos autores para adaptar o processo de engenharia de requisitos de uma empresa chilena [Hurtado Alegría et al. 2013b]. Casare et al (2016) expõem Medusa, uma abordagem para adaptar processos de desenvolvimento de acordo com as necessidades do projeto. Ela se baseia em técnicas e métodos de engenharia de linha de produtos, leva em consideração as semelhanças e diferenças do processo, bem como fragmentos reutilizáveis [Casare et al. 2016]. Santos et al. (2015) relatam uma técnica que usa a mineração de processos para descobrir elementos do processo de software que são candidatos a adaptação. A abordagem analisa os *logs* de execução de várias instâncias de processo que compartilham um processo padrão comum [Santos et al. 2015]. Chaghrouchni, Kabbaj e Bakkoury (2014) apresentam uma nova abordagem para a evolução do processo, com base na adaptação dinâmica. Ao identificar os desvios do processo a adaptação dinâmica é usada para decidir a nova atividade ou conjunto de atividades a serem lançadas para reproduzir entradas e saídas solicitadas e para lidar com o impacto de desvio [Chaghrouchni et al. 2014].

Rouillé et al. (2013) propõe uma abordagem para resolução de variabilidades de processos em Linhas de Processos de Software durante a execução do processo,

sendo que em alguns casos apenas uma parte das variabilidades são resolvidas e o restante é resolvida posteriormente [Rouillé et al. 2013]. Em outro trabalho Rouillé et al. (2012) relatam uma abordagem para definição de uma Linha de Processo de Software e a derivação automática de um processo a partir desta SPL de acordo com os requisitos de um determinado projeto. A variabilidade é gerenciada separadamente do modelo de processo utilizando uma linguagem de variabilidade comum [Rouillé et al. 2012]. Simmonds et al. (2012) apresentam uma combinação de notações e ferramentas para formalizar modelos de processos de software incluindo sua variabilidade, utilizando a ferramenta Modisco/AMW para garantir que somente a variabilidade razoável seja especificada [Simmonds et al. 2012]. Blum, Simmonds e Bastarrica (2015) propõem o algoritmo V que usa dois processos limiares para configurar uma Linha de Processo de Software. Relações muito frequentes são usadas para construir o processo base, as relações variáveis definem a variabilidade do processo e as relações raras são descartadas como ruídos [Blum et al. 2015]. Ayora et al. (2016) derivaram um conjunto de padrões de mudança para famílias de processos de construções de linguagem específicas de variabilidade identificadas na literatura, com objetivo de facilitar o gerenciamento de variabilidade nas famílias de processos, garantindo a correção familiar do processo e reduzir o esforço necessário para tais fins [Ayora et al. 2016].

4. Desenvolvimento da Ferramenta

Nesta seção é apresentado como foi construída a ferramenta para auxiliar na execução do processos. Na subseção 4.1 é descrito o processo de desenvolvimento e o que foi desenvolvido. Na subseção 4.2 é descrito por meio de diagramas UML como está a estrutura de implementação do software.

4.1. Processo

O desenvolvimento do software foi guiado por algumas atividades propostas pelo SCRUM. Como a equipe atual é constituída de dois desenvolvedores, apenas algumas atividades foram seguidas. Abaixo uma lista das atividades propostas pelo SCRUM que utilizamos.

- Planejamento do Sprint
- Revisão do Sprint
- Execução do Sprint

Foi utilizado o conceito de Sprint e Sprint Backlog, no qual nos guiou em todo desenvolvimento. Definimos o tempo de cada Sprint em 2 semanas, no qual foi possível executar 2 Aprints até o desenvolvimento deste artigo. Na tabela 1 é apresentado as histórias de usuário contidas no Sprint Backlog e na tabela 2 e 3 é apresentado as histórias de usuário executadas em cada Sprint.

4.2. Implementação

O software foi criado utilizando a linguagem de programação JAVA com seus recursos da plataforma WEB. Abaixo é apresentado as principais tecnologias utilizadas para construção do software.

- JSF
- Primefaces

Tabela 1. Sprint Backlog

ID	História de Usuário	Prioridade
1	Eu como usuário desejo poder selecionar um arquivo no formato SPEM e ter como saída uma forma de acompanhar a execução do processo.	1
2	Eu como usuário desejo pode finalizar cada atividade do processo como intuito de acompanhar o fluxo das atividades.	2
3	Eu como usuário desejo poder anexar arquivos nos artefatos gerados em cada uma das etapas de execução do meu processo.	3
4	Eu como usuário desejo poder a partir de um arquivo SPEM gerar um arquivo contendo o software para execução do processo	4

Tabela 2. Sprint 1

ID	História de Usuário	Prioridade
1	Eu como usuário desejo poder selecionar um arquivo no formato SPEM e ter como saída uma forma de acompanhar a execução do processo.	1

Tabela 3. Sprint 2

ID	História de Usuário	Prioridade
2	Eu como usuário desejo pode finalizar cada atividade do processo como intuito de acompanhar o fluxo das atividades.	2
3	Eu como usuário desejo poder anexar arquivos nos artefatos gerados em cada uma das etapas de execução do meu processo.	3

- Bootstrap
- XStream
- Maven

O artigo não entrará em detalhes de implementação, apresentando apenas a relação entre as classes e a arquitetura do software.

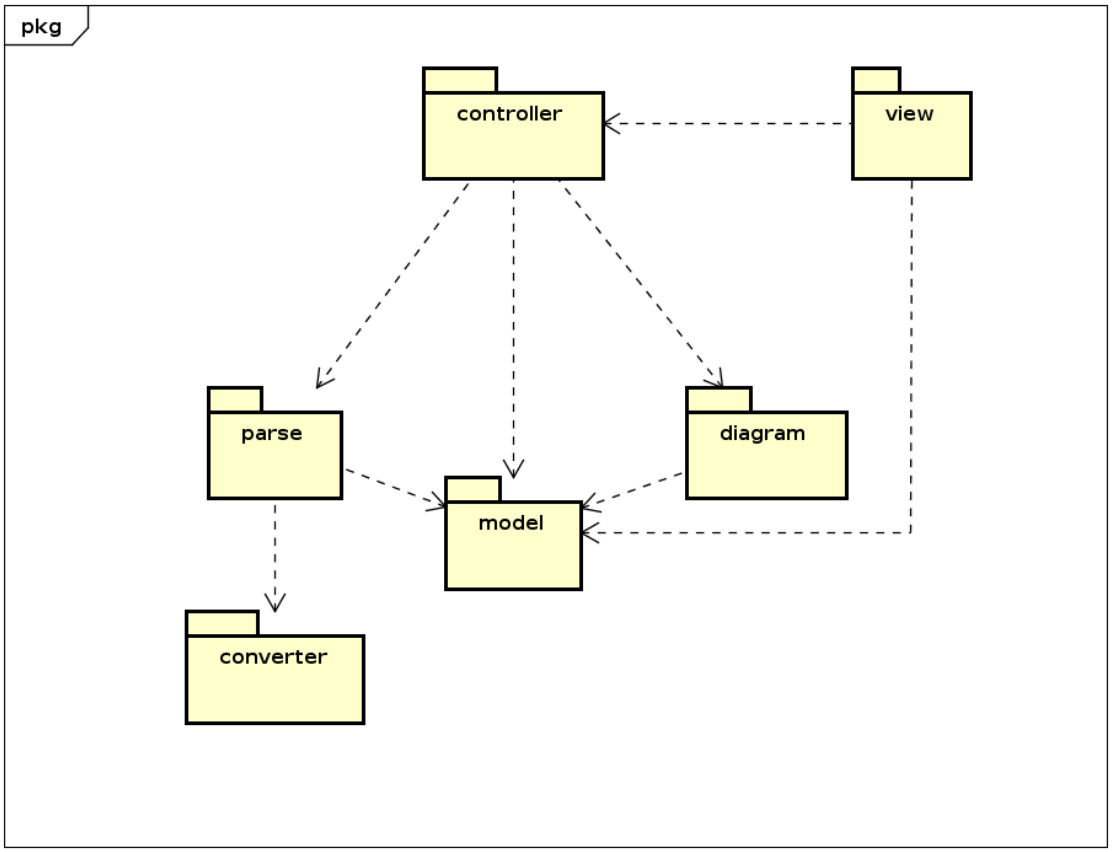
Na subseção 4.2.1 é apresentado a arquitetura do software, descrevendo o objetivo de cada pacote de software. Na subseção 4.2.2 são apresentadas as relações entre as classes do software, descrevendo de forma simplificada a funcionalidade das principais classes envolvidas na geração do diagrama de classes.

4.2.1. Arquitetura

A figura 3 apresenta os principais pacotes envolvidos na geração da estrutura para controlar a execução de processos. Os pacotes model e diagram estão envolvidos com as regras de negócio do software. Os pacotes parse e converter estão envolvidos com a conversão do modelo SPEM representado em XML para objetos no Java em tempo de execução do

software. O pacote controler tem como responsabilidade utilizar todos os demais pacotes oferecendo uma interface para o pacote view executar requisições do usuário.

Figura 3. Arquitetura



4.2.2. Diagrama de Classes

Na figura 4 é apresentado o diagrama de classes, constituído das principais classes envolvidas na construção do software.

5. Resultados

A partir da implementação do software foi possível construir uma ferramenta de auxilio a execução de processos. O software apesar de simples possui uma estrutura robusta, possibilitada pela componentização que a linguagem Java oferece, podendo ter suas funcionalidades aumentadas ao longo do tempo.

Na figura 5 é apresentado a estrutura no qual o usuário seleciona o arquivo SPEM. O arquivo pode ser gerado a partir de um processo modelado na ferramenta EPF Composer.

Após o arquivo ser selecionado o usuário necessita apenas clicar no botão enviar. Com o arquivo no formato SPEM enviado para o servidor, a ferramenta vai gerar um diagrama como mostrado na figura 6.

Figura 4. Diagrama de Classes

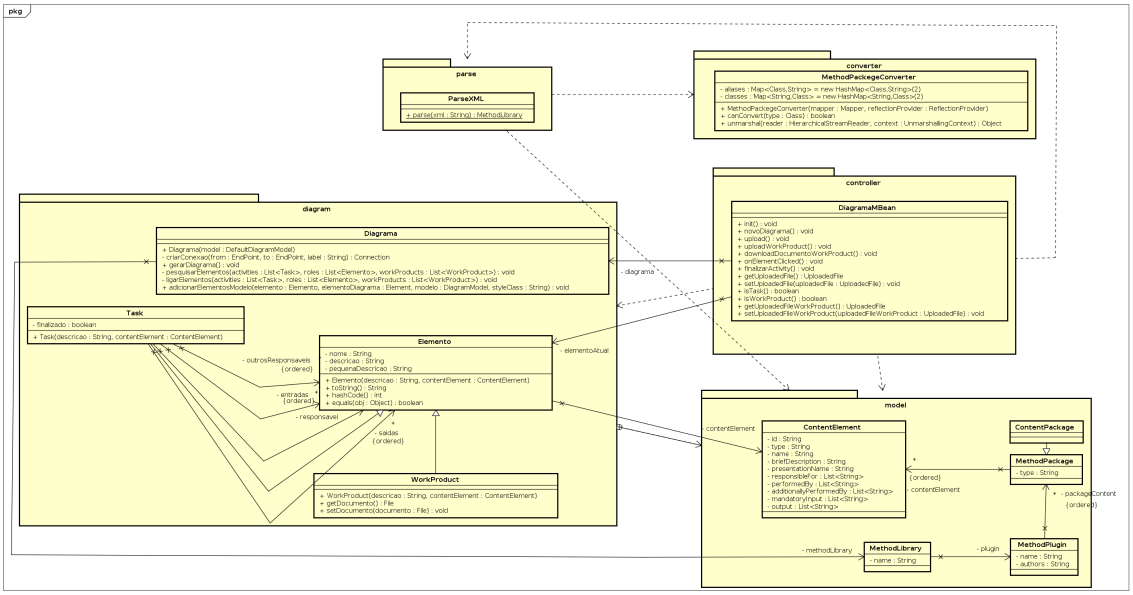
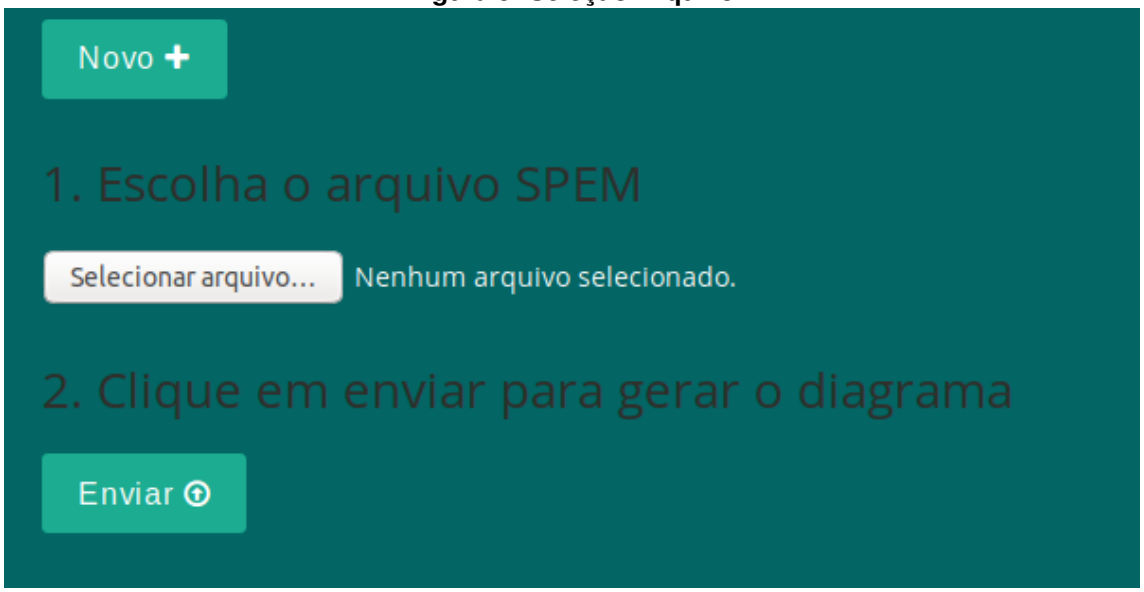


Figura 5. Seleção Arquivo



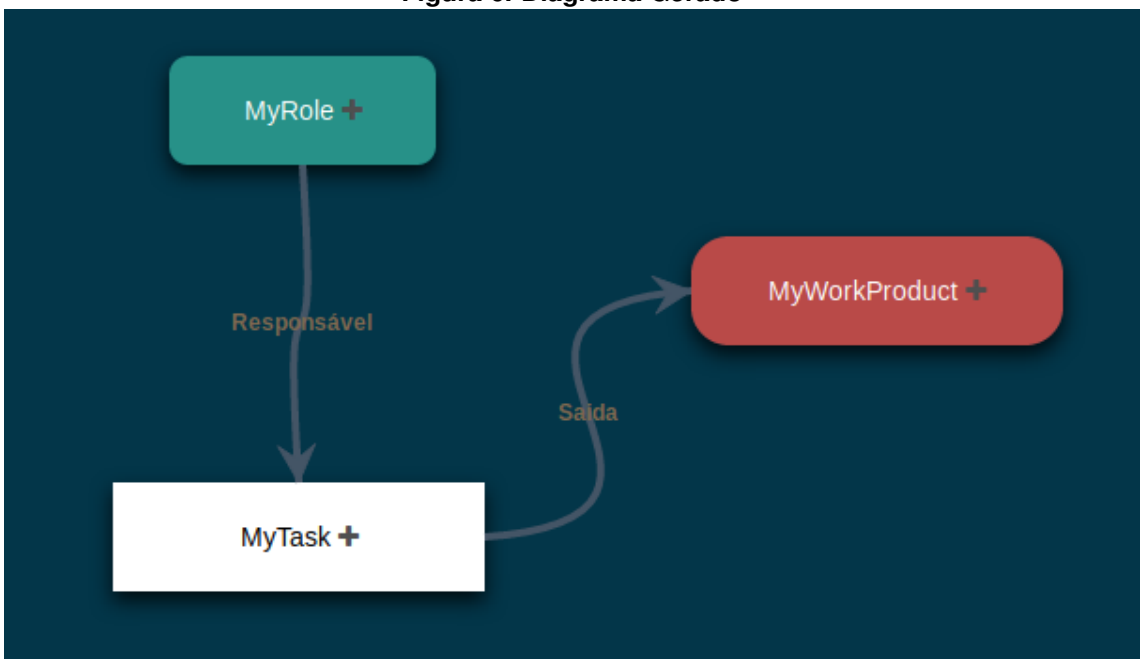
Este diagrama pode ser utilizado para controlar o fluxo das atividades, sendo possível finalizar cada atividade completada. Além disso é possível anexar arquivos a cada artefato da atividade, podendo ser posteriormente acessado novamente.

6. Conclusões

Referências

Abad, Z. S. H., Alipour, A., and Ramsin, R. (2012). Enhancing tool support for situational engineering of agile methodologies in eclipse. In *SERA (selected papers)*, pages 141–152. Springer.

Figura 6. Diagrama Gerado



Ayora, C., Torres, V., de la Vara, J. L., and Pelechano, V. (2016). Variability management in process families through change patterns. *Information and Software Technology*, 74:86–104.

Blum, F. R., Simmonds, J., and Bastarrica, M. C. (2015). Software process line discovery. In *Proceedings of the 2015 International Conference on Software and System Process*, pages 127–136. ACM.

Casare, S., Ziadi, T., Brandão, A. A. F., and Guessoum, Z. (2016). Meduse: an approach for tailoring software development process. In *Engineering of Complex Computer Systems (ICECCS), 2016 21st International Conference on*, pages 197–200. IEEE.

Chaghrouchni, T., Kabbaj, I. M., and Bakkoury, Z. (2014). Towards dynamic adaptation of the software process. In *Intelligent Systems: Theories and Applications (SITA-14), 2014 9th International Conference on*, pages 1–7. IEEE.

Ellner, R., Al-Hilank, S., Jung, M., Kips, D., and Philippsen, M. (2012). An integrated tool chain for software process modeling and execution. In *Proceedings of the European Conference on Modelling Foundations and Applications*, pages 73–82.

Gazel, S., Sezer, E. A., and Tarhan, A. (2012). An ontology based infrastructure to support cmmi-based software process assessment. *Gazi University Journal of Science*, 25(1):155–164.

Hurtado, J. A., Bastarrica, M. C., Ochoa, S. F., and Simmonds, J. (2013). Mde software process lines in small companies. *Journal of Systems and Software*, 86(5):1153–1171.

Hurtado Alegría, J. A., Bastarrica, M. C., and Bergel, A. (2013a). Avispa: a tool for analyzing software process models. *Journal of Software: Evolution and Process*, 26(4):434–450.

Hurtado Alegría, J. A., Bastarrica, M. C., Quispe, A., and Ochoa, S. F. (2013b). Mde-based process tailoring strategy. *Journal of Software: Evolution and Process*, 26(4):386–403.

Kuhrmann, M., Kalus, G., and Then, M. (2014). The process enactment tool framework—transformation of software process models to prepare enactment. *Science of Computer Programming*, 79:172–188.

Portela, C., Vasconcelos, A., Oliveira, S., Silva, A., and Elder, S. (2014). Spider-pe: A set of support tools to software process enactment. In *Proceedings of the 9th International Conference on Software Engineering Advances*.

Rouillé, E., Combemale, B., Barais, O., Touzet, D., and Jézéquel, J.-M. (2012). Leveraging cvl to manage variability in software process lines. In *Software Engineering Conference (APSEC), 2012 19th Asia-Pacific*, volume 1, pages 148–157. IEEE.

Rouillé, E., Combemale, B., Barais, O., Touzet, D., and Jézéquel, J.-M. (2013). Integrating software process reuse and automation. In *Software Engineering Conference (APSEC), 2013 20th Asia-Pacific*, volume 1, pages 380–387. IEEE.

Santos, R., Oliveira, T. C., et al. (2015). Mining software development process variations. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 1657–1660. ACM.

Silvestre, L., Bastarrica, M. C., and Ochoa, S. F. (2014). A model-based tool for generating software process model tailoring transformations. In *Model-Driven Engineering and Software Development (MODELSWARD), 2014 2nd International Conference on*, pages 533–540. IEEE.

Simmonds, J., Bastarrica, M. C., Silvestre, L., and Quispe, A. (2012). Modeling variability in software process models. *Computer Science Department, Universidad de Chile, Santiago, Chile*.