Identificação Automatizada de Classes Utilizando Processamento de Linguagem Natural

Alternative Title: Automated Identification of Classes Using Natural Language Processing

Daniel Antonio Conte
Universidade Alto Vale do Rio do Peixe
conte@uniarp.edu.br

Jean Carlo Rossa Hauck
Universidade Federal de Santa Catarina
jean.hauck@ufsc.br

RESUMO

A documentação correta e suficiente de um sistema de informação tende a facilitar a sua manutenção. Nesse sentido, o diagrama de classes é um importante artefato da UML para o projeto de um sistema de informação orientado a objetos. A elaboração desse tipo de diagrama, no entanto, muitas vezes é onerosa e complexa por envolver diversos papéis e um profundo conhecimento da área de domínio. Algumas experiências têm demonstrado a viabilidade da geração automatizada do diagrama de classes, sendo que a análise baseada em descrições é uma das técnicas possíveis. O presente trabalho busca aplicar o processamento de linguagem natural para apoiar a elaboração do diagrama de classes. Para isso, foi modelado e implementado um protótipo para validação da proposta. A avaliação inicial do uso da ferramenta foi considerada satisfatória.

Palavras-Chave

Processamento de linguagem natural, Diagrama de classes, Orientação a objetos

ABSTRACT

The correct and sufficient documentation of an information system tends to facilitate its maintenance. In this sense, the class diagram is an important UML artifact for the design of an object-oriented information system. The development of this type of diagram, however, is often costly and complex because it involves different roles and a thorough knowledge of the domain area. Some experiences have demonstrated the feasibility of the automated generation of class diagrams, and the analysis based on descriptions is one of the possible techniques. This study aims to apply natural language processing to support the development of the class diagram. For this, an application prototype is modeled and implemented in order to validate the proposal. The initial evaluation of the use of the tool was considered satisfactory.

Categories and Subject Descriptors

D.3.2 [**UML**]

General Terms

Documentation, Design

Keywords

Natural language processing, Class diagram, Object Orientation

SBSI 2016, May 17–20, 2016, Florianópolis, Santa Catarina, Brazil.

1. INTRODUÇÃO

Com o intuito de mitigar os problemas de manutenção de software, como código incompreensível e alto acoplamento em sistemas complexos, os projetos orientados a objetos, e suas formas de documentação, são amplamente aplicados no desenvolvimento de sistemas de informação [16]. Nesse sentido, a UML (*Unified Modeling Language*) é uma notação utilizada para especificar sistemas de software, dentre outras aplicações [10]. Assim, o diagrama de classes é um dos diagramas estruturais da UML, utilizado para descrever as classes, com seus respectivos atributos e operações, bem como suas interfaces e associações [5]. A modelagem do diagrama de classes é componente fundamental para todo o processo de desenvolvimento de sistemas de informação orientados a objetos [11].

No desenvolvimento de sistemas de informação sob o paradigma da orientação a objetos, um objeto pode ser considerado uma representação de uma entidade, ou conceito, que realiza interações e possui responsabilidades [15]. Nesse contexto, as classes são, então, abstrações desses objetos, onde objetos que possuem: características, responsabilidades e relações semelhantes podem pertencer a uma mesma classe, sendo instâncias dessa classe [15]. O diagrama de classes especifica, então, as abstrações, suas responsabilidades e interações [10].

As classes possuem alguns elementos típicos, tais como: atributos, operações e relacionamentos. Os atributos de uma classe nomeiam as características comuns aos objetos daquela classe [5]. As operações caracterizam as ações ou comportamento da classe. Essas ações definem como os objetos podem atuar. Comumente, as operações são o meio pelo qual os objetos interagem. Assim como o atributo, uma operação é compartilhada por todos os objetos de uma mesma classe [10]. O diagrama de classes também representa as relações existentes entre as classes, quando, por exemplo, uma classe necessita conhecer determinados atributos ou enviar determinadas mensagens a outra classe [16]. A base para a elaboração de um diagrama de classes é, em geral, a documentação dos requisitos de software [16], [17].

Os requisitos de um sistema de informação, entretanto, são normalmente escritos em linguagem natural [17] e a revisão de documentos textuais, para a elaboração do diagrama de classes, tende a ser onerosa e demandar atenção do analista ou projetista [3]. Assim, avaliar previamente a documentação de requisitos de maneira automática é um recurso que pode auxiliar o analista na realização de suas tarefas [6].

1.1 Processamento de Linguagem Natural

Nesse sentido, o processamento de linguagem natural propõe o uso de técnicas para análise sintática e automática de textos [12]. A interpretação semântica escrita, é definida por [12] (p. 769) como “o processo de extrair o significado de uma expressão vocal

como uma expressão em alguma linguagem de representação". Tal processo segue uma série de passos bem definidos: primeiramente é feita a chamada análise morfológica, em seguida é realizada a análise sintática, seguida da análise semântica.

Assim, os substantivos e orações equivalentes são tipicamente classificados como classes candidatas [1]. Quando dois ou mais substantivos são encontrados em sequência no texto o primeiro é classificado como classe candidata enquanto os seguintes são classificados como atributos da referida classe [6]. Já a verificação os verbos e orações equivalentes compreendem um conjunto de prováveis operações das classes [16]. Da mesma forma, quando a expressão "é um/uma" está localizada entre duas classes (A e B respectivamente), pode ser apontada uma especificação onde A herda ou implementa o comportamento e estrutura de B [4]. Se entre duas classes há um advérbio, este indica uma possível associação entre as mesmas [9].

Os adjetivos podem assumir o caráter de atributos, contanto que haja uma classe candidata no mesmo parágrafo, em caso positivo os atributos são característicos da referida entidade.

Uma vez que a língua se desenvolve em função do uso e não da lógica, a análise semântica via processamento de linguagem natural nunca será considerada como plena e precisa, o que não impede que se aproxime da verdade [12].

Utilizar essas técnicas de processamento de linguagem natural para identificar, verificar e validar componentes do diagrama de classes agrega valor especialmente pela redução do esforço dedicado [4]. As experiências de utilização do processamento de linguagem natural para identificar as classes de sistemas orientados a objetos a partir da especificação de requisitos têm demonstrado que seu uso é factível. Os estudos e resultados em língua inglesa são bem estruturados e importantes [4], [6]. Em língua portuguesa as iniciativas são escassas e apesar da assertividade, tem sido mostrado um excesso de falsos-positivos [3], [14].

Assim, este trabalho propõe o desenvolvimento de uma ferramenta que apoie o analista ou arquiteto na elaboração de diagramas de classes a partir da descrição dos requisitos de um sistema de informação, e que possa apresentar assertividade similar à elaboração manual, reduzindo o esforço necessário do profissional.

O presente artigo está organizado da seguinte forma: na seção 2 a abordagem metodológica é apresentada, seguida da seção 3 onde são apresentados os trabalhos correlatos, na seção 4 discorre-se sobre o desenvolvimento do protótipo, na seção 5 são apresentados os resultados dos testes iniciais, por fim as considerações finais são apontadas na sessão 6.

2. ABORDAGEM METODOLÓGICA

Como já apresentado, o presente trabalho tem como objetivo principal apoiar a elaboração do diagrama de classes por meio do processamento de linguagem natural de textos de descrições em alto nível de requisitos funcionais e/ou não-funcionais, de forma a verificar sua assertividade quando comparado à análise e elaboração manual do diagrama de classes.

Para atingir este objetivo, inicialmente é realizado um estudo da literatura acerca do tema, envolvendo tanto o conhecimento da notação UML, especificamente do diagrama de classes, quanto do processamento de linguagem natural.

Após o aprofundamento do conteúdo, é realizado o levantamento dos trabalhos correlatos, por meio de um mapeamento da literatura sobre o tema, buscando identificar os trabalhos atualmente sendo realizados que têm buscado objetivos e métodos similares a este trabalho.

Com base no conhecimento adquirido, é então realizada a modelagem e o desenvolvimento de um protótipo de ferramenta para realizar o processamento em linguagem natural para identificação de classes. Uma limitação importante consiste no fato de que somente classes que representam entidades de negócio devem ser identificadas de forma automática, não sendo geradas classes de abstração pura ou similares.

O protótipo é desenvolvido e uma avaliação inicial do seu uso é realizada para avaliar se os resultados obtidos pelo processamento automatizado são similares aos resultados obtidos na análise manual dos requisitos para elaboração do diagrama de classes.

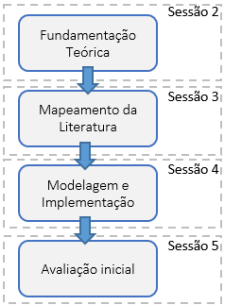

Figura 01 – Abordagem Metodológica

A Figura 01 apresenta a abordagem metodológica adotada e a sua relação com as sessões deste artigo.

3. TRABALHOS CORRELATOS

Dada a importância do diagrama de classes para o desenvolvimento de sistemas orientados a objetos, diversas iniciativas têm sido realizadas de forma a automatizar, mesmo que parcialmente a sua elaboração. Assim, em 2015 foi realizada uma revisão da literatura, utilizando o Google Scholar (https://scholar.google.com.br) como ferramenta de busca, aplicando-se os termos: "diagrama de classes", "orientação a objetos", "processamento de linguagem natural", "identificação", incluindo suas traduções em inglês. A partir da execução da busca, dentre os resultados iniciais obtidos, foram considerados nos critérios de inclusão somente os trabalhos que apresentassem o processamento em linguagem natural de textos de requisitos e a geração automatizada de um diagrama ou ao menos a descoberta das classes, atributos e operações. Como resultado, foram encontrados 5 trabalhos que são apresentados na sequência.

Em [6], o diagrama de classes é construído basicamente verificando as classes gramaticais das palavras. O número restritivo de regras para verificação da categoria dos elementos elencou certa quantidade de falsos positivos, apesar de todas as classes apontadas corretamente.

O trabalho [14] executa um desenvolvimento completo, construindo diagramas de maneira assertiva, usando vários mecanismos de analise sintática e etiquetadores de texto. Há, no entanto, um número limitado de validações. A presente pesquisa considera uma maior quantidade de mecanismos de validação, no intuito de reduzir os falsos positivos. A incidência dos mesmos pode afetar mais a revisão do analista do que a omissão de algum elemento não identificado.

Em [3] a identificação de requisitos, apresentada em casos de uso, utiliza o diagrama de classes como parte do processo. As heurísticas definidas para as classes partem da relação entre substantivos e outros elementos textuais. Os resultados são

alcançados quando o texto é produzido num formato pré-estabelecido.

Em [9] são aplicados vários filtros para evitar o excesso de informação. O protótipo desenvolvido tem resultados relevantes para a modelagem do diagrama entidade-relacionamento.

Em [4] os autores reúnem várias técnicas, baseadas em análise textual, para identificação de classes. A análise sintática é baseada na verificação da estrutura das orações. A integração de diversos modelos produziu resultados bastante satisfatórios, com um alto índice de acerto, onde além da identificação o modelo era desenhado. O presente trabalho vale-se do mesmo caminho, costurando vários meios para apontar os elementos do diagrama de classes, as técnicas de análise sintática são semelhantes. A principal diferença é o idioma, já que neste trabalho realizado os insumos textuais são em língua portuguesa..

4. DESENVOLVIMENTO DO PROTÓTIPO

Para construção de um aplicativo que processe linguagem natural é necessário um léxico, um vocabulário apresenta a classe gramatical de uma determinada palavra [8]. O léxico pode abranger várias áreas do conhecimento, bem como uma somente [7]. Um léxico que contemple toda a língua portuguesa não está disponível de maneira aberta. Assim, para o protótipo desenvolvido foi utilizado o corpus Floresta [13], disponível em http://www.linguateca.pt/floresta/corpus.html. Tal biblioteca é composta de 1.600.000 palavras classificadas, retiradas de textos do Jornal Folha de São Paulo.

O protótipo foi implementado em Python, disponível em http://wiki.python.org.br/. Devido a facilidade de manipulação de textos e a compatibilidade com o corpus Floresta. Assim, o protótipo importa o módulo *tagged_words* do corpus Floresta, através do módulo NLTK (*Natural Language Toolkit*), disponível em http://www.nltk.org.

Com base na literatura, foram então definidas as seguintes regras de processamento para o protótipo:

R01 Substantivos se tornam Classes Candidatas (CC) [1];
R02 Quando duas ou mais CC se sucedem no texto, a primeira mantém-se como CC enquanto as demais são classificadas como Atributos (AT) [6];
R03 Os termos: registro, sistema, informação, histórico, relatório e organização são desconsiderados como CC [9];
R04 Nomes próprios são desconsiderados como CC [4];
Verbos são Operações (OP) [16];
R05 Adjetivos são identificados como AT [16];
R06 Os verbos de ligação são desconsiderados como OP;
R07 Quando os termos: número, data, tipo e nome são sucedidos de um adjetivo, ou substantivo, ou preposição e substantivo, o conjunto de termos forma um único AT;
R08 CPF, RG e Telefone são AT;
R09 São removidas as duplicatas de CC [1] e seus respectivos plurais (quando houverem);
R10 AT e OP são vinculados as CC, contanto que a CC seja sujeito da oração onde se encontram;
R11 Somente CC com um AT ou OP são consideradas e exibidas;

Os insumos textuais são submetidos a cada uma das regras, na ordem disposta. No diagrama de atividades da Figura 02 são apresentados os passos em que um termo é submetido no protótipo para ser processado.

Para executar o protótipo é necessário estar no diretório do mesmo e rodar o comando "python app.py". Os insumos textuais devem ser adicionados ao arquivo "texto.txt" que deve estar no mesmo diretório do arquivo de script "app.py". Os verbos de ligação estão listados do arquivo "verbos.txt". Como resultado as classes são identificadas sequencialmente e então listadas ao final da execução. Os códigos-fonte da ferramenta desenvolvida estão disponíveis em https://github.com/danielconte/appClasses.

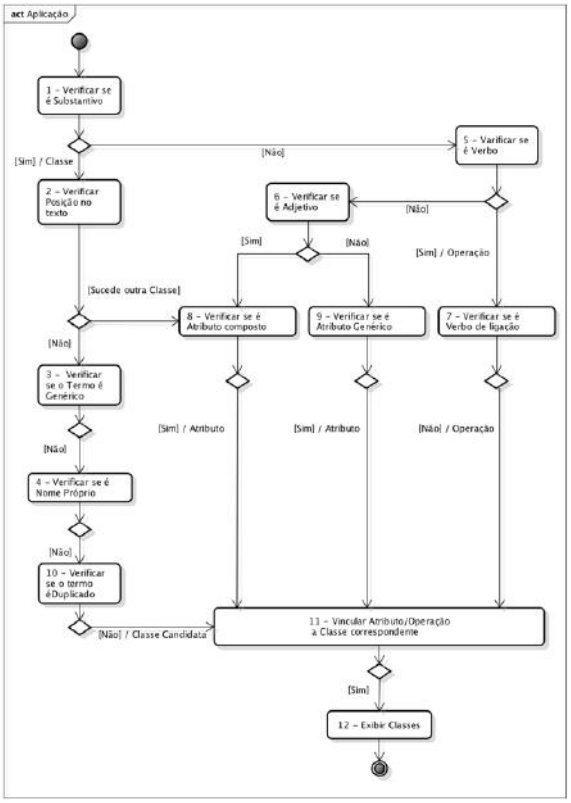

Figura 02 - Diagrama de atividades do processamento de um termo no protótipo

5. AVALIAÇÃO INICIAL DO PROTÓTIPO

No intuito de avaliar a utilização da identificação automática de classes foi realizada um estudo de caso comparando a identificação automática à identificação manual por analistas experientes.

Inicialmente foram selecionados oito analistas de sistemas experientes dentre os alunos do curso de Especialização Lato Sensu em Engenharia de Software de uma Universidade por possuírem pelo menos três anos de experiência na área de análise e desenvolvimento de sistemas, e por conhecerem técnicas de análise, sendo todos graduados na área de informática.

Duas especificações de requisitos, extraídas de casos encontrados na indústria de software, parcialmente apresentadas na tabela 01, foram então distribuídas de maneira aleatória entre os analistas, sendo que cada um dos analistas analisou o texto e identificou as classes presentes. O envio e retorno das solicitações foi realizado por e-mail, sendo que cada analista dispôs de 48 horas para identificar as classes no nível de análise, ou modelo de domínio [5]. O formato da resposta foi livre.

Os mesmos textos foram submetidos ao protótipo, de forma a identificar as classes que seriam posteriormente comparadas àquelas identificadas manualmente pelos analistas. Destas respostas, foram consideradas corretamente identificadas pela

ferramenta aquelas classes que apareceram em pelo menos 50% dos diagramas elaborados manualmente pelos analistas.

Tabela 01- Extrato dos Textos para avaliação do protótipo

Texto 1
O sistema deve gerenciar os treinamentos de uma empresa. Deve controlar os participantes, os instrutores e os temas debatidos. Para cada treinamento podem haver vários participantes. [...]
Texto 2
O sistema deverá gerenciar fisioterapeutas, assistentes de fisioterapeutas, atendentes e pacientes. Para todos eles devem ser cadastrados os dados necessários para emissão de nota fiscal e de contatos. [...]

As classes identificadas pelo protótipo comparadas às apontadas pelos analistas (Figura 03). As classes *comuns* são as classes que aparecem tanto no resultado da ferramenta quanto no elencado pelos analistas.

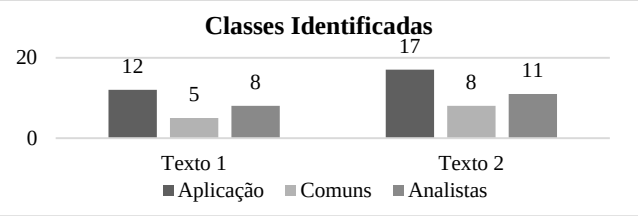

Figura 03 – Classes Identificadas

Analisando-se a Tabela 02, é possível perceber que do texto 1 foram extraídas 62,5% das classes com exatidão. No texto 2 o índice de acerto foi de 72,7%. No geral o resultado foi de 68,4%.

Tabela 02 - Classes Comuns e Diferentes

	Analistas	**Comuns**	**Protótipo**
Texto 1	Certificado, Instrutor, Tema	Cargo, Empregado, Participante, Setor, Treinamento	Carga, Controle, Emissão, Exemplo, Identificação, Título
Texto 2	Fisioterapeuta, Material, Usuário	Assistente, Atendente, Atendimento, Nota, Paciente, Pagamento, Sala, Serviço	Acesso, Comissão, Contato, Dados, Dias, Duração, Emissão, Hora, Procedimento

É possível observar que dentre as palavras não corretamente identificadas estão neologismos e anglicismos, comuns na área de de software, o que mostra que a adição de um léxico atualizado se faz necessária. Dos falso-positivos encontrados, ainda a maioria era de sinônimos ou partes de outras classes.

Comparando-se os resultados obtidos com aqueles encontrados nos trabalhos correlatos, existe o relato de 88% de precisão em língua inglesa [4]. Nos demais trabalhos relatados na seção de trabalhos correlatos, entretanto, os resultados foram expressos em formatos não comparáveis aos da presente pesquisa.

6. CONSIDERAÇÕES FINAIS

Este trabalho apresenta o desenvolvimento de um protótipo para identificação de classes a partir de requisitos utilizando processamento de linguagem natural. Para atingir esse objetivo foi realizado um estudo dos conceitos, seguido de um mapeamento dos trabalhos correlatos e do desenvolvimento um protótipo que realiza tal tarefa baseada em documentos textuais de análise.

Os resultados do processamento foram inicialmente comparados à análise manual realizada por oito analistas de sistemas, tendo como resultado 69,4% de acerto, sendo que 81,1% das classes gramaticais foram corretamente identificadas. Esses dados iniciais, apoiados pelos trabalhos correlatos, levantam indícios de que o uso do protótipo seja viável em termos de assertividade.

Numa próxima versão pretende-se que atributos, operações e associações sejam também identificados, bem como a aplicação das técnicas de treino presentes no processamento de linguagem natural, onde o usuário informaria à ferramenta dos equívocos para futuras utilizações, o que agregará valor ao protótipo.

7. REFERÊNCIAS

[1] Abbott, R. J. 1983. Program design by informal English descriptions. *Communications of the ACM*, *26*(11), 882-894.

[2] Bezerra, E. 2006. *Princípios De Análise E Projeto De Sistemas Com Uml-3ª Edição* (Vol. 3). Elsevier Brasil.

[3] Dias, F., Morgado, G., Oscar, P., da Silveira, D. S., Alencar, A. J., Lima, P., & Schmitz, E. A. 2006. Uma Abordagem para a Transformação Automática do Modelo de Negócio em Modelo de Requisitos. In *WER* (pp. 51-60).

[4] Herchi, H., Abdessalem, W. B. 2012. From user requirements to UML class diagram. International *Conference on Computer Related Knowledge*.

[5] Larman, C. 2007. *Utilizando UML e padrões: uma introdução à análise e ao projeto orientados a objetos e ao desenvolvimento iterativo.* Bookman.

[6] Mala, G. A., & Uma, G. V. 2006. Automatic construction of object oriented design models [UML diagrams] from natural language requirements specification. In *PRICAI 2006: Trends in Artificial Intelligence* (pp. 1155-1159). Springer Berlin Heidelberg.

[7] Mioto, C. 2009. *Sintaxe do Português.* LLC/CCE/UFSC.

[8] Mioto, C., Silva, M., Lopes R. 2000. *Manual de Sintaxe*. Insular.

[9] Omar, N., Hanna, J. R. P., & McKevitt, P. 2004. Heuristic-based entity-relationship modelling through natural language processing. In *Artificial Intelligence and Cognitive Science Conference (AICS)* (pp. 302-313).

[10] OMG. 2005. *Unified Modeling Language UML Version 2.5.*.

[11] Pressman, R. S. 2011. *Engenharia de Software: Uma abordagem profissional*. AMGH.

[12] Russel, S., Norvig, P. 2004. *Inteligência Artificial*. Elsevier.

[13] Freitas, C., Rocha, P., & Bick, E. 2008. Floresta sintá (c) tica: bigger, thicker and easier. In *Computational Processing of the Portuguese Language* (pp. 216-219). Springer Berlin Heidelberg.

[14] da Silva, W. C., & Martins, L. E. G. 2008. PARADIGMA: Uma Ferramenta de Apoio à Elicitação e Modelagem de Requisitos Baseada em Processamento de Linguagem Natural. *WER*, *8*, 140-151.

[15] Shalloway, A., Trott, J. R. 2004. *Explicando padrões de projeto: uma nova perspectiva em projeto orientado a objeto*. Bookman.

[16] Sommerville, I. 2011. *Engenharia de Software*. Prentice Hall.

[17] International Organization for Standardization, ISO/IEC/IEEE 29148:2011 - Systems and software Engineering — Life cycle processes — Requirements Engineering," ISO/IEC/IEEE, Nov. 2011.