Seleção de padrões para a arquitetura de software: uma abordagem baseada em procura de termos e sinônimos

Rafael P. M. Azevedo[1], Liziane Santos Soares[1], José Luís Braga[1]

[1]Departamento de Informática
Universidade Federal de Viçosa (UFV) – Viçosa, MG – Brasil

{rafael.azevedo, liziane.soares, zeluisbraga}@ufv.br

Resumo. Devido ao grande número de padrões disponíveis na literatura, a seleção dos padrões mais adequados para cada sistema é tarefa difícil de ser realizada manualmente. Este artigo apresenta uma abordagem para recomendar padrões através da procura de termos-chave na descrição dos padrões. Atributos de qualidade, parâmetros de atributos de qualidade e táticas arquiteturais foram extraídos da literatura e utilizados como termos-chave. A entrada da abordagem consiste em termos-chave que representem os requisitos de um dado sistema e a saída consiste de uma listagem com os padrões mais recomendados e menos recomendados para este sistema.

Palavras-chave. arquitetura de software, sistema de recomendação, seleção de padrões, reuso de software

1 Motivação

A arquitetura de software relaciona-se com a organização fundamental de um sistema de software e inclui a especificação de seus componentes, relacionamentos, ambiente e princípios nos quais se baseiam seu projeto e evolução [4] [14].

A definição de uma arquitetura envolve decisões cujos efeitos impactam todos os estágios subsequentes do ciclo de desenvolvimento do software. O projeto de uma arquitetura é um processo iterativo e tem como passo inicial a escolha de uma estruturação que suporte a implementação dos requisitos. Esta estruturação pode estar baseada em soluções já testadas, comumente documentadas como padrões de arquitetura [9]. O reuso de soluções diminui riscos de retrabalho e possui impacto direto sobre a qualidade do software. A escolha desta estratégia envolve inúmeras possibilidades, sendo que escolher a alternativa mais adequada é um dos grandes desafios enfrentados pelo arquiteto de softwares [4].

O crescente número de padrões documentados na literatura e disponíveis em repositórios online, torna inviável a tarefa de selecionar padrões manualmente. A título de exemplo, Henninger e Corrêa [13] identificaram 2241 padrões descrevendo soluções para problemas de projeto de software.

A tarefa torna-se particularmente difícil de ser resolvida considerando-se desenvolvedores inexperientes. De acordo com [26] somente engenheiros de software experientes que possuem conhecimento aprofundado sobre padrões, são capazes de utilizá-los de forma eficaz. Estes profissionais são capazes de reconhecer situações genéricas onde um padrão pode ser aplicado. Desenvolvedores menos experientes, mesmo tendo acesso a livros e repositórios sobre padrões, sempre esbarrarão na dificuldade de decidir entre reuso ou desenvolvimento de uma solução de propósito específico.

Este trabalho apresenta uma abordagem para o problema da seleção dos padrões mais adequados a serem utilizados na composição da arquitetura de um sistema. Os pontos fortes da abordagem estão relacionados (i) à maior facilidade de adicionar novos padrões e outros elementos à base de conhecimento, para que sejam considerados na seleção de padrões, (ii) ao fato de a entrada da abordagem consistir em termos que já são descritos entre os requisitos. O documento de requisitos, independente do *template* utilizado, é um artefato imprescindível ao desenvolvimento de um software. Assim, mesmo considerando equipes ou projetos com perfis diferentes, todos eles podem fazer uso da abordagem, já que a sua entrada estará contida neste artefato.

O restante deste artigo está estruturado da seguinte forma: a seção 2 apresenta os trabalhos relacionados a este trabalho. A seção 3 contextualiza os conceitos utilizados na abordagem e que compõem o modelo conceitual que a embasa. A seção 4 apresenta a abordagem. A seção 5 avalia e discute os resultados obtidos a partir da aplicação da abordagem. A seção 6 apresenta os comentários finais sobre o trabalho.

2 Trabalhos relacionados

Um dos desafios inerentes à tarefa de selecionar padrões consiste na organização de bases de conhecimento que possibilitem a obtenção de informações úteis à seleção.

Neste sentido, alguns trabalhos optam por desenvolver suas bases a partir das experiências passadas dos usuários na aplicação de padrões [8]. Outros formalizam as descrições dos padrões de maneira à facilitar o processo de seleção [21] e ainda há os que apostam no mapeamento das relações entre requisitos não funcionais e padrões como sendo elemento central de suas abordagens [10] [25]. Objetivando aproximar requisitos e padrões, alguns trabalhos utilizam ainda linguagens de descrição de requisitos para formalizar essas relações [27].

Contudo, um ponto comum entre a maioria destas abordagens é a necessidade de analisar cada descrição de padrão manualmente para se obter estas relações. Este cenário se revela inviável dentro de um contexto onde existem mais de 2000 padrões de software a serem analisados [13] e outros tantos surgindo a medida que novos problemas são resolvidos.

Estas abordagens apresentam limitações relacionadas à escalabilidade, o que fica evidente quando consideramos os mecanismos de extração de conhecimento utilizados. Os artifícios mais comuns englobam o uso de árvores de decisão, sistemas de regras de inferência [25] ou *Goal Question Metrics* [21]. Em todos estes casos observa-se a necessidade de um especialista capaz de elaborar a estrutura de raciocínio que considere cada padrão da base de padrões. Alguns trabalhos buscam amenizar esta

necessidade utilizando-se de artifícios matemáticos como forma de raciocinar sobre a base de conhecimento [29].

A adequação da abordagem ao ciclo de vida de desenvolvimento de software é outro desafio inerente à tarefa de selecionar padrões. Em alguns trabalhos a disponibilização das recomendações acontece utilizando-se como entrada, requisitos definidos através de notações muito específicas [27] ou respostas a questionários [22]. O problema é que o uso destas formas de entrada pode tornar o ciclo de vida do projeto moroso e cansativo, onerando o desenvolvimento.

Neste sentido, alguns trabalhos utilizam a ocorrência de palavras chave como forma de classificar padrões e inferir conhecimento. A premissa é que a ocorrência de um mesmo termo chave em um requisito e em um padrão indica que o uso deste padrão afeta de alguma maneira, o atendimento ao requisito. Nos trabalhos que seguem tal linha estas palavras variam de conjuntos não estruturados minerados manualmente a partir de cada padrão [11] até conjuntos semiestruturados onde termos chave se encaixam dentro de conceitos que são utilizados como base para a seleção.

3 Contexto

Padrões arquiteturais compreendem soluções bem sucedidas quando utilizadas para criar arquiteturas com características recorrentes em sistemas de software [5]. As vantagens de se utilizar padrões para compor a arquitetura de um sistema ultrapassam os benefícios do reuso de soluções já testadas. O uso de padrões facilita a comunicação de decisões arquiteturais ao time de desenvolvedores [9], melhora o nível de documentação [12] e o grau de conformidade entre arquitetura proposta e arquitetura implementada.

Os padrões arquiteturais costumam ser especificados através da descrição textual de seus componentes, responsabilidades, relacionamentos e maneiras pelas quais eles colaboram entre si. Tal descrição geralmente é dividida em seções que variam de acordo com o *template* adotado. Uma das premissas para a abordagem proposta aqui é que as seções carregam carga contextual suficiente pela qual é possível inferir o teor semântico dos termos que nela ocorrem. Assim, espera-se que uma seção que disserta sobre o contexto de aplicação do padrão ou sobre o problema a que ele se propõe resolver apresente termos que apontem características positivas atingidas pela aplicação do padrão. Em contrapartida, espera-se que termos que aparecem em uma seção que disserta sobre as fraquezas de um padrão indiquem características nas quais a aplicação do padrão impacta negativamente. Existem ainda seções neutras onde a conotação dos termos que nela ocorrem pode apresentar um grau de uniformidade menor. A Tabela 1 apresenta tanto as seções onde a procura por termos chave para a classificação e recomendação de padrões é realizada, como também a conotação esperada para os termos que nelas aparecem. A adoção destas seções especificamente foi feita com base em dois *templates* amplamente utilizados pela comunidade de padrões a saber: (i) *template* utilizado nos volumes da série de livros *Pattern Oriented Software Architecture* e definido em [5] (ii) *template* utilizado no livro *Design Patterns:*

Elements of Reusable Object-Oriented Software [7]. Foram adotados os termos que referenciam as seções nestes *templates*.

Dentre estas, vale destacar o caso particular da seção *consequences:* ambos os *templates* considerados descrevem consequências positivas e negativas provocadas pela aplicação de um padrão nesta mesma seção**.** Apesar da impossibilidade de inferir a conotação de termos chave que ocorrem nesta seção, ela não pode ser desconsiderada durante a procura, uma vez que disserta exatamente sobre o impacto do padrão sobre atributos de qualidade e parâmetros de atributo de qualidade, conceitos nos quais se encaixam a maioria dos termos chave considerados na procura como será visto no decorrer do artigo. Diante disso, fez-se necessário subdividir esta seção em *benefits* e *liabilities*. A separação deve ser realizada manualmente ao se alimentar a base de dados com algum padrão definido em algum *template* que não separe consequências positivas e consequências negativas relativas à aplicação do padrão.

Tabela 1. *Template* utilizado para a descrição de padrões.

***template* (i)**	***template* (ii)**	***template* adotado**	**conotação**
Name	Name	Name	Positiva
Problem	Intent	Problem	Positiva
Context	Applicability	Context	Positiva
Consequences	Consequences	Benefits	Positiva
		Liabilities	Negativa

As recomendações de padrões são baseadas nas ocorrências de termos chaves em descrições de padrões. Um termo chave é qualquer palavra ou expressão que denote um atributo de qualidade, um parâmetro de atributo de qualidade ou uma tática de arquitetura. Na Fig. 1, é apresentado o modelo conceitual que embasa a abordagem. Instâncias do modelo conceitual dão origem a hierarquias de termos chaves (a Fig. 2 apresenta um fragmento de uma dessas instâncias), e estes termos são utilizados para rotular e consequentemente parametrizar a seleção de padrões. Os conceitos presentes no modelo conceitual (Fig. 1) são detalhados no restante desta seção.

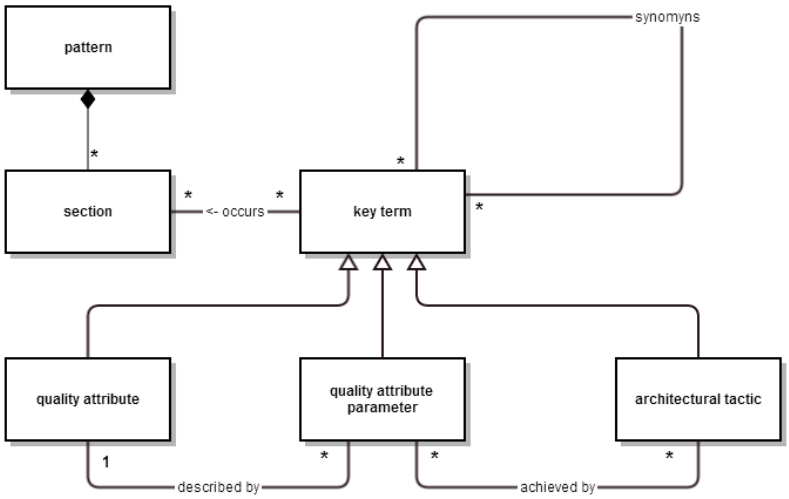

Fig. 1. Modelo conceitual utilizado na abordagem

Atributos de qualidade (AQs) são definidos como propriedades não funcionais desejadas em um sistema [3]. Alguns exemplos de AQs incluem performance, manutenibilidade (*maintainability)*, segurança (*security),* portabilidade *(portability)*, instalabilidade *(instalability)* entre outros. Os atributos de qualidade são usados como critérios para avaliar a execução de um software. Em geral consistem em critérios amplos, estando cada um deles relacionados a outras características ou requisitos mais específicos do sistema. Por exemplo, o atributo de qualidade *performance* pode estar relacionado à quantidade de tempo dispensada para realizar uma determinada tarefa *(latency)*, à quantidade de dados que um sistema é capaz de produzir em uma determinada unidade de tempo (*throughput)* ou mesmo à maneira como os recursos são gerenciados pela aplicação (*resource management)*.

Devido a este caráter amplo, é difícil selecionar padrões baseando-se somente em AQs desejados para o sistema. É necessário estabelecer outra forma que permita descrever em que aspectos o uso de um padrão impacta o alcance de uma característica de qualidade.

O conceito de parâmetro de atributo de qualidade (PAQ) idealizado em [24] preenche esta lacuna e se refere à questões comuns em especificações de software que estejam intimamente ligadas ao atendimento de atributos de qualidade em um sistema sendo desenvolvido. Estas questões comuns caracterizam de forma mais detalhada o atributo de qualidade desejado e servem como critérios que podem ser utilizados para avaliar em que nível o sistema atende ao requisito desejado. Além disso, permitem priorizar os aspectos sob os quais um requisito é desejado.

A utilização de PAQs permite a classificação e consequente seleção de padrões com maior nível de granularidade, isto é, padrões mais específicos para as características do sistema.

O fato de AQs e PAQs pertencerem ao domínio da descrição do problema possibilita categorizar requisitos com níveis aceitáveis de granularidade utilizando-se de termos que os denotem. Descrições de padrões por sua vez, ainda que descrevam elementos do domínio do problema enumeram também elementos contidos no domínio da solução. Estes elementos do domínio da solução apresentados nas descrições de padrões são as *táticas arquiteturais* [4].

Táticas arquiteturais consistem em decisões de projeto que influenciam a maneira como o sistema responde a um atributo de qualidade, dado um estímulo que permita avaliar este atributo [4]. Por exemplo, padrões com impacto positivo na construção de sistemas de baixa latência (*low latency)* costumam utilizar táticas como o aumento do grau de concorrência (*enhance concurrence*) ou o controle da taxa de disparo de eventos (*manage event rate*) para diminuir o tempo de resposta de um sistema. Desta forma, se a descrição de um padrão deixa explícito que aplica conceitos de concorrência para resolver um problema, ou mesmo, resolve um problema decorrente da aplicação de concorrência em um sistema, este padrão contribui para uma arquitetura capaz de suportar funcionalidades com baixo tempo de resposta. Bass, Clements e Kazman discorrem sobre esta relação entre padrões e táticas ao descrever padrões como pacotes de táticas utilizados para resolver problemas dentro de determinados contextos [4].

Por fim, um mesmo termo pode ser mencionado de inúmeras maneiras em um mesmo texto. Por isso, os sinônimos dos termos chave pertencentes à base de dados também devem ser considerados na procura. O modelo conceitual da Fig. 1 apresenta alguns outros conceitos, os quais são descritos brevemente a seguir:

- *Pattern*: este conceito representa os padrões a serem usados na arquitetura e que serão buscados através da abordagem proposta
- *Pattern Section*: todo padrão é descrito através de seções. A descrição de um padrão pode conter várias seções. Neste trabalho, as seções consideradas para a descrição de cada padrão são aquelas definidas no *template* apresentado na Tabela 1.
- *Key Term*: um termo chave é usado na realização da busca pelos padrões. Um termo chave consiste em um atributo de qualidade, parâmetro de atributo de qualidade ou tática. Um termo chave pode possuir sinônimos.

4 Abordagem para a seleção de padrões

A construção da base de conhecimento inicial da abordagem, foi realizada a partir de um conjunto de termos obtidos em [24]. A procura por termos chave na literatura, em particular AQs e PAQs, continuou através da análise de métodos de desenvolvimento de arquitetura [4] [9] [28], métodos e padronizações para avaliação de qualidade da arquitetura [2] [15] [4] [19] e modelos de qualidade para avaliação do software [6] [17] [18] [16]. Por sua vez, táticas arquiteturais e suas relações com PAQs foram mineradas notadamente de [4] e trabalhos percursores deste como [1] e [20].

Bases de dados com léxicos da língua inglesa como *WordNet*[1], *Thesaurus.com*[2] e *MerriamWebster*[3] foram utilizadas para identificar os sinônimos dos termos encontrados na literatura. Após analisar o termos, foram incluídos na base apenas aqueles que se encaixam no contexto de requisitos e arquitetura de software. O modo como estas bases foram utilizadas considerou dois casos: termos simples – formados por uma única palavra e termos compostos – formados por mais de uma palavra.

Termos simples foram submetidos a consultas nestas bases e os resultados obtidos foram analisados manualmente de forma a identificar aqueles utilizados no contexto semântico do termo chave em questão. Para ilustrar essa questão, a Tabela 2 mostra sinônimos encontrados para o termo chave *assurance*. Por sua vez, termos compostos foram decompostos em suas unidades lexicais. Descartou-se unidades lexicais de menor valor semântico (preposições, conectivos, artigos) e as restantes foram submetidas a consultas nas mesmas bases de sinônimos às quais termos simples foram submetidos. Os resultados obtidos de cada unidade lexical do termo composto foram combinados de forma a gerar os sinônimos. A Tabela 3 exemplifica mostrando os resultados obtidos ao aplicar este processo para obter sinônimos para *attack resistance*.

A identificação de termos resultou em uma hierarquia composta por aproximadamente trezentos termos chaves. A Fig. 2 apresenta um fragmento da hierarquia gerada, referente ao atributo de qualidade *security*.

Tabela 2. Sinônimos de "assurance"

termo simples	**sinônimos**
assurance	confidence, sureness

Tabela 3. Sinônimos para "attack resistance"

termo composto	unidade léxica	sinônimos da unidade léxica	sinônimos do termo composto
attack resistance	attack	intrusion, invasion, encroachment, violation, usurpation, trespass	intrusion resistance, invasion resistance, encroachment resistance, violation resistance, usurpation resistance, trespass resistance, attack immunity, intrusion immunity, invasion immunity, encroachment immunity, violation immunity, usurpation immunity, trespass immunity.
	resistance	immunity	

Este trabalho não tem a pretensão de englobar um conjunto absoluto de atributos de qualidade, parâmetros de atributo de qualidade e táticas arquiteturais existentes na literatura. O estudo realizado em busca desses elementos teve como objetivo alcançar um conjunto com o maior número possível de elementos. O conjunto identificado possibilita a construção de uma base de conhecimento inicial que auxilie na validação da abordagem proposta.

[1] http://wordnet.princeton.edu/
[2] http://thesaurus.com/
[3] http://www.merriam-webster.com/

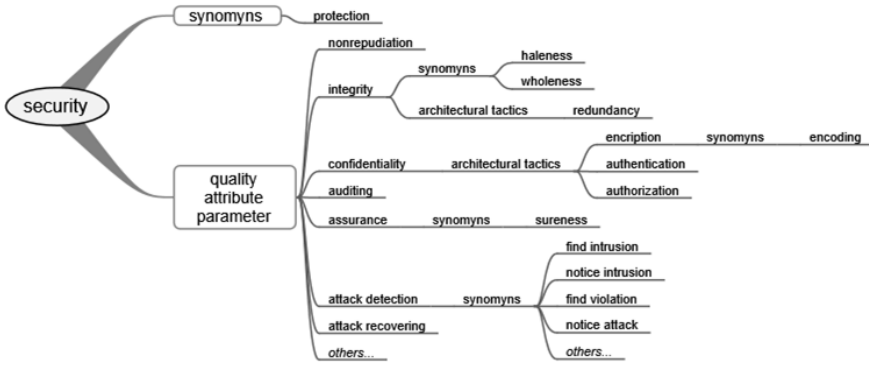

Fig. 2. Hierarquia de termos chave para o atributo de qualidade *security*

Em relação à procura de termos na descrição dos padrões, a procura de termos é precedida de uma fase de pré-processamento onde os elementos lexicais tanto das descrições dos padrões, quanto dos termos chave são submetidos ao algoritmo de *Porter* [23]. Esse algoritmo é capaz de reduzir palavras da língua inglesa a seus radicais. Isto se faz necessário para que a procura não fique sujeita as variações morfológicas de uma palavra. Uma vez realizado o pré-processamento, a contagem da quantidade de vezes que um termo aparece em cada seção de cada padrão é calculado, sendo que no caso dos termos compostos a ocorrência do mesmo é definida pela ocorrência de todas as suas unidades lexicais significantes em uma mesma sentença. Desta forma, o termo chave *multiple user interface* ocorre uma vez na sentença (1), ainda que suas unidades lexicais não apareçam na sentença na mesma forma e nem na mesma ordem em que aparecem na descrição do termo chave.

"In some rich-client ***user interfaces***, ***multiple*** *views of the same data are shown at the same time".* (1)

Na abordagem, a procura de termos permite encontrar o número de ocorrências de cada termo da base em cada seção de cada padrão. Esta informação é utilizada para calcular as recomendações. O cálculo das recomendações é realizado em quatro etapas.

- **Primeira etapa:** Cada ocorrência de termo chave em um padrão é submetida à função *M(tipo do termo, seção)*, definida na Tabela 4. A função mapeia a ocorrência de um termo *t* de um determinado tipo *(AQ- Atributo de Qualidade, PAQ-Parâmetro de Atributo de Qualidade, TA-Tática Arquitetural)* em uma seção *s* da descrição de um padrão para um fator *F* que indica o impacto que cada ocorrência do termo apresenta sobre a adequação deste padrão com este termo. O impacto pode ser positivo ou negativo de acordo com a seção em que o termo ocorre.

Tabela 4. - Função de mapeamento M(t, s)

F=M(t, s)	s = name	s = context	s = problem	s = strengths	s = liabilities
t = AQ	4	1	1	2	-2
t = PAQ	2	0,5	0,5	1	-1
t = TA	1	0,25	0,25	0,5	-0,5

- **Segunda etapa:** Os fatores gerados para cada ocorrência são condensados em uma matriz de afinidade parcial *A(termo, padrão)* onde cada célula representa o somatório dos fatores de impacto gerados para cada ocorrência do termo nos padrões da base. Assim, se o termo *concurrency* ocorreu uma vez na seção *problem*, e duas vezes na seção *strengths* do padrão *Active Object*, o valor da matriz em *A(termo = concurrency, padrão = Active Object) = 0,25 x 1 + 0,5 x 2 = 1,25*.
- **Terceira etapa:** A afinidade final de um termo *t* do tipo = *(AQ ou PAQ)* com um padrão é calculada através do somatório de *A(t ,p)* com as afinidades parciais geradas para todos os termos que estiverem abaixo do termo *t* na hierarquia de termos. Utilizando a Fig. 2 para exemplificar, a afinidade final do parâmetro de atributo de qualidade *integrity* com um padrão p é a soma de *A(t=integrity, p) + A(t=wholeness, p) + A(t=haleness, p) + A(t = redundancy, p)*.
- **Quarta etapa:** Um conjunto C = $\{t_1, t_2, t_3, t_4 ..., t_n\}$ composto por termos que denotam atributos de qualidade e parâmetros de atributo de qualidade é fornecido como entrada para a recomendação de padrões. Esses termos estão entre os requisitos do sistema sendo desenvolvido e são informados pelo usuário da abordagem. Considerando o sistema com os requisitos do conjunto C, o grau de recomendação de um determinado padrão da base de conhecimento para o sistema é calculado somando-se as afinidades finais de todos os termos do conjunto C em relação a cada padrão da base de dados. Os padrões que obtiverem maior grau de afinidade são os mais recomendados para o sistema em questão.

5 Validação

Uma vez proposta a abordagem para a recomendação de padrões, o passo seguinte consistiu em verificar a sua efetividade através do desenvolvimento de um protótipo que implementa a abordagem.

A prova de conceito é composta por dois módulos: módulo de procura e módulo de recomendação. O módulo de procura é responsável por extrair informação a partir da descrição textual de padrões baseada na utilização de termos chave. O módulo de recomendação é responsável por traduzir os resultados da procura em recomendações baseadas nos requisitos da aplicação, informados por meio de um conjunto C = $\{t_1, t_2, t_3, t_4, ..., t_n\}$, onde cada t_i é um termo que representa um requisito não funcional da aplicação.

Os impactos de padrões arquiteturais sobre um conjunto de atributos de qualidade foram analisados manualmente por Harrison e Avgeriou em [12] e os resultados obtidos foram comparados com aqueles obtidos pelo protótipo que implementa a abordagem proposta.

O experimento consistiu das seguintes etapas: para cada atributo de qualidade *a* analisado em [12] determinou-se um conjunto teste C_T = {a}; o conjunto C_T foi utilizado como entrada para a prova de conceito que determinou quais os padrões mais recomendados, menos aconselhados e indiferentes para um sistema onde C_T represente os requisitos não funcionais mais importantes. Os padrões não analisados por Harrison e Avgeriou foram excluídos do conjunto solução, restringindo o experimento aos oito padrões e seis atributos de qualidade analisados por [12].

Os resultados obtidos são apresentados na Tabela 5, onde cada célula apresenta uma dupla PC/HA em que PC representa o resultado obtido pela prova de conceito e HA o resultado obtido manualmente por Harrison e Avgeriou. As células destacadas em cinza indicam onde os resultados obtidos automaticamente pela prova de conceito e manualmente por Harrison e Avgeriou divergem. As letras *N*, *S* e *L* denotam relações de neutralidade (*neutral)*, benefício (*strength)* e fraqueza (*liability)* entre padrões e termos. Estes valores indicam respectivamente que a aplicação do padrão "p" é recomendada, não recomendada, e indiferente em um sistema em que *"a"* seja o único atributo de qualidade desejado. O símbolo (*) indica que a aplicação do padrão possui impactos tanto positivos quanto negativos sobre o atributo de qualidade.

Tabela 5. - Comparação entre os resultados obtidos através do protótipo da abordagem e os resultados obtidos por Harrison e Avgeriou;

	Usability	Security	Maintainability	Performance	Reliability	Portability
Layers	N/N	N/S	S/S	L/L	L/S	S/S
Pipes and Filters	N/L	N/L	S/S	S/*	L/L	S/S
Blackboard	N/N	N/L	S/S	L/L	N/N	N/N
MVC	S/S	N/N	L/L	L/L	N/N	L/L
PAC	S/S	N/N	S/S	L/L	N/N	N/S
Microkernel	N/N	N/N	S/S	L/L	S/S	S/S
Reflection	N/N	N/N	S/S	L/L	N/L	N/S
Broker	N/S	N/S	S/S	L/N	L/L	S/S

Os resultados foram comparados de forma a obter medidas de *similaridade*, *precisão* e *recall*, métricas comumente utilizadas para avaliar trabalhos relacionados a extração de informação. Seguem as definições destas medidas adaptadas para o contexto deste trabalho:

- **Similaridade.** A razão entre a quantidade de recomendações da prova de conceito que coincidem com os resultados obtidos [12] e o total de recomendações analisadas.
- **Precisão.** A precisão é calculada segundo dois pontos de vista neste trabalho: (i) capacidade da prova de conceito recomendar a aplicação de um padrão corretamente, a qual denominamos *precisão de recomendação*; (ii) capacidade da prova de conceito 'desaconselhar' a aplicação de um padrão de forma correta, a qual denominamos *precisão de desaconselhamento*.

- **Recall.** Assim como a precisão, o recall é calculado segundo dois pontos de vista: (i) capacidade da prova de conceito recomendar os padrões que devem ser recomendados dado um atributo de qualidade, a qual denominamos *recall de recomendação*; (ii) capacidade da prova de conceito 'desaconselhar' a aplicação dos padrões que devem ser desaconselhados dado um atributo de qualidade, a qual denominamos *recall de desaconselhamento*.

A Tabela 5 permite inferir que a similaridade entre as classificações obtidas por [12] e pela ferramenta apresentada por este trabalho foi de 77%. Ao analisar a ferramenta em termos de precisão de recomendação encontrou-se que a prova de conceito recomenda padrões com precisão máxima (100%). Por outro lado, cerca de 30% das recomendações obtidas manualmente por [12] não foram obtidas pela ferramenta, o que confere um recall de recomendação de 70%. A ferramenta não mostrou a mesma eficácia ao desaconselhar o uso de padrões dado um atributo de qualidade, obtendo precisão de desaconselhamento de 83%, sendo a taxa de recuperação ao desaconselhar (*recall*) da ordem de 71%.

O índice de precisão de desaconselhamento justifica-se em parte devido a menor quantidade de informação sendo processada para identificar esta relação, uma vez que quatro das cinco seções consideradas durante a procura estão atreladas a descrição de relações positivas entre padrões e termos e apenas uma às relações negativas. Outro problema, é que a descrição de fraquezas costuma ser feita em termos de *trade-offs* a serem realizados ao aplicar o padrão, acarretando na ocorrência de termos que representam fraquezas do padrão em seções não apropriadas. Assim, introduz-se uma fraqueza ao custo de mencionar um benefício do padrão. Esta menção acaba por tornar-se um falso positivo na classificação. Os resultados obtidos para recall ocorrem em boa parte, devido a necessidade de uma análise mais profunda do padrão para identificar algumas relações. Nestes casos, o impacto de um padrão sobre um atributo de qualidade não é consequência direta, mas sim, inferida a partir de outros aspectos deriváveis da aplicação do padrão e portanto difíceis de identificar via procura de termos.

6 Conclusão

Este trabalho propõe uma abordagem para selecionar padrões durante o projeto de arquitetura, apoiada no uso de termos chave e de processamento de linguagem natural. Uma base de conhecimento inicial foi construída utilizando padrões de projeto e de arquitetura e um conjunto de termos chave composto por atributos de qualidade, parâmetros de atributo de qualidade, táticas arquiteturais e termos sinônimos.

Os resultados indicam que os termos chave levantados são encontrados nas descrições dos padrões e que os parâmetros de qualidade conseguem classificar padrões com maior grau de especificidade. A utilização de processamento de linguagem natural associada ao formato de entrada da abordagem (termos descritos entre os requisitos) facilita a aquisição de conhecimento uma vez que não é necessário um especialista para reescrever o padrão em um formalismo entendido pela abordagem o que gera impacto direto na escalabilidade da abordagem proposta.

O fato de o cálculo do grau de afinidade entre um padrão e um termo considerar não apenas a quantidade de ocorrências do termo no padrão, mas também o contexto das ocorrências e as ocorrências de termos relacionados contribue para a robustez do fator determinado e consequente alto grau de assertividade das recomendações, sendo esta uma das fortalezas deste trabalho.

Contudo é preciso observar que as classificações geradas não possuem o nível máximo de acuidade uma vez que nem sempre o teor da seção onde ocorre um termo chave indica com eficácia a conotação que o termo assume no texto. Analisamos a ferramenta de extração de conhecimento em termos das medidas de precisão e recall, além de analisar a efetividade da abordagem. Essa análise foi baseada na comparação dos resultados obtidos com os resultados conseguidos através da extração manual de conhecimento, baseada em critérios semelhantes realizada por trabalhos anteriores.

O próximo passo do trabalho consiste em realizar experimentos que validem a hipótese de que a utilização de parâmetros de atributo de qualidade como entrada para a seleção produz recomendações mais assertivas e granulares. Além disso, pretende-se investigar e aplicar técnicas que permitam diminuir o índice de falsos positivos.

7 Referências Bibliográficas

1. Bachmann, F., Bass, L., Nord, R.: Modifiability Tactics (CMU/SEI-2007-TR-002). In Softw. Eng. Institute, Carnegie Mellon Univ. (2007).
2. Barbacci, M.R., Klein, M., Longstaff, T. Weinstock, C.: Quality Attributes (CMU/SEI-95-TR-021). In Softw. Eng. Institute, Carnegie Mellon Univ. (1995).
3. Barbacci, M.R., Elison, A., Lattanze A.J., Stafford, J.A, Weinstorck, C.B., Wood, W.G.: Quality Attribute Workshops QAWs - Third Edition. In Softw. Eng. Institute, Carnegie Mellon Univ. (2003).
4. Bass, L., Clements, P., Kazman R.: Software Architecture in Practice, Second Edition. Addison Wesley (2003).
5. Buschmann, F., Meunier, R. Rohnert, H., Sommerlad, P., Stal, M.: Pattern-Oriented Software Architecture, Volume 1: A System of Patterns. Wiley, Chichester, UK (1996).
6. CISQ: Software structural quality characteristics. Consortium IT Software Quality - CISQ (2011).
7. Gamma, E. Helm, R., Johnson, R., Vlissides, J.: Design patterns: elements of reusable object-oriented software. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1995).
8. Gomes, P., Pereira, F.C., Paiva, P. Seco, N., Carreiro, P., Ferreira, J.L, Bento, C.: Using CBR for Automation of Software Design Patterns. Proceedings of the 6th European Conference on Advances in Case-Based Reasoning. pp. 534–548 Springer-Verlag, London, UK, UK (2002).
9. Gorton, I.: Essential Software Architecture (2. ed.). Springer (2011).
10. Gross, D., Yu, E.: From Non-Functional Requirements to Design through Patterns. Requir. Eng. 6, 18–36 (2000).
11. Guéhéneuc, Y., Mustapha, R.: A Simple Recommender System for Design Patterns. Proc. 1st Eur. Focus Gr. Pattern Repos. 1–2 (2007).
12. Harrison, N.B., Avgeriou, P.: Leveraging Architecture Patterns to Satisfy Quality Attributes. Lect. Notes Comput. Sci. 4758, 263–270 (2007).

13. Henninger, S., Corrêa, V.: Software pattern communities: current practices and challenges. Proceedings of the 14th Conference on Pattern Languages of Programs. pp. 1–19 ACM (2007).
14. IEEE: IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE Std 1471-2000. IEEE-Std-1471-2000, i–23 (2000).
15. IEEE: IEEE Standard for a Software Quality Metrics Methodology. IEEE Std 1061-1998. (1998).
16. ISO/IEC: ISO/IEC 25010. Systems and software engineering-Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models. ISO/IEC. (2011).
17. ISO/IEC: ISO/IEC 9126. Software engineering -- Product quality. ISO/IEC (2001).
18. ISO/IEC Standard: Software Engineering -- Product Quality -- Part 2: External Metrics. (2003).
19. Kazman, R., Klein, M., Clements, P.: ATAM : Method for Architecture Evaluation (CMU/SEI-2000-TR-004). Softw. Eng. Institute, Carnegie Mellon Univ. August, (2000).
20. Kim, S. et al.: The Journal of Systems and Software Quality-driven architecture development using architectural tactics. J. Syst. Softw. 82, 8, 1211–1231 (2009).
21. Palma, F., Farzin, H.: Recommendation System for Design Patterns in Software Development : An DPR Overview. 1–5 (2012).
22. Pearson, S., Shen, Y.: Context-aware privacy design pattern selection. Proceedings of the 7th international conference on Trust, privacy and security in digital business. pp. 69–80 Springer-Verlag, Berlin, Heidelberg (2010).
23. Porter, M.F.: An algorithm for suffix stripping. Progr. Electron. Libr. Inf. Syst. 40, 3, 211–218 (1980).
24. Soares, L.S., Price, R.T., Pimenta, M.S., Braga, J.L: Seleção de padrões para a arquitetura de software : abordagem baseada em parâmetros de atributo de qualidade. Proceedings of the 9th Latin American Conference on Pattern Languages of Programming (SugarLoafPLoP 2012). , Natal, RN (2012).
25. Soares, L.S., Price, R.T., Pimenta, M.S., Braga, J.L.: Selecting Architectural Patterns through a Knowledge-Based Approach. Proceedings of IADIS Applied Computing 2011. pp. 59–66 , Rio de Janeiro (2011).
26. Sommerville, I.: Software Engineering (9th Edition). Addison Wesley; 9 edition (2010).
27. Wang, J., Song, Y.T., Chung, L.: From software architecture to design patterns: a case study of an NFR approach. Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2005 and First ACIS International Workshop on Self-Assembling Wireless Networks. SNPD/SAWN 2005. Sixth International Conference on. pp. 170–177 (2005).
28. Wojcik, R., Bachmman, F., Bass, L. Clements, P., Merson, P., Nord, R., Wood, W.: Attribute-Driven Design (ADD), Version 2 . 0. (CMU/SEI-2006-TR-023). November, (2006).
29. Xavier, J.R., Werner, C.M.L, Travessos, G.H.: Uma Abordagem para a Seleção de Padrões Arquiteturais Baseada em Características de Qualidade. XVI Simpósio Brasileiro de Engenharia de Software (XVI SBES). pp. 52–67 , Gramado, RS, Brasil (2002).