UMA ABORDAGEM PARA INSPEÇÃO DE DOCUMENTOS ARQUITETURAIS BASEADA EM *CHECKLIST*

Rafael Ferreira Barcelos

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Guilherme Horta Travassos, D.Sc.

Profª. Claudia Maria Lima Werner, D.Sc.

Prof. Julio Cesar Sampaio do Prado Leite, Ph.D.

RIO DE JANEIRO, RJ – BRASIL

ABRIL DE 2006

BARCELOS, RAFAEL FERREIRA

Uma abordagem para inspeção de documentos arquiteturais baseada em *checklist* [Rio de Janeiro] 2006

VIII, 175 p., 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2006)

Dissertação – Universidade Federal do Rio de Janeiro, COPPE

1. Arquitetura de Software
2. Inspeção de Software

I. COPPE/UFRJ II. Título (série)

À minha família.

Agradecimentos

Aos meus pais, por me apoiarem integralmente na realização deste trabalho.

À Michelle, por compreender a minha ausência de dois anos, pelo amor e carinho. Sua atenção e disponibilidade para as centenas de horas falando pelo telefone, me dando força e escutando minhas inquietudes, foram fundamentais neste período.

Ao professor Guilherme Travassos, por apostar em nós e dar oportunidade de mostrar do que somos capazes. Agradeço também pela orientação e pela paciência.

Aos professores Cláudia Werner e Júlio Leite, por aceitarem participar de minha banca.

Ao companheiro de quarto Arilo pelo apoio e por agüentar dividir o grande apartamento por esses dois anos.

Ao Beto e Jobson por terem auxiliado na realização do estudo que foi realizado no contexto dessa pesquisa.

Aos companheiros de COPPE Aline, Ana Cândida, Gladys, Gleison, Kali, Luiz Gustavo, Marco Antônio, Mariano, Murta, Natanael, Paula, Paulo Sérgio, Rodrigo, Sávio, Sômulo, Tayana e Wladmir.

A FAPEAM, pelo apoio financeiro.

A BENQ/SIEMENS-AM pela confiança e pela participação nos estudos realizados nessa pesquisa. Um agradecimento especial para Andrey Patitucci e Rafael Kitauchi por possibilitarem a realização desse acordo.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UMA ABORDAGEM PARA INSPEÇÃO DE DOCUMENTOS ARQUITETURAIS BASEADA EM *CHECKLIST*

Rafael Ferreira Barcelos

Abril / 2006

Orientador: Guilherme Horta Travassos

Programa: Engenharia de Sistemas e Computação

A arquitetura de um software, representada através do documento arquitetural, é de grande importância para os *stakeholders* por ser utilizada em diversos momentos no processo de desenvolvimento do software. Portanto, a sua revisão se torna uma atividade relevante para o sucesso do projeto e para a melhoria da qualidade do software.

Existem diversas abordagens de avaliação descritas na literatura que objetivam avaliar esse artefato, mas elas apresentam limitações que dificultam a sua aplicação, principalmente em meio industrial.

Com o objetivo principal de minimizar as limitações presentes nas abordagens de avaliação, esta dissertação apresenta uma abordagem para inspeção de documentos arquiteturais baseada em *checklist* chamada ArqCheck. Devido à procura cada vez maior por parte das empresas de software brasileiras por abordagens que permitam tanto a melhoria de seus processos quanto a de seus produtos, optamos por avaliar ArqCheck através de uma metodologia experimental que visa a transferência de tecnologias de software para a indústria. Com isso, uma revisão sistemática e dois estudos experimentais foram realizados e os resultados preliminares indicam a viabilidade, a eficácia e o custo / eficiência da abordagem na identificação de defeitos em documentos arquiteturais.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

AN APPROACH TO INSPECT ARCHITECTURAL DOCUMENTS BASED ON *CHECKLIST*

Rafael Ferreira Barcelos

April / 2006

Advisor: Guilherme Horta Travassos

Department: Computer and Systems Engineering

Software architecture, usually represented by an architectural document, is extremely important to stakeholders since it is used in several moments throughout the software development process. Therefore, due to its importance, the reviewing of architectural documents becomes a fundamental activity for the success of the software project and for the software quality improvement.

There are several architectural evaluation approaches described in the literature, however they have some limitations related to their use that make difficult to apply them in industrial environments.

Aiming at to reduce such limitations, this dissertation describes a checklist based approach to inspect architectural documents called ArqCheck. Due to the Brazilian software organizations needs for approaches concerned with process and product quality improvements, we decided to evaluate ArqCheck using an experimental methodology that supports technological transference of software technologies to software organizations. Based on this, a systematic review and two experimental studies were conducted and preliminary results indicated the feasibility, effectiveness and cost/efficiency of this approach to detect defects on software architectural documents.

Sumário

Capítulo 1 – Introdução

Neste capítulo são apresentadas as motivações para a realização deste trabalho, além de seus objetivos e a forma como esta dissertação está organizada.

1.1 – Motivação

Desde a década de 90, com o aumento da complexidade e tamanho dos sistemas de software, a definição dos algoritmos e das estruturas de dados não são mais as maiores preocupações do projetista durante os primeiros estágios do processo de desenvolvimento de um software (GARLAN e SHAW, 1993). Em conseqüência, várias tarefas do processo de desenvolvimento deixaram de ser triviais como, por exemplo, a definição da estrutura da solução computacional a ser construída.

Uma forma encontrada para lidar com essa complexidade cada vez mais crescente foi o uso de abstrações durante a construção desses sistemas. Visto que a arquitetura de software auxilia na definição e visualização da solução computacional em um elevado nível de abstração (GARLAN e SHAW, 1993), ela passou a ser usada como ferramenta na definição da estrutura do software a ser construído.

A arquitetura de um software é considerada o primeiro conjunto de decisões de projeto e possui um grande papel como ponte entre os requisitos e a implementação. Essa estrutura é utilizada principalmente como ferramenta para comunicar a solução projetada aos diversos *stakeholders*[1] que participam do processo de desenvolvimento (GARLAN, 2000). Para que essa comunicação seja possível, essa estrutura deve ser descrita através de uma representação, conhecida como documento arquitetural.

Um documento arquitetural é portanto o documento que descreve a arquitetura de um software. Para isso, de acordo com as recomendações da IEEE 1471 (IEEE, 2000), esse documento deve principalmente identificar os elementos arquiteturais e seus relacionamentos e também descrever o papel de cada um desses elementos dentro da solução representada.

Esse documento é de grande importância para os *stakeholders* visto que ele contém informações que são utilizadas por diversos tipos de stakeholders e por ser utilizado em diversos momentos no processo de desenvolvimento do software.

---

[1] *Stakeholder*: grupo ou indivíduo envolvido, de forma direta ou indireta, em um projeto de software ou que possue algum interesse no resultado obtido por esse projeto.

Portanto, devido à essa importância, a sua revisão se torna uma atividade relevante para o sucesso do projeto e para a melhoria da qualidade do software. Esta afirmação é fortalecida se for considerado que (1) a avaliação desse artefato dificulta a propagação de seus defeitos para os demais artefatos, como diagramas de projeto e código fonte, e (2) o custo de correção desses defeitos é bem menor se for realizada durante os primeiros estágios do projeto (BOEHM, 1981; BOEHM e BASILI, 2001). Contudo, como avaliar se o documento arquitetural descreve uma arquitetura que atenda aos requisitos especificados?

O início das pesquisas que resultou na abordagem para avaliação arquitetural proposta ocorreu no contexto do projeto eSEE. Esse projeto tem como objetivo criar um ambiente que auxilie no planejamento e condução de estudos experimentais em Engenharia de Software (MIAN *et al.*, 2005).

A nossa participação nesse projeto ocorreu principalmente durante a especificação da arquitetura do ambiente (NETO *et al.*, 2004). Os envolvidos nesse projeto identificaram a necessidade em definir a arquitetura desse ambiente devido à sua complexidade e ao fato da equipe do projeto possuir uma alta rotatividade, por ser formada principalmente por alunos de pós-graduação. Com isso, o documento arquitetural do ambiente eSEE foi criado com o objetivo de documentar em um elevado nível de abstração a solução a ser desenvolvida, facilitando o seu entendimento.

Após a definição da arquitetura desse ambiente, optou-se por avaliá-la em relação aos requisitos especificados devido aos ganhos que poderiam ser obtidos com os resultados dessa avaliação nesse estágio do processo de desenvolvimento. Com isso, iniciamos uma busca pelas abordagens de avaliação arquitetural descritas na literatura.

Para essa busca, definimos algumas características que consideramos importantes e que deveriam estar presentes na abordagem selecionada a ser utilizada durante a avaliação da arquitetura do eSEE:

- **Genérica e Adaptável:** para que ela possa ser aplicada na avaliação de diferentes tipos de documentos arquiteturais, ficando independente, portanto da abordagem utilizada para especificá-los;
- **Simples**: para que não seja necessária a execução de elaboradas atividades para sua aplicação, a necessidade de tipos de conhecimento específicos ou a alocação de grandes quantidades de recursos;
- **Extensível**: para que seja facilmente modificável e permita avaliar, além dos conceitos inicialmente determinados, conceitos específicos ao contexto que estão sendo aplicados.

Essas características foram definidas com base em condições normalmente presentes em ambientes industriais e que podem influenciar na adoção de uma determinada tecnologia dentro desse contexto (SHULL *et al.*, 2001).

Sendo assim, após a realização dessa busca através de um estudo secundário (revisão sistemática), foram identificadas diversas abordagens de avaliação descritas na literatura que objetivam avaliar principalmente se a arquitetura projetada atende aos requisitos especificados para o produto (BARCELOS e TRAVASSOS, 2006b). Contudo, nenhuma abordagem atende às características que definimos e apresentam limitações por terem sido criadas principalmente em contextos com grande disponibilidade de recursos que dificultam a sua aplicação, principalmente em meio industrial.

Com a expectativa de minimizar as limitações presentes nas abordagens de avaliação e levando em consideração os resultados que se tem obtido com a aplicação de técnicas de inspeção na revisão de artefatos de software (SHULL *et al.*, 2000; CONRADI *et al.*, 2003), esta dissertação apresenta ArqCheck, uma abordagem para inspeção de documentos arquiteturais baseada em *checklist*.

Ao longo deste trabalho, arquitetura de software será chamada por diversas vezes simplesmente de "arquitetura", para simplificar a escrita e o entendimento do trabalho. O mesmo irá ocorrer com o termo processo de desenvolvimento de software, que será tratado somente como "processo de desenvolvimento".

1.2 – Objetivo do Trabalho

O objetivo desse trabalho é elaborar uma abordagem de avaliação que permita a inspeção de um documento arquitetural, que foi chamada de ArqCheck. Com isso, a hipótese de pesquisa definida é que através de uma abordagem para inspeção baseada em *checklist* que avalie um documento arquitetural seja possível identificar defeitos arquiteturais relacionados ao atendimento dos requisitos do software.

O *checklist* que compõe ArqCheck foi criado com base em conhecimentos obtidos dos principais conceitos e técnicas de projeto arquitetural (SHAW, 1995; BOSCH e MOLIN, 1999; BASS *et al.*, 2003) e das principais abordagens de documentação arquitetural descritas na literatura (IEEE, 2000; CLEMENTS *et al.*, 2004).

Existe atualmente um aumento constante da procura por parte das empresas de software brasileiras por abordagens que permitam tanto a melhoria de seus processos quanto a de seus produtos (MCT/SEPIN, 2005). Devido à essa procura, optamos por avaliar a abordagem para inspeção proposta através de uma metodologia experimental que objetiva analisar e evoluir novas tecnologias visando a sua transferência para um contexto industrial.

Essa metodologia, definida em (SHULL *et al.*, 2001), define uma série de estudos experimentais que devem ser executados para que a tecnologia avaliada adquira maturidade e tenha seus riscos de implantação em um contexto industrial minimizados.

Seguindo essa metodologia, dois dos estudos por ela definidos foram realizados. Esses estudos buscaram avaliar principalmente a viabilidade, a eficácia e o custo/eficiência da abordagem proposta.

No contexto dessa dissertação, definimos eficácia como a quantidade de defeitos encontrados em relação ao número de discrepâncias identificadas pelo inspetor. A caracterização da eficácia de ArqCheck permite fornecer informações em relação ao esforço gasto pelo inspetor que está realmente permitindo a melhoria da qualidade do documento arquitetural. Em relação ao conceito de custo/eficiência, ele consiste na caracterização de quantos defeitos são encontrados por unidade de tempo de inspeção. Através dessa métrica é possível caracterizar o auxílio fornecido pela abordagem na identificação de defeitos.

Através desses estudos, ArqCheck foi avaliada em relação a sua aplicação para revisar tanto documentos arquiteturais gerados em um contexto acadêmico quanto em documentos gerados em um contexto industrial. Os resultados desses estudos permitiram que evoluções fossem feitas na abordagem, e que no futuro a transferência dessa tecnologia para a indústria seja, de certa maneira, simplificada.

1.3 – Metodologia de Trabalho

Para a realização desse trabalho, optamos por nos basear em uma metodologia de pesquisa que utiliza estudos experimentais para (a) obter evidências e conhecimento em relação à área de Arquitetura de Software e (b) avaliar a abordagem para inspeção (MAFRA e TRAVASSOS, 2006)

Portanto, esse trabalho seguiu passos específicos que permitiram que os objetivos fossem atingidos (Figura 1.1):

- **Passo 1: Análise dos conceitos relacionados à construção e documentação de arquiteturas.** Para se entender como um artefato deve ser avaliado é essencial que se entenda em um primeiro momento como ele é construído. Sendo assim, procuramos analisar nesse passo o processo de criação desse artefato, o papel dos requisitos nesse processo e que tipo de informação deve estar presente em um documento arquitetural. Para isso, foi realizada uma análise das técnicas de construção e documentação arquitetural com o objetivo de identificar os principais

conceitos por eles utilizados e que poderiam ser utilizados como base para a definição de uma abordagem de avaliação.

- **Passo 2: Identificação das abordagens de avaliação arquitetural existentes**. Após identificar como a arquitetura de um software é construída, esse segundo passo foi realizado com o objetivo de buscar indícios que permitam identificar e caracterizar as abordagens de avaliação arquitetural existentes. Para isso, uma revisão sistemática (KITCHENHAM, 2004; BIOLCHINI *et al.*, 2005) foi planejada e executada, e a partir dos resultados obtidos uma caracterização dessas abordagens foi realizada.
- **Passo 3: Definição e construção da abordagem para inspeção.** A caracterização das abordagens de avaliação identificadas permitiu a observação de características positivas e algumas limitações. A compreensão desses métodos e a análise de suas características serviram como base para a definição e construção de uma abordagem para inspeção que foi chamada de ArqCheck.
- **Passo 4: Realização de estudos de avaliação**. Após a criação de ArqCheck, estudos primários foram planejados e executados para avaliar a sua viabilidade, eficácia e custo/eficiência em relação à revisão de documentos arquiteturais.

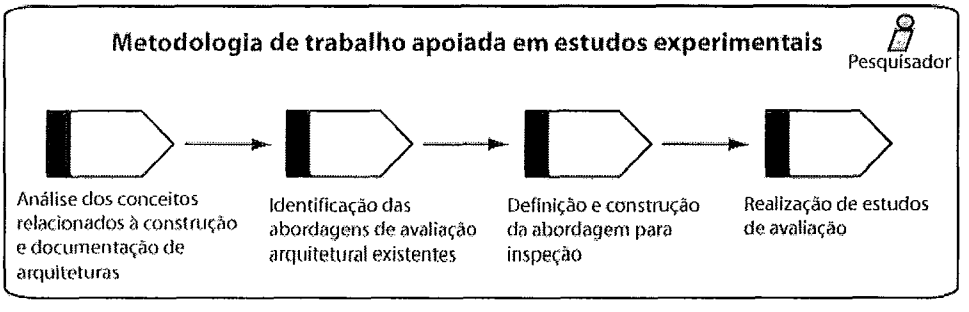


Figura 1.1- Metodologia de trabalho adotada

1.4 – Organização deste Trabalho

Além desta introdução, este trabalho é composto por outros cinco capítulos, quatro apêndices e quatro anexos.

No Capítulo 2, intitulado "Arquitetura de Software", são descritos os conceitos básicos relacionados ao projeto, à documentação e à avaliação arquitetural. Esse capítulo possui como objetivo mostrar o papel e a importância da arquitetura dentro do processo de desenvolvimento de software e também que tipo de conceito, informação e conhecimento são utilizados pelo arquiteto para transpor o *gap* entre os requisitos

especificados para o produto e a solução computacional que será implementada para atender a esses requisitos.

No Capítulo 3, intitulado "Abordagens de Avaliação Arquitetural", são descritos a revisão sistemática e os resultados obtidos. Nesse capítulo, as abordagens de avaliação arquitetural identificadas pela revisão sistemática são caracterizadas e analisadas. O resultado dessa revisão permite identificar o estado da arte em avaliação arquitetural, assim como identificar os pontos positivos e as limitações das abordagens selecionadas.

No Capítulo 4, intitulado "Inspeção de Documentos Arquiteturais", é apresentada a abordagem ArqCheck que é a abordagem proposta para inspeção de documentos arquiteturais baseada em *checklist*. Nesse capítulo, são descritos os motivos que levaram à criação dessa abordagem, como ela foi criada e os elementos que a compõe.

No Capítulo 5, intitulado "Avaliando a Abordagem Proposta", são descritos como os estudos foram planejados, executados e os resultados oriundos da análise dos dados obtidos. Durante a descrição dos resultados, são identificadas a viabilidade, a eficácia e o custo/eficiência em se aplicar ArqCheck na identificação de defeitos em documentos arquiteturais.

No Capítulo 6, intitulado "Considerações Finais", são apresentadas as conclusões, os resultados obtidos e contribuições, suas limitações e próximos passos que podem ser executados para dar continuidade ao trabalho apresentado nessa dissertação.

No Apêndice A, intitulado "Protocolo de Revisão Sistemática", estão descritas as informações que foram usadas como base para a realização da revisão sistemática das abordagens de avaliação arquitetural descritas na literatura, cujos resultados foram apresentados no Capítulo 2 dessa dissertação.

No Apêndice B, intitulado "*Checklist* para Inspeção Arquitetural", estão descritas as diferentes versões do *checklist* utilizadas durante os estudos de avaliação, assim como a sua versão final.

No Apêndice C, intitulado "Relatório de Discrepâncias", está descrito o relatório utilizado pelos inspetores na identificação de defeitos encontrados através do *checklist* de inspeção arquitetural que compõem ArqCheck.

No Apêndice D, intitulado "Plano do Experimento", está descrito o principal artefato gerado durante a fase de planejamento do estudo de observação, realizado para avaliar a viabilidade, a eficácia e o custo/eficiência da abordagem proposta.

No Apêndice E, intitulado "Questionário de avaliação pós-experimento", está descrito o questionário que foi utilizado no estudo de observação para coletar

informações que auxiliassem na análise qualitativa do objeto de estudo, no caso a abordagem proposta.

No Anexo A, intitulado "Notação para a Modelagem de Processo de Software", está descrito de forma resumida a notação que utilizamos nessa dissertação para modelar os processos de software nela descritos.

No Anexo B, intitulado "Táticas Arquiteturais", estão descritas as principais táticas arquiteturais que utilizamos como base para criar os itens de avaliação que compõem o *checklist* que compõem ArqCheck.

Capítulo 2 – Arquitetura de Software

Neste capítulo, são apresentadas algumas definições que permitem identificar o contexto de Arquitetura de Software em que a abordagem proposta nesta dissertação se aplica. Nele, procuramos responder, também, em que consiste uma arquitetura de software e qual benefício que pode ser obtido ao avaliá-la. Para isso, descrevemos os conceitos relacionados à arquitetura de software que utilizamos como base, identificamos a importância desse artefato dentro do processo de desenvolvimento de software e como ele é construído.

2.1 – Introdução

Quando tentamos solucionar um problema, é possível identificar diversas soluções que poderiam ser utilizadas visando resolvê-lo. Contudo, outros fatores como custo e eficiência influenciam na escolha da solução a ser adotada. No contexto do desenvolvimento de software, o mesmo pode ser observado ao se analisar os requisitos visando à construção de um software: várias soluções computacionais podem ser definidas para atender a esses requisitos, mas uma análise deve ser feita para definir a mais adequada ao contexto de desenvolvimento da aplicação.

Para se representar essas soluções, a arquitetura de software é uma das abordagens que podem ser usadas. Com isso, para se obter a arquitetura (solução) mais adequada para atender aos requisitos do software (problema), uma avaliação dessa estrutura deve ser realizada.

A arquitetura consiste em um modelo de alto nível que possibilita um entendimento e uma análise mais fácil do software a ser desenvolvido. O uso de arquitetura para representar soluções de software foi incentivada principalmente por duas tendências (GARLAN e PERRY, 1995; KAZMAN, 2001): (1) o reconhecimento por parte dos projetistas que o uso de abstrações facilita a visualização e o entendimento de certas propriedades do software, e (2) a exploração cada vez maior de *frameworks*[2] visando diminuir o esforço de construção de produtos através da integração de partes previamente desenvolvidas.

---

[2] De acordo com GAMMA *et al.* (1995), um *framework* consiste em uma arquitetura codificada para um domínio de problema que pode ser adaptada para resolver problemas específicos.

Uma outra propriedade da arquitetura é a possibilidade de usá-la como ferramenta para comunicar a solução projetada aos diversos *stakeholders* que participam do processo de desenvolvimento do software (GARLAN, 2000). Contudo, para que essa comunicação seja possível, a arquitetura deve ser representada através de um documento, conhecido como documento arquitetural.

Para se construir a arquitetura de um software, e por conseqüência o documento arquitetural que a representa, os requisitos são as principais informações usadas. Durante o processo de especificação arquitetural (Figura 2.1), além dos requisitos, outras fontes de conhecimento podem ser utilizadas para definir os elementos arquiteturais e a forma como eles devem estar organizados. Entre essas fontes de conhecimento se destacam principalmente a experiência do arquiteto, o raciocínio sobre os requisitos, e os estilos e as táticas arquiteturais.

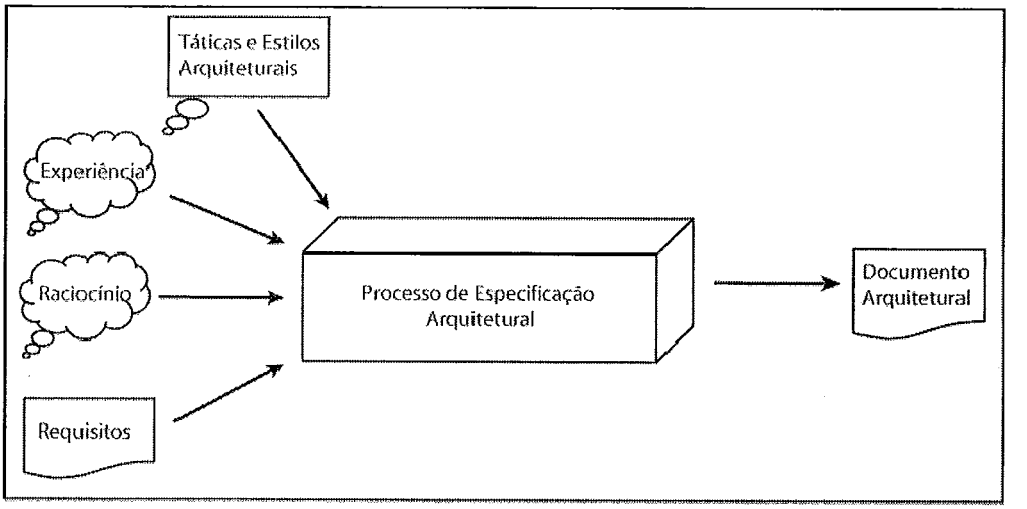


Figura 2.1 - Elementos usados na construção de uma arquitetura representados de acordo com a notação definida em (VILLELA, 2004)[3].

Contudo, existe uma falta de consenso na comunidade em relação tanto aos conceitos e definições básicas quanto à forma de representar uma arquitetura de software (BUSCHMANN *et al.*, 1996; CLEMENTS *et al.*, 2004). Portanto, na próxima seção, com o intuito de buscar a uniformização da terminologia utilizada nessa dissertação, são descritos os termos aqui adotados e seus respectivos conceitos associados. Além disso, na seção 2.3 são descritos a importância e o papel da arquitetura de software no processo de desenvolvimento, e na seção subseqüente são identificadas as principais atividades realizadas durante o processo de especificação arquitetural.

[3] A notação de modelagem de processo de software apresentada em (VILLELA, 2004) será usada ao longo dessa dissertação para representar os diversos processos que descrevemos. Para maiores informações sobre a notação vide Anexo A.

2.2 – Definição dos conceitos relacionados à arquitetura de software

Nessa seção, são definidos os termos utilizados neste trabalho, evitando ambigüidades, visto que terminologias inconsistentes sobre estes termos podem ser encontradas na literatura.

Arquitetura de software representa a estrutura, ou conjunto de estruturas, que compreende os elementos de software, suas propriedades externamente visíveis e seus relacionamentos (BASS *et al.*, 2003).

Para criar essa estrutura, grande parte dos autores concorda que três tipos de elementos básicos podem ser usados (DIAS e VIEIRA, 2000):

- Elementos de software, podendo também ser chamados de módulos ou componentes, são as abstrações responsáveis por representar as entidades que implementam funcionalidades especificadas;
- Conectores, podendo ser chamados de relacionamentos ou interfaces, são as abstrações responsáveis por representar as entidades que facilitam a comunicação entre os elementos de software;
- Organização ou configuração que consiste na forma como os elementos de software e conectores estão organizados.

Além disso, essa estrutura e as entidades que a compõem devem ser representados de uma forma que permita utilizar a arquitetura projetada para seus devidos fins, a essa representação é dado o nome de documento arquitetural. Esse documento é composto por um conjunto de modelos e informações, que descrevem principalmente a estrutura do software especificado para atender aos requisitos. Para compor um documento arquitetural, podemos nos basear, por exemplo, nas recomendações descritas no padrão IEEE-1471 (IEEE, 2000).

Contudo, mesmo existindo padrões que indicam o tipo de informação que deve ser descrito em um documento arquitetural, não é definido exatamente o nível de abstração que deve ser usado na descrição dessas informações.

A arquitetura de um software começa a ser construída nos estágios iniciais de um processo de desenvolvimento de software com o objetivo de definir e visualizar a solução computacional que será implementada. Neste momento, esse artefato é conhecido como arquitetura inicial, pertence ao escopo do problema, tem como principal característica descrever a solução em um elevado nível de abstração e é utilizado por vários *stakeholders* como base para tomada de decisões.

Contudo, ao longo do desenvolvimento do software, a arquitetura sofre refinamentos que diminuem o nível de abstração e permitem, por exemplo, a

representação dos relacionamentos entre os elementos arquiteturais e os arquivos de código fonte responsáveis por implementá-los (CLEMENTS *et al.*, 2004). Neste momento, a arquitetura passa a pertencer ao escopo da solução e incorpora também informações relacionadas às decisões de projeto, como elementos específicos à tecnologia que será usada para implementar a solução.

O fato da arquitetura representar informações em diferentes níveis de abstração ao longo do processo de desenvolvimento é um dos motivos que leva à falta de consenso na comunidade, pois ainda não se padronizou a granularidade que deve ser usada para descrever esse artefato.

No contexto desse trabalho, iremos trabalhar somente com a arquitetura inicial, ou seja, a que representa a estrutura em um elevado nível de abstração. Escolhemos trabalhar com esse artefato, pois a abordagem de avaliação que propomos nessa dissertação foi definida para inspecionar o documento arquitetural antes que a fase de projeto do software seja iniciada, visando evitar a propagação de defeitos. Nesse momento, somente a arquitetura inicial terá sido construída.

Além do mais, acreditamos que o uso de arquitetura para representar a solução em um baixo nível de abstração não é adequado devido à existência de diversos tipos de representação de projeto de baixo nível, como diagramas de classe e de seqüências, que permitem uma representação mais completa desse tipo de informação.

Contudo, qual o benefício em se avaliar esse artefato? Para responder a essa pergunta, é preciso identificar os papéis que a arquitetura possui no processo de desenvolvimento de software e os benefícios que podem ser obtidos ao avaliá-la, o que torna sua avaliação uma atividade extremamente desejável pelos *stakeholders*.

2.3 – Papel da arquitetura em um processo de desenvolvimento de software e os benefícios de sua avaliação

Ao revisar um artefato de software vários benefícios para o projeto e para a melhoria da qualidade do software podem ser obtidos. Contudo, para que essa atividade seja realizada, recursos devem ser alocados, o que pode aumentar o custo final do projeto.

Portanto, antes de realizar a revisão de um artefato, é imprescindível que a importância desse artefato dentro do processo de desenvolvimento seja identificada, permitindo definir o custo/benefício de sua revisão.

A principal motivação para avaliar a arquitetura de um software está relacionada ao seu papel dentro do processo de desenvolvimento.

Possuindo o documento arquitetural do sistema, os *stakeholders* podem utilizá-lo como artefato de entrada na realização de algumas atividades do processo ou então como base para tomada de decisões no contexto do projeto. Para cada *stakeholder*, a arquitetura do software é utilizada com diferentes propósitos (GACEK, 1995; XAVIER, 2001; CLEMENTS *et al.*, 2004):

- **Cliente.** O cliente é a pessoa ou empresa que contrata uma equipe de desenvolvimento para a construção de um sistema de sua necessidade. Na fase inicial do projeto, esse *stakeholder* necessita de uma estimativa de certos fatores, normalmente econômicos, que podem ser obtidos após a definição da estrutura principal do software. O cliente, por exemplo, tem interesse em estimativas de custo, confiabilidade e manutenibilidade do software que podem ser obtidos principalmente através de uma análise da arquitetura. Portanto, é de extrema importância para o cliente que a arquitetura atenda os requisitos do software de forma a representar suas reais expectativas em relação ao que foi especificado.
- **Gerentes**. A arquitetura permite aos gerentes tomarem certas decisões de projeto por possibilitar a sumarização das diversas características do sistema. Um gerente pode, por exemplo, usar a arquitetura como base para definir as equipes de desenvolvimento de acordo com os elementos arquiteturais que estão identificados na arquitetura e que devem ser construídos.
- **Desenvolvedor.** Da arquitetura de um software, o desenvolvedor busca uma especificação que descreva a solução com detalhes suficientes e que satisfaça os requisitos do cliente, mas que não seja tão restritiva a ponto de limitar a escolha das abordagens para a sua implementação. Os desenvolvedores usam a arquitetura como uma referência para a composição e o desenvolvimento dos elementos do sistema, e para a identificação e reutilização de elementos arquiteturais já construídos.
- **Testadores.** A arquitetura fornece, numa visão de caixa preta, informações aos testadores relacionadas ao correto comportamento dos elementos arquiteturais que se integram. Sendo assim, este artefato pode ser um dos artefatos bases utilizados durante o planejamento e execução de testes de integração e de sistema.
- **Mantenedor.** A descrição arquitetural do software fornece aos mantenedores uma estrutura central da aplicação que idealmente não deve ser violada. Qualquer mudança deve preservá-la, buscando, se possível, uma modificação puramente dos elementos arquiteturais e não da forma como estão organizados.

Visto como os principais *stakeholders* podem utilizar a arquitetura de um software, percebemos que o principal papel desse artefato é servir como instrumento para comunicar a solução proposta (GARLAN, 2000).

Sendo assim, o principal benefício em se avaliar um documento arquitetural está na diminuição das chances de um *stakeholder* utilizar um documento defeituoso nas atividades subseqüentes do processo de desenvolvimento de software.

Contudo, para permitir uma melhor compreensão sobre como e o que deve ser avaliado em um documento arquitetural, devemos primeiro entender como esse artefato é criado.

2.4 – Processo de especificação arquitetural

Existem na literatura diversas abordagens que objetivam a especificação de arquiteturas de software. Após avaliar algumas das principais abordagens (GACEK, 1995; SHAW e GARLAN, 1996; BOSCH e MOLIN, 1999; BACHMANN *et al.*, 2000; BASS *et al.*, 2003) identificamos um processo genérico de especificação arquitetural.

Esse processo é composto principalmente pelos seguintes elementos (Figura 2.2): duas macros atividades (projeto e avaliação arquitetural) e a tarefa de documentação da arquitetura. O que diferencia essas abordagens é principalmente a forma como cada um desses elementos são realizados.

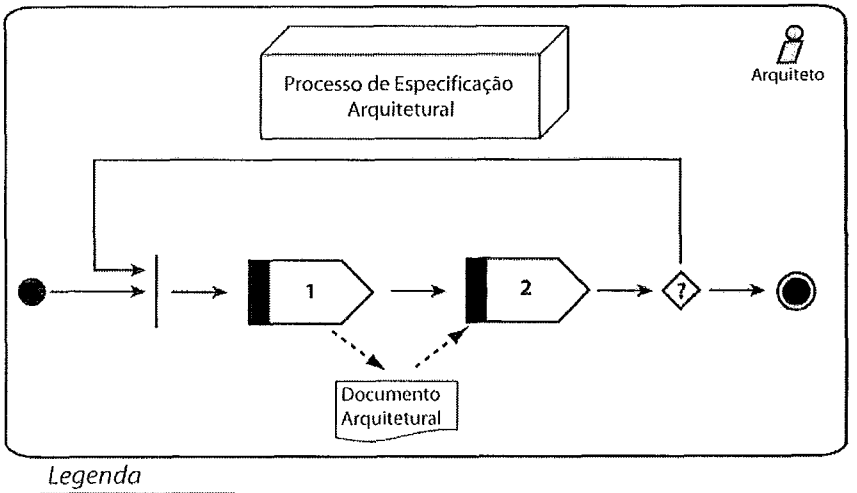


Figura 2.2 - Processo genérico de especificação arquitetural

Nesse processo, a característica comum às duas macro-atividades identificadas é a presença da tarefa de documentação responsável por criar e atualizar o documento que representa a arquitetura de software. Esse documento arquitetural é criado

durante a macro-atividade de projeto arquitetural e é responsável por registrar as decisões e os elementos arquiteturais.

Após identificarem que a solução descrita na arquitetura atende a todos os requisitos especificados, os arquitetos dão início à atividade de avaliação arquitetural que utiliza como principal artefato de entrada o documento arquitetural.

Após a avaliação, dependendo da qualidade do documento arquitetural e por conseqüência da arquitetura projetada, o arquiteto decide se o artefato será reavaliado visando atingir a qualidade desejada ou então se o processo de especificação arquitetural será finalizado.

A seguir, é mostrado para cada um dos elementos do processo de especificação arquitetural que tipo de informações e abordagens podem ser utilizadas para realizá-los.

2.4.1 – Projeto Arquitetural

O projeto arquitetural consiste na atividade em que a solução computacional e, por conseqüência, a arquitetura do software são definidas. Durante essa atividade, o raciocínio sobre os requisitos é realizado e decisões arquiteturais são tomadas, visando identificar e organizar os elementos arquiteturais para que os requisitos especificados possam ser atendidos.

Ao se analisar como essa atividade é realizada nas principais abordagens de especificação arquitetural, observamos a importância dos requisitos de qualidade no projeto de uma arquitetura e a existência de várias abordagens que podem ser utilizadas para atendê-los.

2.4.1.1 – Requisitos de Qualidade

Os requisitos de um software podem ser classificados como requisitos funcionais e os não-funcionais.

Os requisitos funcionais são responsáveis por descreverem as funcionalidades que o software deve apresentar. Já os não-funcionais descrevem características que o software deve apresentar, muitas vezes podem ser enxergadas como restrições ou especialidades do produto final. Os requisitos podem ter várias sub-categorias, como por exemplo, requisitos de qualidade, requisitos legais e etc.

Dentre os diferentes tipos de requisitos, tanto funcionais quanto não-funcionais, os requisitos de qualidade são os que mais influenciam na construção da arquitetura. Isso ocorre visto que, diferente dos requisitos funcionais onde na maioria dos casos uma modificação ocasiona alterações em um conjunto específico de elementos

arquiteturais, alterações em um requisito de qualidade podem implicar na total reestruturação da arquitetura (BASS *et al.*, 2003).

Contudo, nem todos os requisitos de qualidade são relevantes a nível arquitetural, pois determinados tipos de requisitos podem ser atendidos somente durante a etapa de codificação ou disponibilização (XAVIER, 2001). Um requisito de inteligibilidade, por exemplo, só poderá ser implementado no momento da definição da interface do sistema com o usuário.

Existem diferentes taxonomias para se classificar requisitos de qualidade (ISO/IEC, 1998; BASS *et al.*, 2003). No contexto desse trabalho, adotamos a taxonomia descrita por BASS *et al.* (2003) visto que ela identifica os tipos de requisitos de qualidade que são relevantes a nível arquitetural, ou seja, quais os tipos de requisitos de qualidade que influenciam na construção da arquitetura de um software.

Portanto, de acordo com BASS *et al.* (2003), esses tipos de requisitos são:

- **Desempenho**: Descrevem o comportamento do sistema em relação a restrições de tempo e de recurso computacional;
- **Disponibilidade**: Descrevem o comportamento de determinada parte do sistema em caso de falha;
- **Modificabilidade**: Descrevem quais as prováveis modificações que podem acontecer no sistema e as flexibilidades que devem estar nele presentes para que essas modificações sejam facilmente realizadas;
- **Segurança**: Descrevem o comportamento de determinada parte do sistema em relação ao acesso de seus dados ou funcionalidades;
- **Testabilidade**: Descrevem o comportamento de determinada parte do sistema em relação às facilidades que elas devem fornecer para a realização de testes;
- **Usabilidade**: Requisitos desse tipo, em um contexto arquitetural, descrevem facilidades que o sistema deve possuir, mas que não são consideradas funcionalidades do sistema. Exemplo dessas facilidades são operações de *undo* e *redo*.

2.4.1.2 – Atendendo os requisitos de qualidade

Durante o projeto de uma arquitetura, para atender aos requisitos de qualidade, as principais abordagens utilizam diversas fontes de conhecimento, tanto tácito quanto explícito para definir quais serão os elementos arquiteturais e com estarão organizados. Um exemplo de conhecimento tácito seria a experiência do arquiteto, e em relação ao conhecimento explícito teríamos os estilos e as táticas arquiteturais.

A experiência de um arquiteto é uma característica importante para o sucesso do projeto de uma arquitetura, pois a partir de suas lições aprendidas, o arquiteto

consegue facilmente identificar que elementos arquiteturais devem ser criados e como eles devem ser organizados. Mas, por ser um conhecimento tácito, é difícil de ser externalizado e utilizado por terceiros.

Por outro lado, estilos e táticas arquiteturais são conhecimentos explícitos amplamente difundidos na literatura e bastante utilizados por arquitetos de software (SHAW, 1995; BUSCHMANN *et al.*, 1996; BASS *et al.*, 2003).

Um estilo arquitetural, ou padrão arquitetural, consiste em um conhecimento que pode ser diretamente aplicado pelo arquiteto na identificação dos elementos arquiteturais. Isso é possível por ele ser composto por um conjunto de regras que permitem a identificação dos tipos de componentes e de conectores que serão usados na composição do software levando em conta as restrições impostas (SHAW e GARLAN, 1996).

Na literatura, existe um outro conceito, chamado de padrões de projeto, que é muito semelhante ao conceito de estilos arquiteturais. Em BUSCHMANN *et al.* (1996), é feita a diferenciação entre padrões arquiteturais e padrões de projeto. Essa diferença encontra-se principalmente no nível de abstração onde cada um desses padrões atua. Os padrões de projeto são utilizados somente durante a fase de definição do projeto de baixo-nível, onde refinamentos são feitos nos elementos arquiteturais que formam a arquitetura, e que foram definidos com base nos padrões arquiteturais. Contudo, muitos dos conceitos presentes em padrões arquiteturais e padrões de projeto são semelhantes, mas o que os diferencia é o fato de serem utilizados em níveis de abstração diferentes.

No contexto dessa dissertação, abordaremos somente padrões arquiteturais pois são eles que possuem os principais conceitos relevantes a nível arquitetural. Para evitar confusões, utilizaremos a denominação de estilos arquiteturais quando abordarmos esses conceitos.

Com isso, uma característica particular aos estilos arquiteturais é que o uso de um único estilo possibilita o atendimento a vários tipos de requisitos de qualidade. XAVIER (2001), por exemplo, descreve uma abordagem que, a partir dos tipos de requisitos de qualidade que devem ser atendidos pelo software, permite identificar os estilos arquiteturais mais adequados que devem ser usados na construção desse software.

Além dos estilos, um outro tipo de conhecimento explícito que pode ser utilizado no projeto arquitetural são as táticas arquiteturais. Uma tática arquitetural consiste em um conhecimento mais abstrato, utilizado principalmente para auxiliar o atendimento a um tipo de requisito de qualidade. Portanto, por serem mais abstratas, essas táticas descrevem principalmente possíveis características que uma arquitetura deve apresentar para atender a um determinado tipo de requisito.

Em BASS *et al.* (2003), essas táticas são identificadas e categorizadas em grupos, de acordo com os atributos de qualidade que elas influenciam.

Uma característica particular a essas táticas é que quando agrupadas e especializadas, podem ser usadas como base para a criação de estilos arquiteturais. ZHU (2004), por exemplo, realizou uma análise dos principais estilos arquiteturais por eles utilizados e identificou as táticas arquiteturais que os compõem.

Sendo assim, a partir do uso desse tipo de conhecimento, o arquiteto consegue definir a estrutura principal da arquitetura. Essa estrutura é em seguida povoada com elementos arquiteturais identificados principalmente a partir da análise dos requisitos funcionais.

2.4.2 – Documentação Arquitetural

Uma característica única em Engenharia de Software em relação às outras áreas de engenharia é que os produtos por ela construídos não são completamente materializáveis. Diferente de um engenheiro civil que pode inspecionar, por exemplo, as partes de um prédio, um engenheiro de software não consegue inspecionar um pedaço do software em si. Para isso ele deve utilizar representações desse software (LAITENBERGER e ATKINSON, 1999).

A arquitetura é um exemplo da parte de um software que não é materializável. Durante uma inspeção, por exemplo, é o documento arquitetural que deve ser revisado, por impossibilidade de se inspecionar diretamente a arquitetura projetada. Sendo assim, durante o seu projeto, a arquitetura tem que ser documentada para que ela possa ser usada para os seus devidos fins.

A arquitetura é uma entidade complexa que não pode ser descrita de uma forma unidimensional (CLEMENTS *et al.*, 2004). Uma forma efetiva de lidar com essa complexidade é descrevendo-a a partir de diferentes perspectivas, também conhecidas como visões arquiteturais.

Em cada visão, a forma como os elementos arquiteturais e seus relacionamentos são documentados coloca em evidência propriedades distintas do software que eles representam. De acordo com EGYED e MEDVIDOVIC (1999), ao criar uma visão arquitetural, os desenvolvedores conseguem reduzir a quantidade de informação que são obrigados a lidar em um determinado momento. Portanto, essas visões representam um aspecto parcial da arquitetura que mostram propriedades específicas do software.

Na Figura 2.3, podemos identificar três visões arquiteturais usadas para descrever um conjunto de elementos arquiteturais. Independente da notação gráfica utilizada, é possível notar as diferentes propriedades que cada visão permite identificar.

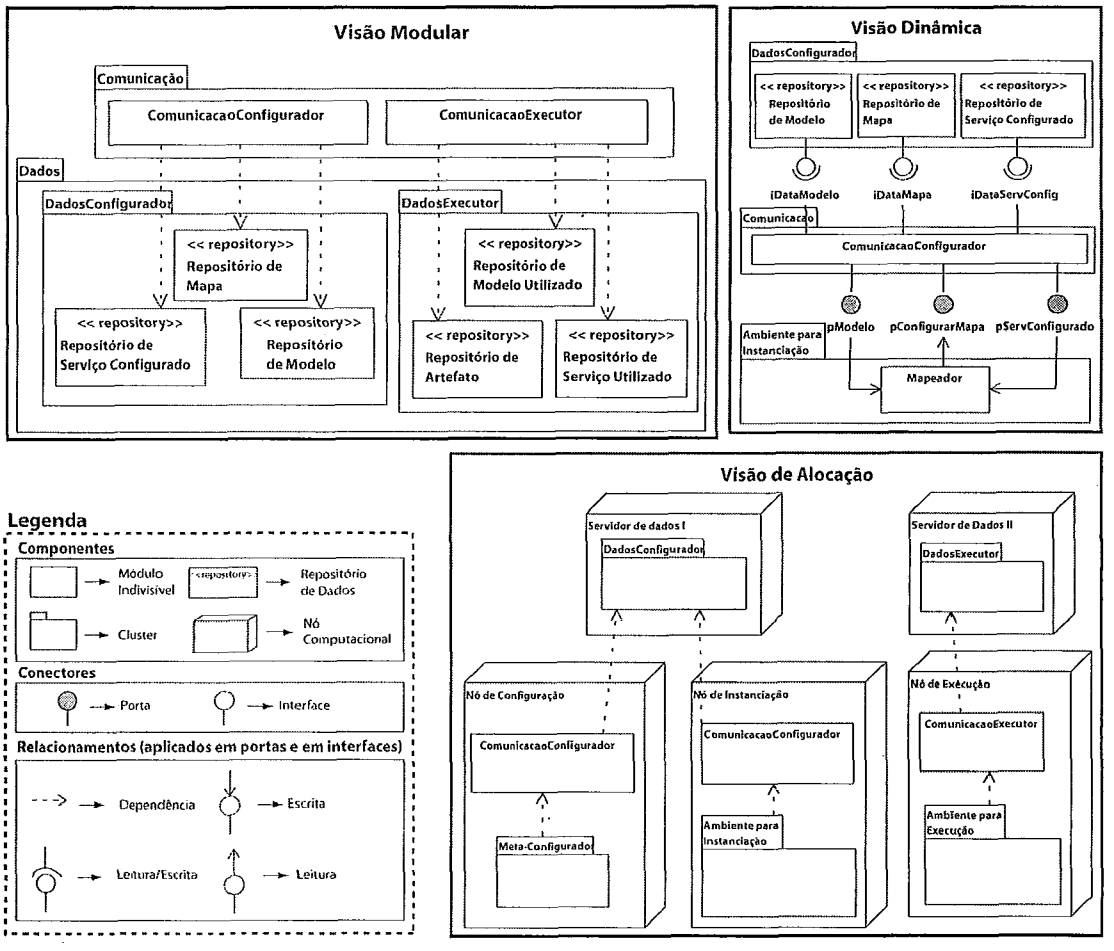


Figura 2.3 – Exemplo de visões arquiteturais

Existe um grande número de visões arquiteturais propostas na literatura que propõem soluções similares para a representação de uma arquitetura (KRUCHTEN, 1995; HOFMEISTER *et al.*, 2000; CLEMENTS *et al.*, 2004). As principais visões são:

- **Visão Modular**: Esta perspectiva representa os elementos que compõem a arquitetura, responsáveis por realizar um conjunto de funcionalidades, e as dependências entre eles. Para isso, um conjunto de diagramas pode ser criado para representar através de diferentes níveis de abstração, os elementos, seus elementos internos (caso haja) e como eles se relacionam entre si.
- **Visão Dinâmica**: Esta perspectiva procura descrever o comportamento dos elementos arquiteturais durante a realização dos diferentes fluxos de execução que pertencem ao sistema.

- **Visão de Alocação**: Esta perspectiva busca representar o mapeamento das unidades de software para elementos físicos do ambiente (hardware, arquivos do sistema, equipe de desenvolvimento).
- **Visão de contexto geral:** Essa perspectiva tem como objetivo representar uma visão geral dos principais componentes que formam a arquitetura do software e de como ele se relaciona com os elementos externos ao seu contexto (atores e sistemas externos).

A escolha das visões a serem documentadas deve ser feita com base nas características de qualidade que se deseja por em evidência, uma vez que diferentes visões expõem características de qualidade distintas.

Para CLEMENTS *et al.* (2004), documentar uma arquitetura consiste em documentar as visões arquiteturais relevantes, explicar como essas visões se relacionam e como um *stakeholder* deve utilizar esse material.

No contexto dessa dissertação, utilizamos algumas das recomendações definidas pelo padrão IEEE-1471, que abordam a descrição arquitetural de sistemas de software, para definir as principais informações que devem ser descritas em um documento arquitetural. Sendo assim, um documento arquitetural deve:

- Identificar os elementos arquiteturais que compõem a solução a ser construída, assim como a forma que esses elementos estão organizados;
- Descrever o papel de cada elemento dentro da arquitetura;
- Identificar como cada requisito relevante a nível arquitetural está sendo atendido através da arquitetura documentada. Essa identificação pode ser feita principalmente através do rastreamento de que requisito está sendo atendido e quais requisitos justificam a criação de determinado elemento arquitetural;

Representar o software através de diferentes perspectivas, como por exemplo, através do uso de visões arquiteturais.

2.4.3 – Avaliação Arquitetural

A avaliação arquitetural consiste em caracterizar e avaliar os documentos arquiteturais através de métodos ou procedimentos sistemáticos (BAHSOON e EMMERICH, 2003). Essa avaliação verifica principalmente se as informações descritas no documento estão consistentes e se a arquitetura nele representada atende aos requisitos especificados para o produto.

Visto que são os requisitos de qualidade os que mais influenciam a construção de uma arquitetura, portanto, é principalmente sob a perspectiva desse tipo de requisitos

que a avaliação deve ser realizada (DOBRICA e NIEMELA, 2002; BABAR *et al.*, 2004).

A realização da atividade de avaliação é de extrema importância para a melhoria da qualidade do produto de software e para o sucesso do projeto. Esta afirmação é fortalecida se for considerado que (1) a avaliação da arquitetura impede que seus defeitos se propaguem para os demais artefatos, como diagramas de projeto e código fonte, e (2) o custo de correção desses defeitos é bem menor se for realizada durante os primeiros estágios do projeto (BOEHM, 1981).

Além dos benefícios listados anteriormente, MARANZANO *et al.* (MARANZANO *et al.*, 2005) identificaram os seguintes benefícios, após aplicar a avaliação arquitetural em diversos projetos no contexto da empresa em que trabalha, que podem ser obtidos através dessa prática:

- Permite um melhor aproveitamento do conhecimento de seus especialistas, pois são alocados em avaliações arquiteturais que analisam arquiteturas de projetos em que não tiveram participação, utilizando assim suas experiências e conhecimentos para auxiliá-los;
- Permite um melhor gerenciamento dos fornecedores de componentes de software da empresa;
- Permite que a alta gerência tenha uma maior compreensão de problemas, principalmente de ordem técnica, que ocorrem durante a gerência dos projetos da empresa;
- Possibilite a identificação de necessidades de treinamentos ao nível de projeto ou organizacional com base em tipos de problema freqüentemente identificados durante as avaliações. Por exemplo, fornecer cursos em otimização de sistemas quando as avaliações identificarem principalmente problemas arquiteturais relacionados à característica de desempenho.

A avaliação de documentos arquiteturais é um tema que tem sido bastante discutido no contexto de vários grupos de pesquisa, como no grupo do *Software Engineering Institute* (SEI) (KAZMAN *et al.*, 1994; CLEMENTS *et al.*, 2002), por exemplo. Portanto, existem diversas abordagens que realizam essa avaliação visando diferentes objetivos ou utilizando diferentes técnicas. No capítulo 3 dessa dissertação, os principais métodos são identificados e caracterizados.

2.5 – Conclusão

Ao longo deste capítulo, foram descritos os principais conceitos em relação à arquitetura de software, dando ênfase principalmente nas atividades que estão relacionadas ao seu processo de especificação.

Através da análise desses conceitos e processos, foi possível identificar (1) a importância da arquitetura dentro do processo de desenvolvimento de software, (2) como esse artefato é construído e principalmente (3) que informações devem estar representadas nesse artefato e que devem ser analisadas durante o processo de avaliação para que se determine a corretude do documento arquitetural.

Sendo assim, dando continuidade à metodologia que foi definida para realizar esse trabalho, o segundo passo a ser realizado consiste em identificar as abordagens de avaliação existentes com o objetivo de observar suas características, pontos positivos e limitações. No capítulo seguinte, intitulado "Abordagens de avaliação arquitetural", descrevemos os resultados obtidos neste contexto.

Capítulo 3 – Abordagens de avaliação arquitetural

Neste capítulo, as principais abordagens de avaliação arquitetural descritas na literatura técnica são identificadas e caracterizadas. Além disso, descrevemos como essa identificação foi feita e os resultados obtidos após a análise dessas abordagens. São esses resultados que motivaram a criação da abordagem de avaliação proposta nessa dissertação.

3.1 – Introdução

Sabemos que a avaliação de um documento arquitetural não permite medir com exatidão as características de qualidade do produto de software final (BOSCH, 2000). Isso ocorre devido ao fato do documento arquitetural avaliado não representar todas as informações que irão compor o projeto detalhado e a implementação, artefatos gerados em atividades posteriores, e também devido ao fato dessa avaliação ser realizada normalmente sobre a arquitetura inicial do sistema, artefato que não engloba diversas decisões de projeto, como por exemplo, decisões ligadas a questões tecnológicas.

O principal objetivo de uma avaliação arquitetural consiste em verificar se as informações descritas no documento estão consistentes e se a arquitetura nele representada atende aos requisitos especificados para o software. Portanto, esse tipo de verificação possibilita principalmente a diminuição da propagação dos defeitos contidos nesse documento para as fases subseqüentes que o utilizam como artefato de entrada.

Contudo, o que se pode dizer sobre avaliação de documentos arquiteturais? É possível identificar abordagens que realizam esse tipo de avaliação? É possível caracterizar como tais abordagens realizam essa avaliação?

A resposta a esses questionamentos é extremamente importante pois permite entender como se caracteriza o estado da arte dessa área. A partir desse entendimento, é também possível identificar benefícios e possíveis limitações das abordagens existentes.

Visto que o nosso objetivo é definir uma abordagem de avaliação de documentos arquiteturais, a identificação das abordagens de avaliação arquitetural existentes e a caracterização de como eles avaliam um documento arquitetural é uma das etapas que definimos para atingir o nosso objetivo.

Portanto, realizamos uma revisão sistemática (KITCHENHAM, 2004; BIOLCHINI *et al.*, 2005) para identificar as principais abordagens de avaliação arquitetural descritas na literatura técnica e caracterizamos essas abordagens visando entendê-las.

Nas seções a seguir, é descrito em que consiste uma revisão sistemática, como foi a condução desse tipo de revisão visando identificar abordagens de avaliação arquitetural e quais foram os resultados obtidos após a análise das abordagens identificadas.

3.2 – Revisão Sistemática

A avaliação de documentos arquiteturais é um tema que tem sido bastante discutido no contexto de diversos grupos de pesquisa, como no grupo do *Software Engineering Institute* (SEI) (KAZMAN *et al.*, 1994; CLEMENTS *et al.*, 2002), por exemplo. Contudo, não foram encontrados registros na literatura técnica de revisões formais que tenham sido realizadas visando à identificação das abordagens de avaliação arquitetural.

As revisões conduzidas anteriormente (CLEMENTS *et al.*, 2002; DOBRICA e NIEMELA, 2002; BAHSOON e EMMERICH, 2003; BABAR *et al.*, 2004) não descrevem o escopo em que as buscas foram executadas nem os critérios de seleção utilizados. Sendo assim, essas revisões não podem ser auditadas, não são passíveis de serem repetidas e a obtenção dos resultados depende dos revisores que a executaram, aumentando assim a possibilidade de viés nos resultados (MAFRA e TRAVASSOS, 2005).

Para evitar as limitações observadas nas revisões existentes, optamos por realizar uma revisão sistemática que colete, em algumas das principais fontes da comunidade de Engenharia de Software, estudos que abordem o tema de avaliação arquitetural e permitam a identificação das abordagens de avaliação arquitetural existentes.

3.2.1 – Definição

Normalmente, para se estudar uma nova área de conhecimento, pesquisadores costumam utilizar a revisão bibliográfica como abordagem de busca por publicações. Contudo, essa abordagem não é realizada de forma sistemática e não fornece nenhum apoio que permita evitar viés na escolha das publicações que serão selecionadas para serem analisadas, por exemplo. Uma forma de sistematizar a identificação e análise de publicações, e estudos primários em geral, é através de revisões sistemáticas.

Uma revisão sistemática é um meio de identificar, avaliar e interpretar os resultados de pesquisas disponíveis e relevantes a um conjunto de critérios, de acordo

com uma pergunta de pesquisa, uma área específica ou um fenômeno de interesse (KITCHENHAM, 2004). Essa abordagem segue uma seqüência precisa e bem definida de passos, de acordo com um protocolo de pesquisa previamente planejado.

Os estudos identificados por uma revisão sistemática são classificados como estudos primários. No contexto deste trabalho, esses estudos são publicações científicas. A revisão sistemática em si é considerada um estudo secundário (KITCHENHAM, 2004), uma vez que ela depende dos resultados dos estudos primários para ser conduzida.

KTCHENHAM (2004) descreve vários motivos para a realização de uma revisão sistemática, entre os principais temos:

- Para sumarizar evidências relacionadas a uma área de conhecimento ou a uma tecnologia. Como por exemplo, sumarizar evidências experimentais sobre os benefícios e limitações de determinado método;
- Identificar falhas em uma pesquisa existente visando sugerir que tópicos deveriam ser investigados com mais rigor;
- Caracterizar determinadas áreas do conhecimento visando propor atividades de pesquisa que poderiam ser realizadas para evoluir essas áreas.

No contexto deste trabalho, realizamos esse tipo de revisão com o intuito de sumarizar conhecimento e identificar evidências em relação às abordagens de avaliação arquitetural identificadas, permitindo a investigação de seus benefícios e limitações.

Uma revisão sistemática se destaca por ser executada através de um processo de condução. Além disso, para que seja possível uma posterior auditoria e repetição, as informações relacionadas aos passos executados durante essa revisão devem ser avaliadas e registradas em um documento conhecido como protocolo de revisão sistemática (KITCHENHAM, 2004; BIOLCHINI *et al.*, 2005).

3.2.2 – Processo de condução de uma revisão sistemática

O uso de um processo de condução é extremamente importante por permitir uma sistematização dos passos que devem ser realizados. Um dos processos que pode ser utilizado na condução de uma revisão sistemática é o descrito em (BIOLCHINI *et al.*, 2005). Este processo é composto pelos seguintes passos:

- **Planejamento da revisão**: Durante esse passo, os objetivos da pesquisa devem ser definidos através da especificação do que será pesquisado, dos locais onde a pesquisa será realizada e dos critérios utilizados para definir se um estudo identificado é válido ou não para a pesquisa. No final dele, uma versão refinada do

protocolo é criada e a viabilidade em realizar o que foi descrito nesse protocolo tem que ser avaliada.

- **Execução da revisão**: Durante esse passo, é feita a identificação dos estudos que estejam relacionados ao objetivo da pesquisa e que satisfaçam os critérios de seleção definidos. Essa identificação é realizada através de buscas, usando palavras chave previamente definidas, que devem ser executadas nas fontes de pesquisa que foram previamente definidas. As publicações identificadas são filtradas nesse momento, de acordo com os critérios de inclusão e exclusão.
- **Análise dos resultados**: Durante esse passo, é feita a análise das publicações selecionadas visando extrair informações para responder às questões de pesquisa.

3.3 – Conduzindo uma revisão sistemática para identificar métodos de avaliação arquitetural

A revisão sistemática realizada no contexto dessa dissertação foi planejada e conduzida com base nos conceitos descritos em (BIOLCHINI *et al.*, 2005). O objetivo desse estudo consiste em:

Tabela 3.1 - Objetivos do estudo seguindo o método GQM (BASILI *et al.*, 1994)

Analisar abordagens que avaliam documentos arquiteturais
Com o propósito de caracterizar
Com respeito às características presentes em técnicas de inspeção de software
Do ponto de vista dos inspetores de artefatos de software
No contexto dos estudos primários que descrevem essas abordagens

A seguir, são mostradas as principais informações relacionadas à execução dos passos dessa revisão sistemática.

3.3.1 – Planejamento da revisão sistemática

Durante o planejamento dessa revisão, um protocolo foi criado visando registrar todas as decisões tomadas em relação à sua realização. Esse protocolo seguiu o template descrito em (BIOLCHINI *et al.*, 2005). Na Tabela 3.2, estão descritas as principais informações que compõem o protocolo, para uma versão completa desse artefato veja o Apêndice A.

Tabela 3.2 - Principais informações descritas no protocolo da revisão sistemática.

Foco da pergunta de pesquisa: Identificar e caracterizar abordagens utilizadas para avaliar a qualidade de documentos arquiteturais;

Pergunta: De que forma arquiteturas de software e documentos arquiteturais são avaliados? Quais as abordagens e em que contexto elas são utilizadas?

Contexto: Essas abordagens de avaliação devem ter sido executadas em projetos de desenvolvimento de software que (1) possuam seus requisitos bem definidos, (2) possuam atividades que apóiem a construção da arquitetura do software a ser implementado e (3) utilizam abordagens de avaliação para determinar se os atributos de qualidade desejados foram respeitados.

Palavras-chave[4]: "software architecture", "software modularity", "software component", "architectural model", "high level design", "high level model", "evaluate", "assurance", "review", "inspection", "verification", "analysis";

Critérios de seleção de fontes: Disponibilidade de consulta através da web; possuírem um nível de avaliação Qualis Capes superior a "C";

Listagem de fontes: Portal IEEE - IEEE Xplore (IEEE, 2004), Portal da ACM (ACM, 2004), Relatórios técnicos do SEI (SEI, 2004), Livros da área (Documenting Software Architecture (CLEMENTS *et al.*, 2004), Evaluating Software Architecture (CLEMENTS *et al.*, 2002) e Software Architecture in Practice, Second Edition (BASS *et al.*, 2003)).

Critérios de inclusão e exclusão dos estudos: Os artigos devem: estar disponíveis no momento da pesquisa; descrever ou apresentar alguma abordagem que avalie a arquitetura de um software ou seus modelos;

3.3.2 – Execução da revisão sistemática

A execução de uma revisão sistemática consiste na realização de duas atividades. Em um primeiro momento, é feita a identificação dos estudos científicos. Essa identificação é realizada através de buscas nas máquinas de buscas das fontes selecionadas, usando as palavras chave que foram definidas.

Após isso, para cada estudo identificado, é feita uma análise dos artigos visando avaliá-los e selecioná-los de acordo com os critérios de exclusão e inclusão definidos no protocolo. Nas revisões por nós realizadas, essa análise foi feita através da leitura das seções de resumo e de introdução, e o critério utilizado foi selecionar as publicações que permitissem identificar alguma abordagem de avaliação arquitetural.

No contexto desse trabalho, duas execuções do protocolo da revisão sistemática foram realizadas (Tabela 3.3). A primeira execução foi realizada durante o mês de Dezembro de 2004 e 80 estudos científicos foram por ela identificados. Contudo, somente 26 foram selecionados por atenderem aos critérios de inclusão e exclusão definidos no protocolo.

[4] Conjunto de palavras que foram usadas como base para a criação das *strings* de busca nas diferentes máquinas de busca.

Durante a realização da primeira atividade de execução dessa revisão, tivemos problemas com a máquina de busca da ACM (ACM), relacionados principalmente a inconsistências observadas nos resultados obtidos após a execução de uma busca, o que impossibilitaria a obtenção dos mesmos resultados quando essa busca fosse repetida. Visto que uma das principais características da revisão sistemática é permitir que a obtenção dos resultados seja repetível de forma independente dos pesquisadores que a realiza, decidimos eliminar essa fonte de pesquisa, assim como as publicações através dela identificadas.

Tabela 3.3 – Quantidade de publicações identificadas e selecionadas em cada revisão

Revisões	Qtd. Identificados	Qtd. Selecionados
Dezembro 2004 (IEEE Xplore)	80	26
Dezembro 2005 (IEEE Xplore + ACM)	119	38

Todavia, em Dezembro de 2005, a revisão sistemática foi re-executada. Optamos por re-executá-la por dois motivos: (1) identificar novas publicações e abordagens arquiteturais que tenham surgido durante esse intervalo de tempo e (2) verificar a viabilidade de se identificar publicações na máquina de busca da ACM, que sofreu evoluções desde a última revisão que realizamos.

Além da inclusão da biblioteca da ACM como fonte de pesquisa, o protocolo de revisão sofreu evoluções após a análise dos resultados obtidos na primeira execução. A principal modificação ocorreu nas palavras chave, que são usadas para criar as strings de busca utilizadas na identificação das publicações nas fontes de pesquisa.

Ao observar os estudos identificados na primeira execução do protocolo, percebemos que algumas abordagens de avaliação arquitetural eram classificadas também como abordagens de análise arquitetural. Sendo assim, a palavra "analysis" foi incluída na lista de palavras chave, o que permitiu a identificação de publicações que utilizam essa taxonomia durante a busca.

Como resultado dessa segunda revisão, identificamos 119 publicações relevantes e após a aplicação dos critérios de inclusão e exclusão 38 foram selecionados. Ao fazer uma análise dos artigos selecionados percebemos que esse novo conjunto contém todo o conjunto de artigos selecionados na primeira revisão e mais os publicados em 2005 e os que eram exclusivos à biblioteca da ACM, eliminada na primeira revisão.

Sendo assim, a partir de uma análise mais completa dos estudos selecionados, 27 abordagens de avaliação arquitetural foram identificadas.

3.4 – Abordagens de avaliação arquitetural identificadas

Para que seja possível ter um melhor entendimento e permitir uma comparação entre essas 27 abordagens de avaliação identificadas, é necessário que cada uma seja caracterizada.

O principal objetivo das abordagens de avaliação identificadas é a melhoria da qualidade da arquitetura de software. Visto que resultados de estudos experimentais têm mostrado os benefícios que a utilização de técnicas de inspeção aportam à melhoria da qualidade de artefatos de software (SHULL *et al.*, 2000; CONRADI *et al.*, 2003), optamos por usar um conjunto de características normalmente presentes nesse tipo de técnicas para definir um *framework* que permita caracterizar as abordagens de avaliação arquitetural.

Nas subseções abaixo, descrevemos os conceitos básicos de inspeção de software, identificamos cada uma das características que compõem essa abordagem de revisão e justificamos a importância em utilizá-las para caracterizar as abordagens de avaliação arquitetural. Em seguida, descrevemos sucintamente cada abordagem identificada e os resultados de sua caracterização.

3.4.1 – Inspeção de Software

Inspeção de software (FAGAN, 1976) é um método rigoroso e formal executado com o objetivo de examinar artefatos de software. Esse tipo particular de revisão possui como característica principal um processo de detecção de defeitos rigoroso e bem definido.

Para caracterizar uma abordagem de inspeção, compreender o seu funcionamento e sua influência no ambiente em que ela é aplicada, é necessário avaliar as diferentes dimensões que formam essa abordagem. LAITENBERGER E DEBAUD (1998) identificaram, a partir de um survey realizado na literatura, cinco dimensões que podem ser utilizadas para caracterizar a natureza de uma abordagem de inspeção de software:

- **Dimensão Técnica:** usada para caracterizar diferentes abordagens de inspeção através da identificação de similaridades e diferenças;
- **Dimensão Gerencial:** usada para prover informações dos efeitos da abordagem de inspeção no projeto e vice versa;
- **Dimensão Organizacional:** usada para prover informações sobre a influência da abordagem no contexto de uma organização e vice versa;
- **Dimensão Avaliação:** usada para comparar o custo / benefício de uma abordagem em uma determinada situação;

- **Dimensão Apoio Ferramental:** usada para prover informações sobre como a abordagem pode ser apoiada por ferramentas.

3.4.2 – Critérios usados para caracterizar as abordagens de avaliação arquitetural

No contexto dessa dissertação, visto que objetivamos caracterizar as abordagens identificadas pela revisão sistemática e identificar similaridades e diferenças entre elas, utilizamos os conceitos pertencentes à dimensão técnica (LAITENBERGER e DEBAUD, 1998) como base para identificar alguns critérios para a nossa caracterização (Tabela 3.4).

Além dos conceitos presentes na dimensão técnica, decidimos adicionar ao *framework* de caracterização alguns critérios presentes em outros *frameworks* descritos na literatura técnica (DOBRICA e NIEMELA, 2002; BABAR *et al.*, 2004) e que achamos relevantes para atingir o nosso propósito. Esses critérios adicionais são: o momento do processo de especificação arquitetural que a avaliação é realizada, a existência de alguma ferramenta que auxilie essa avaliação e a existência de estudos que avaliem a abordagem.

Tabela 3.4 – Questionamentos que buscamos responder a partir de cada critério

Critérios	Questionamentos
Tipo de arquitetura avaliada	Que tipo de informação é usado para descrever a arquitetura a ser avaliada? Qual o nível de abstração utilizado?
Tipo de técnica usada na avaliação	Como as discrepâncias são encontradas?
Processo seguido durante a avaliação	Quais os passos que são necessários para realizar essa avaliação?
Artefatos requeridos	Que artefatos devem ser disponibilizados para os envolvidos na avaliação?
Artefatos produzidos	Que artefatos são gerados por essa avaliação?
Características de qualidade avaliadas	Que características cada abordagem busca avaliar?
Momento da avaliação	Em que momento do seu processo de especificação, a arquitetura é avaliada?
Apoio ferramental	A abordagem possui algum apoio ferramental para a sua realização?
Avaliação da abordagem	A abordagem teve sua viabilidade e utilidade avaliadas?

A seguir, justificamos a importância em utilizar cada um desses critérios na caracterização de abordagens de avaliação arquitetural.

3.4.2.1 – Tipo de arquitetura avaliada

Uma das características específicas a uma abordagem de avaliação é o tipo de artefato que ela avalia.

A princípio, no contexto de avaliação arquitetural, todas as abordagens identificadas avaliam arquiteturas de software. Porém, na área de Arquitetura de

Software não existe um consenso em relação ao que é uma arquitetura de software, mais especificamente ao conteúdo do documento que a descreve. Sendo assim, ao caracterizar uma abordagem através desse critério, buscamos identificar o que exatamente é avaliado pela abordagem.

Com isso, definimos que o tipo de arquitetura avaliada deve ser caracterizado através dos tipos de informação usados para descrever a arquitetura (representações gráficas, descrições textuais, rastreabilidade com os requisitos, decisões arquiteturais, descrição usando diferentes perspectivas de representação) e o nível de granularidade usado para descrever essas informações (arquitetura inicial ou refinada).

3.4.2.2 – Tipo de técnica usada na avaliação

Uma forma de caracterizar abordagens de avaliação arquitetural consiste em identificar as técnicas ou tipos de métodos que são utilizados para realizar essa avaliação (ABOWD *et al.*, 1997).

Identificar a técnica utilizada por uma abordagem é de extrema importância, pois é ela que define como o avaliador irá realizar a avaliação do artefato e, portanto, que procedimentos ele deve seguir para atingir o seu objetivo.

No contexto de avaliação arquitetural, ABOWD *et al.* (1997) identificou três tipos de técnicas de avaliação que podem ser usadas: técnicas de questionamento, de medição e híbridas.

Técnicas de questionamento

Diferente das técnicas de medição, que requer a existência de algum tipo de "artefato executável" para que elas possam ser utilizadas, as técnicas de questionamento podem ser usadas para investigar qualquer parte da arquitetura, não importando o estado que ela se encontre (CLEMENTS *et al.*, 2002).

Esse tipo de técnica é utilizado principalmente quando se possui um conjunto de questionamentos e se deseja aplicá-los visando observar como eles são respondidos a partir da arquitetura selecionada, ou seja, ela busca revisar a arquitetura de software a partir de critérios pré-definidos. Esse tipo de avaliação é também conhecido como revisão de software e é uma das possíveis abordagens usadas para garantir a qualidade de artefatos de software.

As principais técnicas que se enquadram nessa categoria são principalmente as que fazem uso de cenários, questionários e *checklists*.

Técnicas de medição

Em vez de prover meios para gerar os questionamentos que serão feitos durante a avaliação, esse tipo de técnica busca por respostas que os membros da equipe de avaliação já formularam em relação a uma determinada característica da arquitetura. Contudo, esse tipo de técnica necessita a presença de artefato implementacional sobre o qual as medições podem ser feitas.

As principais abordagens que utilizam esse tipo de técnica são aquelas que fazem uso de métricas, simulação, protótipos ou experimentação sobre os sistemas derivados da arquitetura que está sendo avaliada.

Técnicas híbridas

Muitas abordagens de avaliação fazem o uso simultâneo de técnicas de medição e de questionamento, ficando assim conhecidos como técnicas híbridas.

Essa abordagem é utilizada quando alguma técnica de medição é utilizada para responder a questionamentos levantados por técnicas de questionamento (CLEMENTS *et al.*, 2002).

3.4.2.3 – Processo seguido durante a avaliação

Processos são importantes porque impõem consistência e estrutura a um conjunto de atividades, guiam ações e permitem examinar, entender, controlar e melhorar as atividades que o compõem (PFLEEGER, 2004).

A presença de um processo para uma abordagem de avaliação permite a sistematização do método e é uma forma de identificar que atividades da abordagem de avaliação devem ser realizadas para que os resultados planejados sejam obtidos.

Nessa caracterização, pretendemos identificar principalmente quais são as atividades que devem ser realizadas para executar a abordagem. Com base nessa informação, é possível se ter uma idéia dos custos envolvidos nessa avaliação.

3.4.2.4 – Artefatos requeridos e artefatos produzidos pela avaliação

Além do documento arquitetural, outros artefatos são também necessários para que uma avaliação seja realizada. Além disso, devido aos diferentes objetivos que levam a realização de uma avaliação arquitetural, os artefatos produzidos nem sempre consistem em uma lista de discrepâncias, por exemplo.

Procuramos então identificar nas abordagens quais são os artefatos que eles necessitam e produzem. Isso é feito pois a partir deles é possível identificar, entre outras coisas, o contexto em que essas abordagens de avaliação arquitetural podem ser aplicadas.

3.4.2.5 – Características de qualidade avaliadas

A arquitetura de um software é criada com base nos seus requisitos, principalmente os requisitos de qualidade. Devido a importância desse tipo de requisito para o projeto arquitetural, um dos principais objetivos de uma abordagem de avaliação arquitetural é identificar defeitos relacionados ao não atendimento de requisitos de qualidade.

Devido à existência de vários tipos de requisitos de qualidade, procuramos identificar através desse critério, que características de qualidade cada abordagem de avaliação procura avaliar e por conseqüência identificar defeitos nos tipos de requisitos relacionados.

3.4.2.6 – Momento da avaliação

A qualidade de um software não pode ser imposta depois que o produto estiver finalizado, portanto a sua qualidade deve ser um aspecto que deve ser tratado simultaneamente à execução do processo de desenvolvimento do software (CLEMENTS *et al.*, 2002).

Visto que a arquitetura auxilia no desenvolvimento do software, é importante que ela tenha a sua qualidade avaliada antes do início da implementação do software. Além disso, principalmente para a reengenharia de sistemas legados, é interessante também que uma avaliação arquitetural seja realizada visando caracterizar a arquitetura desse sistema.

Portanto, visando entender melhor cada abordagem, procuramos caracterizar cada abordagem a partir do momento em que ela é executada. Sendo assim, definimos que uma avaliação arquitetural pode ser realizada nos seguintes momentos:

Durante construção

Avaliações podem ser realizadas durante a execução do processo de especificação da arquitetura, quando nem todos os elementos arquiteturais tiverem sido gerados. Nesse caso, a avaliação auxilia na construção do restante dos elementos.

Após construção

Após a finalização da construção da arquitetura, a arquitetura de um software pode ser avaliada com os seguintes propósitos: (1) avaliar o atendimento a todos os requisitos de qualidade, para que se possa decidir se o processo de especificação deve ser finalizado ou não; ou (2) entender a arquitetura de um sistema já desenvolvido durante a realização de uma reengenharia ou manutenção do software.

3.4.2.7 – Avaliação da abordagem

De acordo com (BABAR *et al.*, 2004), desenvolver uma abordagem de avaliação baseada em experiências pessoais, observações gerais, heurísticas de especialistas, ou boas práticas publicadas não é uma boa forma de garantir a viabilidade e a utilidade da abordagem. O criador de uma abordagem pode avaliar o seu trabalho através de diversas técnicas experimentais disponibilizadas no domínio de Engenharia de Software.

Através desse critério, procuramos então identificar se cada abordagem foi realmente avaliada. Estamos interessados em identificar principalmente que tipo de estudos foi realizado.

No contexto dessa caracterização, classificamos a forma como uma abordagem foi avaliada através da seguinte taxonomia:

Estudo experimental

Um estudo experimental é um possível tipo de estudo que pode ser usado para avaliar uma abordagem de avaliação arquitetural.

Esse tipo de estudo consiste em observar de forma sistemática o efeito de uma tecnologia em aplicações práticas (CIOLKOWSKI *et al.*, 2002). Ele se caracteriza principalmente pela formalização empregada na sua execução e possui como principal objetivo obter respostas a questionamentos específicos que foram formalizados antes de sua realização.

O mais comum tipo de estudo experimental é o estudo de caso. No nosso contexto, consideramos estudos de caso um estudo experimental que tem como objetivo aplicar a tecnologia avaliada em um contexto de desenvolvimento de software real (SHULL *et al.*, 2001).

Prova de conceito

Estudo que foi realizado visando mostrar a viabilidade de se aplicar a abordagem de avaliação proposta. Geralmente, esse tipo de estudo é realizado através da descrição do funcionamento da abordagem em um determinado exemplo, que pode ser real ou não.

A principal característica desse tipo de estudo é que o autor não utilizou conceitos relacionados à experimentação na sua condução, o que inviabiliza tirar conclusões sobre a real viabilidade e funcionalidade da abordagem avaliada.

Vale lembrar que vários autores classificam os estudos visando avaliar a abordagem que propõem como estudos de caso. Contudo, na grande maioria das vezes nos os consideramos como simples provas de conceito devido à falta de

formalização durante a realização do estudo e que é necessária a um estudo experimental (KITCHENHAM *et al.*, 2002).

3.4.2.8 – Apoio ferramental

Uma avaliação arquitetural se caracteriza por apresentar algumas dificuldades, devido ao intenso conhecimento envolvido e à presença de atividades tediosas e repetitivas, que dificultam a sua adoção em ambientes industriais.

Através de um apoio ferramental, o verdadeiro potencial de uma avaliação poderia ser mais explorado e o seu uso em um contexto industrial poderia se tornar viável.

Portanto, através desse critério, pretendemos identificar qual abordagem possui esse apoio ferramental.

3.4.3 – Caracterização das abordagens

Visando facilitar a descrição e caracterização dessas abordagens, as seguintes medidas foram tomadas:

- O nome dado às abordagens nem sempre corresponde a seus nomes oficiais. Em alguns casos, criamos a sigla com base no nome definido pelos criadores para denominá-las.
- A ordem de descrição de cada abordagem foi definida com base na data da publicação dos trabalhos que a originaram.
- Visando descrever e caracterizar cada abordagem, utilizamos principalmente as informações contidas nas publicações selecionadas pela revisão.
- O símbolo (-) será utilizado visando indicar quando não foi possível identificar como a abordagem caracterizada atende a determinado critério.
- Para algumas abordagens de avaliação, a revisão sistemática identificou publicações que a descrevem mas não as publicações que a originaram. Nesses casos, essas publicações que originaram a abordagem não foram identificadas na tabela de caracterização.

A seguir as abordagens são identificadas e caracterizadas de acordo com o *framework* que definimos.

3.4.3.1 – SAR

SAR (*Software Architecture Review*) (ERICKSON *et al.*, 1993) é uma abordagem de avaliação que objetiva fornecer um melhor entendimento do software sobre a evolução de novos serviços, a robustez do sistema, e o gerenciamento de falhas. Portanto, essa abordagem avalia a arquitetura sobre a perspectiva dos atributos de evolução e confiabilidade.

Essa avaliação é feita através de *checklists* que são desenvolvidos a partir das características do sistema e dos critérios que se deseja avaliar. Contudo, visto que eles são orientados ao produto, novos *checklist* devem ser gerados a cada nova avaliação.

Tabela 3.5 – Caracterização da abordagem

SAR	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Questionamento (*checklists*)
Processo seguido	1. Desenvolver *checklists* orientados ao produto a ser analisado com informações sobre os critérios necessários para realizar a revisão. 2. Averiguar junto aos projetistas da arquitetura as características atuais do sistema através da execução dos *checklists*; 3. Caracterizar a informação obtida em (2); 4. Relatar os resultados da revisão aos envolvidos;
Artefatos requeridos	-
Artefatos produzidos	-
Características de qualidade	▪ Confiabilidade ▪ Modificabilidade
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	-
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (ERICKSON *et al.*)	

3.4.3.2 – SAAM

SAAM (*Software Architecture Analysis Method*) (KAZMAN *et al.*, 1994) é uma abordagem que avalia a arquitetura de um software através da análise de como os requisitos de modificabilidade, de variabilidade e funcionais foram atendidos.

Essa abordagem pode ser utilizada para avaliar a qualidade de uma arquitetura ou então para identificar, dentro de um conjunto de arquiteturas, a mais adequada para atender ao problema proposto.

O desenvolvimento dessa abordagem foi motivado pela observação de que os arquitetos regularmente fazem afirmações que não conseguem provar. SAAM tenta avaliar essas afirmações através do uso de cenários que colocam em evidência os atributos de qualidade relacionados com a afirmativa que se deseja provar. Com isso, a principal função do SAAM é indicar os locais onde a arquitetura falha em se adequar aos requisitos de qualidade evidenciados.

SAAM é uma abordagem de avaliação arquitetural bastante utilizada. Além disso, ela é usada também como base para a criação de novas abordagens de avaliação.

Quando utilizada para criar outras abordagens, algumas características de SAAM são modificadas visando adaptá-la às características do novo contexto em que será utilizada. Essas modificações podem ser mínimas (LASSING *et al.*, 1999b; GRAAF *et al.*, 2005) ou não. Algumas das abordagens que utilizam SAAM como base foram identificadas na revisão sistemática e também estão descritas nesse capítulo.

Tabela 3.6 – Caracterização da abordagem

SAAM	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) ▪ Rastreabilidade com os requisitos funcionais Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Questionamento (cenário)
Processo seguido	1. Especificação, por parte dos *stakeholders*, do conjunto de cenários que representam as mudanças por eles desejadas; 2. Inspeção dos cenários especificados previamente priorizados; 3. Mapeamento dos cenários para a representação da arquitetura; 4. Identificação das discrepâncias que foram evidenciadas através do mapeamento.
Artefatos requeridos	▪ Descrição do problema ▪ Documento de requisitos ▪ Representação da arquitetura
Artefatos produzidos	▪ Lista com os requisitos priorizados ▪ Lista de discrepâncias ▪ Mapeamento da representação arquitetural aos atributos de qualidade, permitindo o rastreamento entre os elementos arquiteturais e os requisitos que eles implementam
Características de qualidade	Modificabilidade
Momento da avaliação	Após construção
Apoio ferramental	▪ SAAMTool ▪ Intelligent Tool Based-Agent for Software Architecture Evaluation
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (KAZMAN *et al.*, 1994) ⇒ (ABOWD, 1995) ⇒ (ABOWD *et al.*, 1996) ⇒ (EICKELMANN e RICHARDSON, 1996) ⇒ (KAZMAN *et al.*, 1996) ⇒ (MCCRICKARD e ABOWD, 1996) ⇒ (LASSING *et al.*, 1999b) ⇒ (GANNOD e LUTZ, 2000) ⇒ (CLEMENTS *et al.*, 2002) ⇒ (DOBRICA e NIEMELA, 2002)	⇒ (KAZMAN e BASS, 2002) ⇒ (LAND, 2002) ⇒ (BAHSOON e EMMERICH, 2003) ⇒ (SVAHNBERG, 2003) ⇒ (BABAR e GORTON, 2004) ⇒ (BABAR *et al.*, 2004) ⇒ (BENARIF *et al.*, 2004) ⇒ (FOLMER e BOSCH, 2005) ⇒ (GRAAF *et al.*, 2005)

3.4.3.3 – SAA

SAA (*System Architecture Analysis*) (GLOGER *et al.*, 1996) é uma abordagem de avaliação arquitetural desenvolvida pela Siemens e que busca otimizar a arquitetura

do software para atender tanto estratégias de mercado definidas pela organização quanto aspectos tecnológicos desejáveis.

Essa abordagem é baseada em quatro princípios essenciais: (1) Análise da arquitetura através de avaliações das decisões de projeto; (2) Avaliações realizadas levando em consideração a aplicação a ser desenvolvida, assim como as expectativas do mercado; (3) Identificação de potenciais de otimização através da comparação entre soluções (arquiteturas) alternativas; (4) Avaliação objetiva realizada por diversas equipes de *stakeholders*.

A SAA objetiva avaliar se a arquitetura é uma "boa arquitetura". Para SAA, uma "boa arquitetura" consiste em uma arquitetura criada a partir de decisões de projeto que foram tomadas visando implementar os requisitos definidos para o sistema.

Tabela 3.7 – Caracterização da abordagem

SAA	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ - Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Questionamento (Questionário)
Processo seguido	1. Identificação dos critérios de avaliação: Consiste na priorização dos principais requisitos que serão usados para avaliar a arquitetura; 2. Identificação das características do sistema e de soluções alternativas: Consiste em usar uma equipe de experts para identificar as principais características do sistema e definir arquiteturas alternativas para essas características. Essas soluções (arquiteturas alternativas) são utilizadas como comparativo com a arquitetura avaliada; 3. Avaliação sistemática das diferentes arquiteturas: Consiste em avaliar as arquiteturas (proposta + alternativa) levando em consideração principalmente quão bem cada uma implementa os requisitos; 4. Avaliação dos resultados obtidos: Procura-se identificar os pontos fortes e fracos de cada arquitetura que serão usados como base para gerar recomendações para otimizar a arquitetura final.
Artefatos requeridos	▪ Documento de requisitos ▪ Representação arquitetural
Artefatos produzidos	▪ Recomendações descrevendo o que deve ser feito para "otimizar" a arquitetura final
Características de qualidade	▪ Desempenho ▪ Usabilidade ▪ Escalabilidade
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (GLOGER *et al.*, 1996)	

3.4.3.4 – SAAMER

SAAMER (*Software Architecture Analysis Method for Evolution and Reusability*) (LUNG *et al.*, 1997) é uma abordagem que objetiva caracterizar a arquitetura de um

software com o objetivo de projetar a sua evolução e reutilização em projetos do mesmo domínio.

Por ter sido construída com base na abordagem SAAM, SAAMER aplicou vários conceitos de SAAM no contexto de reutilização e evolução de software.

Uma das principais características dessa abordagem é de ter sido baseada em um *framework* de obtenção de informação e de análise arquitetural para a sua concepção. Devido a essa característica, essa abordagem possui um processo de avaliação organizado, científico e repetível.

Entre as alterações feitas na abordagem base (SAAM), destaca-se a inclusão de novos artefatos requeridos, como o modelo do domínio, que pode auxiliar na comparação entre arquiteturas candidatas, por exemplo.

Tabela 3.8 – Caracterização da abordagem

SAAMER	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) ▪ Rastreabilidade com os requisitos funcionais Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Questionamento (cenário)
Processo seguido	-
Artefatos requeridos	▪ Descrição do problema ▪ Documento de requisitos ▪ Modelo do domínio ▪ Representação da arquitetura
Artefatos produzidos	-
Características de qualidade	▪ Reusabilidade ▪ Modificabilidade ▪ Manutenibilidade
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	-
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (DOBRICA e NIEMELA, 2002) ⇒ (BABAR *et al.*, 2004)	⇒ (FOLMER e BOSCH, 2005)

3.4.3.5 – PASA

PASA (*Performance Assesment of Software Architecture*) (WILLIAMS e SMITH, 1998) é uma abordagem que busca avaliar se os requisitos de desempenho especificados foram atendidos na arquitetura.

Após a sua realização, com base nas discrepâncias de desempenho identificadas, é possível ao arquiteto tomar os seguintes tipos de decisões visando resolver essas discrepâncias: "aliviar" os requisitos de desempenho, omitir a funcionalidade, aumentar o poder do hardware ou aplicar outros estilos arquiteturais para atender a esses requisitos.

Tabela 3.9 – Caracterização da abordagem

PASA	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Questionamento (cenário)
Processo seguido	1. Identificar os casos de uso críticos que são importantes para o bom funcionamento do sistema 2. Definir os cenários de desempenho para cada caso de uso crítico 3. Avaliar a arquitetura através dos cenários
Artefatos requeridos	▪ Casos de uso do sistema ▪ Representação da arquitetura
Artefatos produzidos	Lista de discrepâncias
Características de qualidade	Desempenho
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	-
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (BAHSOON e EMMERICH, 2003)	⇒ (BABAR e GORTON, 2004)

3.4.3.6 – SAEM

SAEM (*Software Architecture Evaluation Model*) (DUENAS *et al.*, 1998) é uma abordagem que procura organizar e utilizar métricas de qualidade, agrupadas a partir da aplicação do GQM (BASILI *et al.*, 1994), para avaliar modelos arquiteturais.

Tabela 3.10 – Caracterização da abordagem

SAEM	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando ADL (Architectural Description Language) Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Medição (GQM)
Processo seguido	-
Artefatos requeridos	-
Artefatos produzidos	-
Características de qualidade	Múltiplos
Momento da avaliação	Durante e Após construção
Apoio ferramental	-
Avaliação	-
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (DOBRICA e NIEMELA, 2002)	

3.4.3.7 – SBAR

SBAR (*Scenario-Based Architecture Reengineering*) (BENGTSSON e BOSCH, 1998) é uma abordagem de reengenharia arquitetural que utiliza diversos tipos de técnicas de avaliação como forma de identificar se o processo de reengenharia já

pode ser finalizado. Essa abordagem se divide principalmente em três grandes etapas: (1) incorporação dos novos requisitos funcionais na arquitetura (se houver); (2) caracterização dos atributos de qualidade especificados; (3) transformação arquitetural.

A etapa (2) consiste no uso de técnicas de avaliação para verificar se os atributos de qualidade do sistema se adequam aos requisitos. Caso isso não ocorra, transformações arquiteturais são realizadas, seguido de uma nova caracterização, até que os resultados esperados sejam obtidos.

Para cada atributo de qualidade que será avaliado, SBAR permite a escolha de uma técnica de avaliação: cenário, simulação, modelagem matemática ou experiência baseada em raciocínio.

Em relação às atividades para a avaliação arquitetural, a abordagem requer que cenários sejam especificados para cada requisito de qualidade. Em seguida, esses cenários são analisados de acordo com a técnica de avaliação escolhida e os resultados são interpretados.

Tabela 3.11 – Caracterização da abordagem

SBAR	
Características da abordagem	
Tipo de arquitetura	-
Tipo de técnica	Híbrido
Processo seguido	1. Especificação de cenários 2. Avaliação usando a técnica escolhida 3. Interpretação dos resultados 4. Evolução da arquitetura (se necessário)
Artefatos requeridos	▪ Documento de requisitos ▪ Representação da arquitetura
Artefatos produzidos	-
Características de qualidade	Múltiplos
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (DOBRICA e NIEMELA, 2002)	⇒ (BABAR *et al.*, 2004)

Uma característica interessante da avaliação realizada por essa abordagem é que ela pode ser executada de forma completa ou de forma estatística. A principal diferença é que a forma estatística busca identificar um conjunto de cenários representativos que não precisem cobrir todos os possíveis casos. Isso foi uma solução encontrada para o problema relacionada ao custo de se gerar o máximo de cenários possíveis.

3.4.3.8 – ALPSM

ALPSM (*Architecture-Level Prediction of Software Maintenance*) (BENGTSSON e BOSCH, 1999) é uma abordagem que busca analisar a manutenibilidade de um sistema de software através da observação do impacto de cenários à nível arquitetural.

Para isso, essa abordagem utiliza um perfil de manutenção, ou seja, um conjunto de cenários que representam as tarefas de manutenção referente principalmente às adaptações ou melhorias que se deseja realizar na arquitetura avaliada.

Com o uso desse perfil, a arquitetura é avaliada através de uma análise de impacto das mudanças necessárias para se adequar aos cenários, assim como a grandeza dos esforços de manutenção necessária para cada cenário avaliado.

A principal diferença entre o ALPSM e o SAAM está na forma que os cenários são definidos. Os cenários do SAAM utilizam as perspectivas de todos os *stakeholders* disponíveis, contudo os do ALPSM utiliza somente as do especialista do domínio.

Tabela 3.12 – Caracterização da abordagem

ALPSM	
Características da abordagem	
Tipo de arquitetura	-
Tipo de técnica	Questionamento (cenário)
Processo seguido	1. Identificar os diferentes tipos de tarefas de manutenção que serão realizadas na arquitetura. 2. Criar os cenários que refletem essas manutenções; 3. Priorizar os cenários. Feita com base na probabilidade do cenário ocorrer durante o tempo de vida do sistema 4. Estimar o tamanho dos componentes envolvidos na alteração. Estimativa feita para determinar o esforço necessário para realizar a modificação; 5. Simular a execução dos cenários visando identificar o impacto das mudanças na arquitetura; 6. Calcular o esforço de manutenção.
Artefatos requeridos	▪ Documento de requisitos ▪ Representação da arquitetura ▪ Histórico com dados de manutenção da arquitetura avaliada
Artefatos produzidos	▪ Perfil definido de manutenção ▪ Previsão do esforço médio das tarefas de manutenção
Características de qualidade	Manutenibilidade
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (LASSING *et al.*, 1999b) ⇒ (DOBRICA e NIEMELA, 2002)	⇒ (BABAR *et al.*, 2004)

3.4.3.9 – ESAAMI

Quando aplicada ao contexto de reutilização, engenharia de domínio ou linha de produtos, a importância da arquitetura é ainda maior que quando aplicada em

sistemas tradicionais. Isso ocorre por ela ser especificada com o principal objetivo de ser amplamente reutilizada e se tornar a base de diferentes produtos de software.

Portanto, discrepâncias arquiteturais no contexto de reutilização devem ser evitadas e uma das ferramentas para diminuir a incidência desse tipo de defeitos é através da aplicação de métodos de análise.

Com esse objetivo, foi definida em (MOLTER, 1999) a abordagem ESAMI (*Extending SAAM by Integration in the Domain*) com o objetivo de integrar o método SAAM em processos de desenvolvimento de domínios específicos ou baseados em reutilização.

A principal modificação introduzida pelo ESAAMI à metodologia utilizada pelo SAAM está na reutilização do conhecimento de domínio, durante a avaliação, através de templates de análise.

Tabela 3.13 – Caracterização da abordagem

ESAAMI	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ -
Tipo de técnica	Questionamento (cenário)
Processo seguido	-
Artefatos requeridos	▪ Descrição do problema ▪ Documento de requisitos ▪ Representação arquitetural ▪ Template de análise focado nas características da arquitetura avaliada
Artefatos produzidos	-
Características de qualidade	▪ Modificabilidade ▪ Variabilidade ▪ Reusabilidade
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	-
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (DOBRICA e NIEMELA, 2002)	⇒ (BABAR *et al.*, 2004)

3.4.3.10 – SAAF

A abordagem SAAF (*Software Architecture Analysis of Flexibility*) (LASSING *et al.*, 1999a) é similar à abordagem SAAM, contudo ela disponibiliza suporte para a identificação e utilização de cenários que possuem uma alta complexidade.

Essa abordagem parte do princípio que a complexidade dos cenários é o mais importante fator de risco durante a avaliação de uma arquitetura, e então ela busca direcionar a forma de lidar com os cenários mais complexos.

Tabela 3.14 – Caracterização da abordagem

SAAF	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ - Nível de abstração: ▪ Arquitetura Refinada
Tipo de técnica	Questionamento (cenário)
Processo seguido	-
Artefatos requeridos	▪ Representação arquitetural ▪ Categorias de cenários complexos ▪ Instrumentos de medida
Artefatos produzidos	-
Características de qualidade	▪ Adaptabilidade
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (DOBRICA e NIEMELA, 2002)	⇒ (BABAR *et al.*, 2004)

3.4.3.11 – ATAM

O principal objetivo da abordagem ATAM (*Architecture Trade-off Analysis Method*) (KAZMAN *et al.*, 2000) é prover uma forma de entender a adequação de uma arquitetura com respeito a vários atributos de qualidade, levando em consideração a existência de *trade-offs* entre esses atributos. *Trade-off* entre duas características de qualidade ocorre quando se tenta maximizar uma das características e, por conseqüência, a segunda é automaticamente minimizada. Como exemplo, existe o *trade-off* entre desempenho e reutilização: se tentarmos obter o máximo de desempenho estaremos automaticamente diminuindo a capacidade em reutilizar a arquitetura visto que teremos que adaptá-la a um contexto específico; e vice-versa.

ATAM foi idealizada com base em três pontos: (a) os conceitos e o conhecimento obtido com estudos sobre estilos arquiteturais; (b) as comunidades que se dedicam ao estudo dos diferentes tipos de atributos de qualidade que podem ser utilizados em uma avaliação arquitetural; (c) os conceitos e lições aprendidas com o método SAAM, o predecessor ao método ATAM.

Sendo assim, ATAM se concentra principalmente em identificar as abordagens e os estilos arquiteturais utilizados na especificação da arquitetura analisada, pois eles representam a forma que o arquiteto utilizou para adequar a arquitetura aos requisitos de qualidade especificados.

Tabela 3.15 – Caracterização da abordagem

ATAM	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura refinada
Tipo de técnica	▪ Híbrido
Processo seguido	Macro-atividades: 1. Apresentação: atividades responsáveis por relatar os objetivos do negócio e a descrição da arquitetura; 2. Investigação e análise: atividades responsáveis por realizar um levantamento dos requisitos de qualidade que devem ser atendidos e das abordagens arquiteturais utilizadas para atendê-los. 3. Teste: atividades responsáveis por checar os resultados encontrados na análise usando cenários criados por outros participantes. 4. Exposição: atividades responsáveis por apresentar os resultados obtidos
Artefatos requeridos	▪ Representação arquitetural ▪ Objetivos do negócio ▪ Perspectivas dos *stakeholders* descritas através dos cenários ▪ Documento de requisitos
Artefatos produzidos	▪ Requisitos de qualidade priorizados ▪ Identificação das abordagens arquiteturais utilizadas ▪ Mapeamento requisitos com as abordagens ▪ Lista de riscos (discrepâncias) ▪ *Trade-off* points
Características de qualidade	Múltiplos
Momento da avaliação	Durante ou após construção
Apoio ferramental	▪ Intelligent Tool Based-Agent for Software Architecture Evaluation ▪ ArchE
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (CARRIERE *et al.*, 1999) ⇒ (IVEZIC *et al.*, 2000) ⇒ (KAZMAN *et al.*, 2000) ⇒ (LEE *et al.*, 2001) ⇒ (CLEMENTS *et al.*, 2002) ⇒ (DOBRICA e NIEMELA, 2002) ⇒ (KAZMAN e BASS, 2002) ⇒ (BAHSOON e EMMERICH, 2003)	⇒ (SMITH e MERSON, 2003) ⇒ (SVAHNBERG, 2003) ⇒ (BABAR e GORTON, 2004) ⇒ (BABAR *et al.*, 2004) ⇒ (BENARIF *et al.*, 2004) ⇒ (FOLMER e BOSCH, 2005) ⇒ (LEE e CHOI, 2005)

3.4.3.12 – FAV

A abordagem FAV (*Framewok for Software Architecture Verification*) (LICHTNER *et al.*, 2000) consiste em um *framework* utilizado para analisar representações arquiteturais através do uso de prova formal.

Para isso, a arquitetura, que deve estar descrita através de uma ADL, é traduzida para uma representação matemática alternativa. A partir dessa representação, é feita uma conversão para PVS (Prototype Verification System), lógica de alta ordem, que permite a verificação automática da arquitetura.

Tabela 3.16 – Caracterização da abordagem

FAV	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando ADL (*Architectural Description Language*) ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ -
Tipo de técnica	Medição
Processo seguido	-
Artefatos requeridos	▪ Restrições arquiteturais (Requisitos de qualidade) ▪ Representação arquitetural
Artefatos produzidos	-
Características de qualidade	-
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (LICHTNER *et al.*, 2000)	

3.4.3.13 – SAASS

A abordagem SAASS (*Software Architecture Analysis based on Statechart Semantics*) (DIAS e VIEIRA, 2000) busca desenvolver e avaliar sistemas baseados em arquitetura e componentes.

Essa avaliação é feita com base em informações obtidas da representação arquitetural do sistema adicionada de informações relacionadas ao comportamento de seus componentes.

Tabela 3.17 – Caracterização da abordagem

SAASS	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando ADL (*Architectural Description Language*) ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura inicial ▪ Arquitetura refinada
Tipo de técnica	Medição (Simulação e *Model Checking*)
Processo seguido	-
Artefatos requeridos	▪ Representação arquitetural
Artefatos produzidos	-
Características de qualidade	▪ Estrutural ▪ Restrições relacionadas ao estilo arquitetural C2
Momento da avaliação	Durante e após construção
Apoio ferramental	Argus-I
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (DIAS e VIEIRA, 2000)	

Para isso, a arquitetura deve implementar o estilo arquitetural C2 (MEDVIDOVIC *et al.*, 1999) e estar descrita através de uma ADL associada. Após isso, é feito o uso do ambiente Argus-I que permite a avaliação dessa arquitetura através de diversas técnicas, como *Model checking* e Simulação.

3.4.3.14 – ARID

A abordagem ARID (*Active Reviews for Intermediate Designs*) (CLEMENTS, 2000) avalia a consistência de uma parte da arquitetura com respeito às demais partes já definidas. Para isso, ela combina a filosofia de revisões ativas com abordagens de avaliações baseados em cenários, como o SAAM.

Com essa combinação, ARID utiliza: (a) os revisores ativos das ADR, que asseguram principalmente resposta de alta qualidade sobre as questões realizadas durante a avaliação e (b) a abordagem de definir os cenários a serem analisados pelos *stakeholders*.

Essa abordagem, por revisar partes da arquitetura antes do final de sua inteira construção, permite obter indícios sobre a viabilidade de construção do artefato como um todo, além de possibilitar a descoberta de erros, inconsistências e elementos inadequados.

Para a sua execução, os revisores definem um conjunto de cenários, cuja ordem de execução é definida por votação dos envolvidos.

Tabela 3.18 – Caracterização da abordagem

ARID	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ - Nível de abstração: ▪ Arquitetura refinado
Tipo de técnica	Questionamento (cenário)
Processo seguido	-
Artefatos requeridos	▪ Representação arquitetural incompleta ▪ Documento de requisitos
Artefatos produzidos	-
Características de qualidade	Adequação da parte da arquitetura avaliada com os requisitos
Momento da avaliação	Durante construção
Apoio ferramental	-
Avaliação	-
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (CLEMENTS *et al.*, 2002) ⇒ (BAHSOON e EMMERICH, 2003)	⇒ (BABAR *et al.*, 2004)

3.4.3.15 – Qasar

A abordagem Qasar (*Quality Attribute-oriented Software ARchitecture*) (BOSCH, 2000) consiste em uma abordagem de projeto arquitetural que durante a sua execução realiza avaliações arquiteturais.

Essas avaliações são realizadas quando os arquitetos acreditam que atenderam a todos os requisitos. Portanto, o seu resultado permite identificar o que falta a ser feito ou então finalizar o processo de construção.

Tabela 3.19 – Caracterização da abordagem

Qasar	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ - Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Híbrida
Processo seguido	1. Elicitação dos requisitos com os *stakeholders* 2. Projeto inicial da arquitetura visando atender somente os requisitos funcionais 3. Avaliação preliminar visando identificar se os requisitos de qualidade estão atendidos. Essa avaliação pode utilizar tanto técnicas qualitativas (cenários) quanto quantitativas (simulação, modelagem matemática) 4. Análise dos resultados com relação aos resultados esperados. Se resultado for positivo o processo é finalizado; 5. Caso contrário é feita uma reengenharia e novas avaliações até que todos os requisitos estejam atendidos
Artefatos requeridos	▪ Documento de requisitos ▪ Representação arquitetural
Artefatos produzidos	-
Características de qualidade	Múltiplos
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (FOLMER e BOSCH, 2005)	

3.4.3.16 – Avaliação baseada em Model Checking

Mesmo com a exatidão e os benefícios comprovados das técnicas formais de avaliação, essas técnicas são pouco usadas na avaliação de arquiteturas devido à grande complexidade envolvida na sua aplicação.

Tendo em vista essas dificuldades, a ferramenta ARCADE (BARBER *et al.*, 2001) foi desenvolvida com o papel de automatizar a aplicação dessas técnicas mais complexas em avaliações arquiteturais.

A ARCADE (*Architecture Analysis Dynamic Environment*) combina principalmente simulação e *Model checking* para auxiliar o arquiteto na caracterização de um software a partir de sua arquitetura. Através dessa caracterização, essa ferramenta avalia principalmente a confiabilidade da arquitetura do software.

Tabela 3.20 – Caracterização da abordagem

Avaliação baseada em *Model Checking*	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Medição (Simulação e *Model Checking*)
Processo seguido	1. Detectar as violações das propriedades através da aplicação de *model checking*; 2. Corrigir a especificação; 3. Reaplicar a técnica até que nenhuma outra violação seja identificada.
Artefatos requeridos	▪ Documento de Requisitos ▪ Representação arquitetural
Artefatos produzidos	Log de violações
Características de qualidade	Adequação da arquitetura avaliada com os requisitos
Momento da avaliação	Após construção
Apoio ferramental	ARCADE
Avaliação	-
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (BARBER *et al.*, 2001)	

3.4.3.17 – CBAM

A abordagem CBAM (*Cost Benefit Analysis Method*) (KAZMAN *et al.*, 2001) é uma extensão da ATAM que visa realizar a modelagem econômica do software através da análise de sua arquitetura.

Conceitualmente, o CBAM continua a partir do ponto em que o ATAM acaba. Nessa abordagem, os custos e benefícios são analisados em relação aos atributos de qualidade do sistema, ou seja, objetiva-se saber como questões referentes ao custo e benefício influenciam ou são influenciados pelos atributos de qualidade do sistema. Para isso, é feita a adição de dimensões monetárias (custo e benefício) ao resultado do ATAM como um atributo adicional de *trade-off*.

Tabela 3.21 – Caracterização da abordagem

CBAM	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura refinada
Tipo de técnica	Híbrido
Processo seguido	1. Escolher os cenários e estratégias arquiteturais; 2. Caracterizar os benefícios obtidos pelos atributos de qualidade; 3. Quantificar os benefícios oferecidos pelas estratégias arquiteturais escolhidas; 4. Quantificar os custos e implicações no cronograma; 5. Calcular o que for desejável e possível de ser feito; Realizar a tomada de decisão.

Artefatos requeridos	▪ Representação arquitetural ▪ Objetivos do negócio ▪ Perspectivas dos *stakeholders* descritas através dos cenários ▪ Documento de requisitos
Artefatos produzidos	▪ Requisitos de qualidade priorizados ▪ Identificação das abordagens arquiteturais utilizadas ▪ Mapeamento requisitos com as abordagens ▪ Lista de riscos (discrepâncias) ▪ *Trade-off* points ▪ Estimativas de custos e benefícios referente à cada abordagem utilizada
Características de qualidade	▪ Múltiplos (qualidade) ▪ Custo e benefícios (financeiro)
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	-
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (BAHSOON e EMMERICH, 2003)	⇒ (LEE e CHOI, 2005)

Os resultados obtidos pelo CBAM podem auxiliar os *stakeholders* em determinar o conjunto de estratégias arquiteturais que abordam os cenários de qualidade com maior benefício e menor custo.

3.4.3.18 – ALMA

O método ALMA (*Architecture-Level Modifiability Analysis*) (BENGTSSON, 2002) utiliza cenários para avaliar a arquitetura sob a perspectiva das características de modificabilidade. Este método foi criado a partir da união do ALPSM e SAAF formando, de acordo com os seus criadores, o método unificado de análise de modificações arquiteturais.

Por assegurar a validação somente de um atributo de qualidade, seus criadores aconselham a utilizá-la em conjunto com um ou mais métodos que suprem a necessidade de se avaliar os atributos de qualidade restantes (BENGTSSON *et al.*, 2004).

Tabela 3.22 – Caracterização da abordagem

ALMA	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Questionamento (cenário)
Processo seguido	1. Determinar o objetivo da análise. 2. Descrever as partes relevantes da arquitetura que serão analisadas. 3. Elicitar os cenários de evolução ou alteração relevantes. 4. Avaliar os cenários na arquitetura. 5. Interpretar os resultados.
Artefatos requeridos	▪ Documento de requisitos

	▪ Representação da arquitetura ▪ Log de manutenção
Artefatos produzidos	-
Características de qualidade	Modificabilidade
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (SVAHNBERG, 2003) ⇒ (BABAR e GORTON, 2004) ⇒ (BABAR *et al.*, 2004)	⇒ (FOLMER e BOSCH, 2005)

3.4.3.19 – Metodologia de avaliação baseada no modelo "4+1"

Essa abordagem (CHOI e YEOM, 2002) tem como objetivo avaliar arquiteturas de software que utilizam a abordagem "4+1" (KRUCHTEN, 1995) e a UML como formas de documentação e representação arquitetural. Através de questionamentos, busca avaliar a coerência entre as visões e as decisões tomadas para atender aos requisitos.

O resultado final dessa avaliação não ajuda somente na identificação dos riscos em um projeto, mas também é muito útil na compreensão da adequação da arquitetura aos seus atributos de qualidade.

Tabela 3.23 – Caracterização da abordagem

Metodologia de avaliação baseada no modelo "4+1"	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) ▪ Rastreabilidade com os requisitos Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Questionamento (cenário)
Processo seguido	1. Os passos referentes à avaliação arquitetural são: 2. Identificar os sub-projetos arquiteturais através dos requisitos funcionais. 3. Determinar o conjunto de estilos arquiteturais utilizados para a especificação da arquitetura avaliada. 4. Determinar o raciocínio para cada estilo arquitetural. 5. Identificar as relações entre os estilos arquiteturais identificados.
Artefatos requeridos	▪ Documento de requisitos ▪ Representação arquitetural
Artefatos produzidos	Lista com as decisões arquiteturais que influenciaram o atendimento aos requisitos de qualidade
Características de qualidade	Múltiplos
Momento da avaliação	Durante construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (CHOI e YEOM, 2002)	

3.4.3.20 – Simulação usando RAPIDE

Em (PEREZ *et al.*, 2002), uma abordagem de avaliação foi definida através do uso de simulação para avaliar o desempenho de arquiteturas representadas através da ADL RAPIDE.

Durante a execução dessa abordagem, a simulação é feita através do envio de mensagens para cada componente que faz parte da arquitetura visando analisar como eles reagem ao tipo e à quantidade de mensagens enviadas (avaliando principalmente a funcionalidade e o desempenho do componente).

Tabela 3.24 – Caracterização da abordagem

Simulação usando RAPIDE	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura refinada
Tipo de técnica	Medição (simulação)
Processo seguido	-
Artefatos requeridos	-
Artefatos produzidos	-
Características de qualidade	▪ Usabilidade ▪ Funcionalidade ▪ Confiança ▪ Manutenibilidade
Momento da avaliação	Após construção
Apoio ferramental	RAPTOR
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (PEREZ *et al.*, 2002)	

3.4.3.21 – OASE

A abordagem OASE foi descrita em (JAZAYERI, 2002). Ela é usada para avaliar a arquitetura sob a perspectiva de características de estabilidade.

Para isso, essa abordagem utiliza métricas simples (tamanho do módulo, número de módulos alterados e número de módulos adicionados em diferentes *releases*) para sumarizar o padrão de evolução do software entre as diferentes *releases*.

Tabela 3.25 – Caracterização da abordagem

OASE	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ - Nível de abstração: ▪ Arquitetura refinada
Tipo de técnica	Medição
Processo seguido	-
Artefatos requeridos	-
Artefatos produzidos	-

Características de qualidade	▪ Estabilidade ▪ Evolução
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (BAHSOON e EMMERICH, 2003)	

3.4.3.22 – (SVAHNBERG et al., 2002)

A abordagem descrita em (SVAHNBERG *et al.*, 2002) consiste em uma abordagem utilizada para entender as vantagens e desvantagens de diferentes estruturas arquiteturais em relação aos requisitos de qualidade especificados. Sendo assim, a avaliação realizada por essa abordagem permite identificar a arquitetura mais adequada a ser usada entre as alternativas disponíveis.

Durante essa avaliação, cada revisor realiza uma comparação das diferentes arquiteturas com relação aos atributos de qualidade e também uma comparação dos diferentes atributos de qualidade com relação às arquiteturas disponíveis. Essas comparações produzem uma lista de valores para cada arquitetura e aquela que possuir os maiores valores é a mais adequada.

Tabela 3.26 – Caracterização da abordagem

(SVAHNBERG *et al.*, 2002)	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura inicial ▪ Arquitetura refinada
Tipo de técnica	Medição
Processo seguido	1. Identificar principais requisitos de qualidade e potenciais arquiteturas 2. Criar o *framework* de comparação 3. Análise dos *frameworks* obtidos 4. Identificação da arquitetura mais adequada
Artefatos requeridos	▪ Documento de requisitos ▪ Representação arquitetural
Artefatos produzidos	▪ Documento apontando as diferenças entre cada arquitetura candidata em relação aos requisitos priorizados
Características de qualidade	Múltiplos
Momento da avaliação	Durante construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (SVAHNBERG, 2003)	

3.4.3.23 – ArchOptions

ArchOptions (BAHSOON, 2003) é uma abordagem para avaliar arquiteturas sob a perspectiva de características de estabilidade utilizando insights sobre possíveis

alterações e sobre como a arquitetura irá se comportar em relação a essas modificações. Assim como CBAM, essa abordagem também leva em consideração durante o processo de avaliação questões econômicas que influenciam o projeto da arquitetura.

Tabela 3.27 – Caracterização da abordagem

ArchOptions	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ - Nível de abstração: ▪ Arquitetura refinada
Tipo de técnica	Medição
Processo seguido	-
Artefatos requeridos	-
Artefatos produzidos	-
Características de qualidade	▪ Estabilidade ▪ Evolução
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	-
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (BAHSOON e EMMERICH, 2003)	

3.4.3.24 – ASAAM

Usualmente, a qualidade de uma arquitetura é avaliada através da análise do impacto de cenários pré-definidos sobre os elementos da arquitetura. Uma possível forma de adequar essa arquitetura aos cenários especificados é através do uso de *refactoring*.

Tabela 3.28 – Caracterização da abordagem

ASAAM	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura refinada
Tipo de técnica	Questionamento (cenário)
Processo seguido	1. Desenvolvimento da arquitetura candidata; 2. Especificação dos cenários a partir das perspectivas de vários *stakeholders*; 3. Avaliação individual de cada cenário e identificação dos aspectos arquiteturais. 4. Execução desses cenários para possibilitar a classificação dos componentes. 5. *Refactoring* da arquitetura baseado nos resultados obtidos pela execução dos cenários.
Artefatos requeridos	▪ Documento de requisitos ▪ Representação das arquiteturas candidatas ▪ Descrição do problema
Artefatos produzidos	-
Características de qualidade	▪ Modificabilidade

	▪ Variabilidade ▪ Atendimento aos requisitos funcionais
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (TEKINERDOGAN, 2004)	

Contudo, é observado que nem sempre os componentes podem ser coesos devido à presença de propriedades similares executadas por vários componentes.

3.4.3.25 – Avaliação de desempenho proposto por (PURHONEN, 2004)

Durante o desenvolvimento de software embarcado, deve ser levado em consideração vários tipos de requisitos específicos a esse domínio, como restrições de tempo, de recursos e alto desempenho. Além do mais, a mesma arquitetura deve ser facilmente adaptada devido a constante evolução do hardware onde o software é embarcado.

Tabela 3.29 – Caracterização da abordagem

Avaliação de desempenho proposto por (PURHONEN, 2004)	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Híbrido
Processo seguido	1. Determinar o escopo da avaliação. 2. Descrever a arquitetura através do uso de visões arquiteturais 3. Analisar o impacto. O impacto dos atributos de qualidade, nesse caso o de desempenho, é analisado com o auxílio de técnicas específicas desse domínio (RMA, abordagens baseadas em filas, etc.). 4. Analisar os resultados. Os resultados obtidos são comparados com os objetivos determinados durante a análise de requisitos. Refinamentos são propostos ao final dessa análise.
Artefatos requeridos	▪ Documento de requisitos ▪ Representação arquitetural
Artefatos produzidos	▪ Refinamentos na arquitetura para atender os requisitos de desempenho
Características de qualidade	Desempenho
Momento da avaliação	Durante construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (PURHONEN, 2004)	

Levando em consideração essas características, um abordagem de avaliação descrita em (PURHONEN, 2004) foi definida visando avaliar o desempenho de

arquiteturas de sistemas embarcados. Esse método utiliza cenários, simulações e outras técnicas específicas à esse domínio durante a avaliação.

3.4.3.26 – Saluta

Saluta (*Scenario based Architecture Level UsabiliTy Assessment*) (FOLMER, 2005) consiste em uma abordagem que busca avaliar como as decisões arquiteturais tomadas visando construir uma determinada arquitetura afetam os requisitos de usabilidade. Para isso, Saluta utiliza o *framework* SAU (FOLMER *et al.*, 2003) para identificar o suporte da arquitetura para a usabilidade.

Tabela 3.30 – Caracterização da abordagem

Saluta	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Questionamento (cenário)
Processo seguido	1. Criação de um perfil de utilização que descreve a usabilidade requerida. Com a criação desse perfil, cenários são definidos; 2. Análise da arquitetura visando identificar a usabilidade provida pela arquitetura; 3. Avaliação da usabilidade provida em relação aos cenários visando identificar se a implementação da usabilidade 4. Interpretação dos resultados
Artefatos requeridos	Representação arquitetural
Artefatos produzidos	-
Características de qualidade	Usabilidade
Momento da avaliação	Durante construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (FOLMER e BOSCH, 2005)	

3.4.3.27 – ASAV

A ASAV (*AT&T's Software Architecture Validation*) (MARANZANO *et al.*, 2005) é uma abordagem que avalia arquiteturas de software através de reuniões presenciais.

Através de um *checklist*, os revisores se preparam para essa reunião. Esse *checklist* é composto por questões que os arquitetos devem considerar durante essa avaliação.

Tabela 3.31 – Caracterização da abordagem

ASAV	
Características da abordagem	
Tipo de arquitetura	-
Tipo de técnica	Questionamento (*Checklist*)
Processo seguido	1. Definir se determinado projeto precisa ter usa arquitetura

	avaliada. 2. Preparação da avaliação: escolha da equipe, definição da documentação necessária, definição do problema 3. Reunião de revisão: apresentação do problema por parte da equipe de arquitetos 4. Follow-up: os revisores informam os problema encontrados
Artefatos requeridos	▪ Descrição do problema ▪ Entrevista com arquitetos ▪ Documento de Requisitos ▪ Representação arquitetural
Artefatos produzidos	Documento descrevendo os principais problemas encontrados
Características de qualidade	Múltiplos
Momento da avaliação	Durante construção
Apoio ferramental	-
Avaliação	Prova de conceito
Artigos selecionados pela revisão que permitiram identificar essa abordagem	
⇒ (MARANZANO *et al.*)	

3.5 – Análise das abordagens

Após identificar e caracterizar as abordagens de avaliação, dois tipos de análise foram realizadas: uma análise dos estudos que descrevem as abordagens e uma outra sobre as abordagens de avaliação em si.

Essas análises foram realizadas principalmente para fornecer informações que permitam entender como as abordagens existentes avaliam os documentos arquiteturais.

3.5.1 – Análise das publicações selecionadas

Devido ao fato de que, em alguns casos, as publicações que descrevem determinadas abordagens não são as publicações que a originaram, foi limitada a obtenção de informações suficientes para possibilitar uma caracterização mais completa dessas abordagens. Mesmo assim, com a análise das informações contida nessas publicações, conseguimos fazer constatações que permitem identificar algumas características do estado da arte em avaliação arquitetural.

A primeira constatação foi a falta de estudos experimentais para avaliar as abordagens de avaliação arquitetural identificadas. Nós procuramos por esse tipo de informação nas publicações devido aos possíveis ganhos que podem ser obtidos ao utilizar experimentação. Entre esses ganhos podemos citar, por exemplo, uma compreensão mais rigorosa da abordagem de avaliação e de sua aplicação.

Os estudos conduzidos com as abordagens identificadas consistem basicamente em provas de conceito que demonstram como elas são aplicadas e as melhorias que podem ser obtidas quando usadas para avaliar documentos arquiteturais (KAZMAN *et al.*, 1994; LEE *et al.*, 2003). Contudo, em nenhum artigo foi descrito algum estudo que,

por exemplo, realize uma comparação entre a abordagem descrita e uma abordagem *ad-hoc* ou qualquer outra que também objetive avaliar documentos arquiteturais.

A segunda constatação foi a existência de revisões bibliográficas cujo objetivo, assim como o nosso, é de identificar abordagens de avaliação arquitetural (CLEMENTS *et al.*, 2002; DOBRICA e NIEMELA, 2002; BAHSOON e EMMERICH, 2003; BABAR *et al.*, 2004). Contudo, essas revisões não descreveram os critérios de seleção que utilizaram para selecionar essas abordagens e nem como a busca por essas abordagens foi realizada, impedindo que sejam repetidas por outros pesquisadores.

Entre essas revisões, duas se destacaram (DOBRICA e NIEMELA, 2002; BABAR *et al.*, 2004) por terem definido um *framework* para caracterizar as abordagens de avaliação que identificaram. Usamos alguns dados por elas identificados para auxiliar na caracterização das abordagens que encontramos em comum.

Um fato marcante em algumas dessas revisões é que seus autores as realizaram principalmente com o objetivo de identificar motivações que justifiquem a abordagem de avaliação por eles criada. Todavia, quando uma revisão é conduzida com esse propósito, como em (BAHSOON e EMMERICH, 2003), normalmente as abordagens identificadas são somente aquelas que apresentam as limitações que o pesquisador pretende tratar através da sua nova abordagem. Isso ocorre pelo fato do pesquisador acabar utilizando, talvez de forma inconsciente, a limitação que ele pretende tratar como principal critério de seleção durante a busca por trabalhos relacionados, inserindo assim viés nos resultados obtidos pela revisão, pois ele pode ter deixado de identificar abordagens que já propuseram uma solução para a limitação que ele deseja solucionar.

A terceira constatação observada foi o número de métodos identificados pela revisão descrita nesse trabalho em relação às revisões existentes. A revisão sistemática identificou 27 abordagens enquanto as revisões *ad-hoc* (CLEMENTS *et al.*, 2002; DOBRICA e NIEMELA, 2002; BAHSOON e EMMERICH, 2003; BABAR *et al.*, 2004) não identificaram mais que 8 métodos cada.

3.5.2 – Análise das abordagens de avaliação identificadas

O principal objetivo das abordagens identificadas é a melhoria da qualidade da arquitetura de software através da análise dos documentos que a descreve.

Visto que estudos experimentais têm demonstrado que a utilização de técnicas de inspeção beneficia a melhoria da qualidade de artefatos de software (SHULL *et al.*, 2000; CONRADI *et al.*, 2003), visando um melhor entendimento optamos por

caracterizar as abordagens identificadas a partir de um conjunto de critérios normalmente presentes em técnicas de inspeção de software.

Um dos motivos pela realização dessa revisão sistemática é a necessidade que o nosso grupo de pesquisa e as empresas de software têm por uma abordagem de avaliação arquitetural que seja extensível, simples de ser aplicada, genérica e adaptável. Portanto, ao realizar essa caracterização e analisar os conceitos relacionados a cada abordagem identificada, procuramos identificar as abordagens que possuíssem essas características.

Contudo, não foi possível encontrar nenhuma abordagem que atendesse a todas essas características simultaneamente. Observamos que isso ocorre devido à ocorrência de quatro problemas principais (Tabela 3.32)[5] presentes nessas abordagens: grande subjetividade das abordagens, elevado custo de aplicação, dificuldades para avaliar simultaneamente o atendimento a várias características de qualidade e contexto limitado para a aplicação de algumas abordagens.

No contexto desse trabalho, se entende por subjetividade a falta de uma definição exata e completa de que características arquiteturais serão avaliadas durante a execução de uma abordagem de avaliação arquitetural. Para identificar a existência de subjetividade em alguma abordagem, avaliamos principalmente o tipo de técnica usada durante a avaliação.

Portanto, ao analisar as abordagens identificadas, uma elevada subjetividade foi observada principalmente nas abordagens que utilizam cenários como técnica de avaliação. Esse problema é causado pela incapacidade dos *stakeholders* em identificar e gerar todos os possíveis cenários que deveriam ser usados na avaliação.

Para conseguir gerar os cenários, esses *stakeholders* se vêem obrigados a utilizar subjetividade e criatividade como abordagens para definir o conjunto de cenários de avaliação (DOBRICA e NIEMELA, 2002). Além disso, durante a especificação dos cenários, os *stakeholders* inconscientemente podem definir cenários que não avaliam a arquitetura de forma completa, devido a sua familiarização com a arquitetura, provocando distorção nos resultados da avaliação.

O custo de aplicação de uma abordagem de avaliação está relacionado ao tipo de atividades que devem ser realizadas, à quantidade de pessoas que devem participar do processo e ao uso de ferramentas complexas, como simulação e métodos formais, que necessitam de especialistas em diversas áreas de conhecimento. Devido a isso, uma abordagem pode requerer uma grande quantidade de tempo e investimento financeiro para que possa ser executada, o que pode inviabilizar a sua aplicação em

[5] As informações obtidas em relação ao método SAEM não foram suficientes para caracterizá-lo, portanto não o incluímos na Tabela 3.32.

um contexto industrial. Para identificar esse custo de aplicação, analisamos principalmente os tipos de papéis, o processo executado e o tipo de técnica usada durante a avaliação.

Tabela 3.32- Mapeamento entre as abordagens de avaliação e problemas identificados durante a análise

Método de Avaliação	Grande subjetividade	Elevado custo de aplicação	Não avalia múltiplas características de qualidade	Contexto de aplicação limitado
SAR			X	X
SAAM	X	X	X	
SAA	X	X		
SAAMER	X	X	X	
PASA	X	X	X	
SBAR	X			
ALPSM	X		X	
ESAAMI	X	X	X	
SAAF	X	X	X	
ATAM	X	X		
FAV		X		X
SAASS		X	X	X
ARID	X	X		
QASAR	X	X		
Model Checking		X	X	X
CBAM	X	X		
ALMA	X	X	X	
4+1		X		X
Simulação usando RAPIDE		X	X	X
OASE		X	X	
SVAHNBERG	X	X		
ArchOptions		X	X	
ASAAM	X	X		
AD		X	X	X
SALUTA	X	X	X	
ASAV	X	X		

Sendo assim, após a análise, percebemos que o elevado custo de aplicação observado em determinadas abordagens está relacionado principalmente ao fato de terem sido desenvolvidas em um contexto acadêmico ou então para projetos de grande porte, que geralmente possuem alta disponibilidade de recursos. Com isso, somente um pequeno número de empresas consegue aplicar de forma correta essas avaliações (LATTANZE, 2005).

Outra característica importante a uma abordagem de avaliação é a quantidade de características de qualidade presentes na arquitetura que avalia. DOBRICA e NIEMELA (2002) sugerem que uma avaliação arquitetural deve ser feita sob a perspectiva de múltiplos requisitos, permitindo uma melhor compreensão dos pontos fracos e fortes dos produtos de software atuais.

Portanto, ao analisar as abordagens em relação a essa característica, foi levado em consideração se as abordagens foram construídas especificamente para avaliar um conjunto restrito de características ou se elas podem ser configuradas para avaliar as características desejadas.

Sendo assim, foi identificado que várias abordagens avaliam a arquitetura e seus respectivos documentos sob a perspectiva de um número restrito de características de qualidade. Algum desses métodos (GLOGER *et al.*, 1996; BENGTSSON *et al.*, 2004), conscientes dessa limitação, aconselham a execução de outro método de avaliação em paralelo. Mas tal abordagem poderia aumentar ainda mais o custo da avaliação.

O quarto problema identificado está relacionado à falta de consenso na comunidade em relação tanto às definições básicas quanto à forma de representar uma arquitetura (BUSCHMANN *et al.*, 1996; CLEMENTS *et al.*, 2004). Para identificar se determinada abordagem tem sua aplicação limitada a determinados contextos, procuramos determinar que peculiaridades relacionadas principalmente ao tipo de arquitetura avaliada, à técnica de avaliação realizada ou então aos artefatos requeridos, que implicavam na necessidade de um tipo específico de conhecimento, para que abordagem fosse executada, o que impossibilitaria a sua aplicação em determinados contextos

Ao realizarmos esse tipo de análise, identificamos que algumas abordagens de avaliação são baseadas em abordagens específicas de documentação ou utilizam técnicas de avaliação dominadas somente pelo grupo que propõem a abordagem de avaliação, o que dificulta a sua aplicação em diferentes projetos ou contextos.

3.6 – Conclusão

Ao longo deste capítulo, foram identificadas e caracterizadas as principais abordagens de avaliação arquitetural descritas na literatura técnica. Utilizamos como ferramenta para realizar essa revisão bibliográfica o conceito de revisão sistemática.

O uso de revisão sistemática, principalmente a sua etapa de planejamento, permitiu uma definição mais completa do que deveria ser procurado na literatura técnica, facilitando a busca e aprimorando o foco para a leitura e análise das publicações coletadas.

Após a realização dessa revisão, algumas evidências em relação ao uso de revisão sistemática na área de Engenharia de Software puderam ser observadas. Uma dessas evidências são os benefícios obtidos ao se utilizar revisão sistemática como ferramenta para levantamento bibliográfico, visto a grande quantidade de métodos que foram identificados em comparação com as revisões *ad-hoc* já realizadas.

Contudo, para a que a aplicação de revisões sistemáticas possa obter resultados mais completos é preciso (1) que seja definida uma taxonomia padrão para ser utilizada na classificação das publicações escritas na área, permitindo assim a obtenção de resultados mais completos quando se usar a busca por palavras chaves, e (2) que as bibliotecas digitais disponibilizem ferramentas, como máquinas de busca, que permitam a recuperação de estudos de forma confiável, evitando assim problemas como os ocorridos com a máquina de busca da ACM.

A principal contribuição dessa revisão consiste em uma identificação mais completa e uma caracterização das abordagens de avaliação arquitetural existentes, em relação às revisões *ad-hoc* encontradas na literatura. A partir da caracterização que foi realizada foi possível identificar os benefícios e limitações desses métodos.

Uma das principais motivações para a realização dessa busca por abordagens de avaliação arquitetural foi a necessidade de identificar uma abordagem facilmente adaptável, extensível, genérica o suficiente para ser aplicada em diferentes contextos e simples de ser realizada, o que facilitaria a sua aplicação em um contexto industrial, por exemplo. Contudo, com base nas informações e limitações observadas nas abordagens identificadas pela revisão sistemática constatamos que não existe ainda uma abordagem que atenda a todos esses requisitos (Tabela 3.33).

Tabela 3.33 - Mapeamento entre as características procuradas e as limitações que as inviabilizam

Característica Requerida	Limitações que a inviabiliza
Adaptável	• Contexto de aplicação limitado
Extensível	• Contexto de aplicação limitado
Genérica	• Elevado custo de aplicação • Não avalia múltiplas características de qualidade • Contexto de aplicação limitado
Simples	• Grande subjetividade • Elevado custo de aplicação

Com isso, devido ao conhecimento acumulado pela equipe após a realização dessa revisão, nos vimos motivados a desenvolver uma abordagem de avaliação arquitetural que possuísse tais características.

Esta abordagem para inspeção foi chamada de ArqCheck e consiste em um *checklist* configurável que objetiva a avaliação principalmente das características de qualidade representadas nos documentos arquiteturais em relação aos requisitos especificados para o seu projeto (BARCELOS e TRAVASSOS, 2005).

Este *checklist* foi criado com base na hipótese de que (1) o tipo de conhecimento utilizado pelo arquiteto de software na especificação de arquiteturas de software e que foi usado como base para especificar os itens de questionamento e (2) a configuração do *checklist* de acordo com as características que se deseja avaliar, permitem

identificar defeitos em documentos arquiteturais e minimizar as limitações observadas nas abordagens de avaliação arquitetural existentes, o que nos fornece uma abordagem de avaliação arquitetural próxima as nossas expectativas.

A proposta em questão é descrita no capítulo seguinte, intitulado "Inspeção de documentos arquiteturais".

Capítulo 4 – Inspeção de documentos arquiteturais

Neste capítulo, apresentamos ArqCheck, a abordagem proposta para inspecionar documentos arquiteturais através do uso de checklist. Justificamos também o uso dos conceitos que compõem a abordagem e o papel de cada um deles na avaliação arquitetural.

4.1 – Introdução

O custo de corrigir defeitos em artefatos de software aumenta exponencialmente com o decorrer do processo de desenvolvimento (BOEHM, 1981; BOEHM e BASILI, 2001). Portanto, iniciativas para avaliar a qualidade desses artefatos devem ser realizadas no sentido de identificar e corrigir os defeitos tão logo sejam introduzidos.

O documento arquitetural, responsável por descrever a arquitetura de um software, é um desses artefatos que deve ter a sua qualidade avaliada. Ao avaliá-lo, procura-se principalmente melhorar a qualidade da arquitetura nele descrita.

A avaliação desse documento é justificada pela sua importância para os *stakeholders*, uma vez que ele é utilizado em diversos momentos no processo de desenvolvimento do software. Contudo, como avaliar se a arquitetura descrita no documento arquitetural atende aos requisitos especificados?

Como vimos no Capítulo 3, existem diversas abordagens de avaliação descritas na literatura que possuem como foco principal avaliar a arquitetura, principalmente em relação à forma como ela atende aos requisitos. Essas abordagens utilizam diferentes técnicas de avaliação e já foram aplicadas em diferentes contextos. Todavia, algumas limitações presentes nessas abordagens dificultam a sua aplicação, principalmente em ambiente industrial (LATTANZE, 2005).

Estudos têm mostrado que abordagens baseadas em revisão de artefatos de software, quando utilizadas visando à melhoria da sua qualidade, apresentam bons resultados e estão sendo cada vez mais usados no contexto industrial (SHULL *et al.*, 2000; CONRADI *et al.*, 2003). Isso ocorre devido à sua eficiência e ao seu baixo custo para identificar defeitos, reduzindo assim o retrabalho, o custo total do projeto e melhorando a qualidade dos produtos (LAITENBERGER e ATKINSON, 1999; AURUM *et al.*, 2002).

Entre as abordagens de revisão, a inspeção de software é uma das que se destaca devido aos resultados que se tem obtido com a sua aplicação (SHULL *et al.*, 2000; CONRADI *et al.*, 2003). Portanto, visto esses resultados obtidos, essa

dissertação propõe uma abordagem para inspeção de documentos arquiteturais (BARCELOS e TRAVASSOS, 2006a)

Nas seções a seguir, justificamos a decisão em utilizar inspeção como abordagem para revisar documentos arquiteturais, mostramos como pretendemos minimizar as limitações presentes nas abordagens identificadas pela revisão sistemática e descrevemos a abordagem proposta.

4.2 – Inspecionando documentos arquiteturais

Qualidade de software consiste na conformidade do produto final e dos artefatos de software intermediários (1) aos requisitos que tenham sido especificados, (2) aos padrões de desenvolvimento que tenham sido claramente documentados e (3) às características implicitamente esperadas de todo software a ser desenvolvido (PRESSMAN, 2001).

Entre as atividades que podem ser utilizadas para verificar a qualidade de um software se encontram (MELO *et al.*, 2001):

- Revisões de software;
- Testes;
- Padrões e procedimentos formais;
- Controle de mudanças;
- Métricas de software;
- Procedimentos para coleção e disseminação de informações.

Revisão de software, de acordo com o padrão 1028 da IEEE (1988), visa avaliar produtos de software por uma equipe tecnicamente qualificada com o intuito de identificar discrepâncias. Diversos métodos têm sido propostos para a realização de revisões técnicas e esses métodos se diferenciam principalmente pelo seu grau de formalidade (WEINBERG e FREEDMAN, 1984). Inspeção de software, por exemplo, é um método de revisão de software que se destaca por ser rigoroso, formal e executado com o objetivo de examinar artefatos de software (FAGAN, 1976).

Uma outra maneira de detectar defeitos em artefatos é através de testes. No entanto, a aplicação de testes descobre apenas sintomas de problemas e, desta forma, pode necessitar da realização de um trabalho refinado e custoso para detectar os defeitos que causaram o sintoma da falha (MELO *et al.*, 2001).

BASILI e SELBY (1987) afirmam que inspeções descobrem defeitos que podem não ser descobertos nos testes. BOEHM e BASILI (2001) ressaltam ainda que inspeções e testes capturam diferentes tipos de defeitos em diferentes momentos do

processo de desenvolvimento de software. Portanto, é interessante aplicar tanto inspeções quanto testes para detectar defeitos em artefatos de software.

Contudo, o teste de um software consiste na análise dinâmica do produto, ou seja, na execução do produto com o objetivo de verificar a presença de falhas (NETO e TRAVASSOS, 2005). Visto que no contexto dessa dissertação objetivamos avaliar documentos arquiteturais, abordagens de testes não podem ser aplicadas a esse tipo de artefato por ele não ser passível de execução. A aplicação de uma abordagem de revisão de software se mostra uma opção mais adequada para a avaliação da qualidade de documentos arquiteturais.

Entre as abordagens de avaliação identificadas pela revisão sistemática (Capítulo 3), a maioria das abordagens que objetivam garantir a qualidade através da revisão da arquitetura (técnicas de questionamento) utiliza a execução de cenários como técnica de revisão. Cenários são usados por permitirem uma fácil representação das características que se deseja avaliar (ABOWD *et al.*, 1997).

Contudo, as abordagens que utilizam essa técnica como base apresentam problemas que dificultam a sua aplicação (BARCELOS e TRAVASSOS, 2006b). Sendo assim, optamos por definir uma abordagem usando conceitos relacionados a inspeção de software para avaliar os documentos arquiteturais.

4.2.1 – Benefícios ao aplicar inspeção de software

Diferente de outras abordagens de revisão, inspeção de software se caracteriza por visar principalmente a detecção de defeitos nos artefatos avaliados (YOUNESSI, 2002) e por poder ser utilizada na revisão dos diferentes artefatos criados durante o processo de desenvolvimento (Figura 4.1).

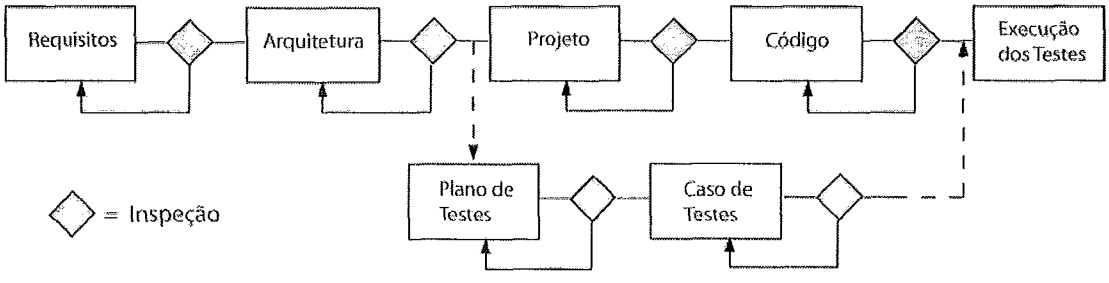


Figura 4.1 - Inspeções de software nos diferentes artefatos. Adaptado de (KALINOWSKI, 2004)

Quando inspeção é aplicada em artefatos de software visando a melhoria da qualidade é possível capturar até 60% dos defeitos contidos nesses artefatos (BOEHM e BASILI, 2001). Além disso, a aplicação de inspeção de software permite beneficiar o projeto sobre outras perspectivas, como por exemplo, diminuição de esforço de desenvolvimento.

De acordo com BOEHM e BASILI (2001), o esforço gasto por organizações de software com retrabalho varia em média entre 40% e 50% do esforço total do desenvolvimento de um projeto. A aplicação de inspeção de software permite que esse esforço seja reduzido em média para 10% a 20% (JONES, 1991). Esta redução no retrabalho é maior se a inspeção for aplicada, entre outros, sobre o artefato que representa a arquitetura do software (BOEHM *et al.*, 2000).

Portanto, visto os benefícios que se obtém ao utilizar inspeção de software e o fato de que a revisão do documento arquitetural contribui para a melhoria da qualidade do software (BABAR *et al.*, 2004), desenvolvemos como trabalho de mestrado uma abordagem para inspecionar documento arquiteturais que permita a identificação e correção de defeitos no início da fase de projeto, momento esse onde os custos de correção são menores.

4.3 – ArqCheck: Uma abordagem para inspeção de documento arquitetural

De acordo com LAITENBERGER e DEBAUD (1998), do ponto de vista técnico, uma abordagem para inspeção de software é formada pelo tipo do artefato de software a ser avaliado, pela forma como a equipe de profissionais será definida para realizar a inspeção, pela técnica de avaliação utilizada e pelo processo de execução seguido. Portanto, uma possível forma para se definir uma abordagem para inspeção seria através da realização das seguintes tarefas:

- Determinar o artefato de software que deve ser avaliado e o tipo de informação que o compõem;
- Identificar os perfis das pessoas que devem ser envolvidas nessa inspeção, assim como o papel que desempenham durante a avaliação;
- Definir os tipos de defeitos que a abordagem busca identificar;
- Definir a técnica de avaliação utilizada para identificar os defeitos no artefato a ser avaliado, e;
- Especificar o processo composto pelas atividades a serem executadas para atingir o objetivo final da avaliação.

A seguir, descrevemos essas tarefas com o objetivo de caracterizar os elementos que compõem ArqCheck e descrever como pretendemos minimizar as limitações encontradas nas demais abordagens de avaliação arquitetural.

4.3.1 – Artefato de software avaliado

ArqCheck objetiva a melhoria da qualidade da arquitetura de um software através da identificação de defeitos no seu documento arquitetural.

Segundo TRAVASSOS *et al.* (1999), artefatos criados durante o ciclo de desenvolvimento de um software devem possuir as seguintes características: (1) devem refletir de forma completa o sistema especificado através dos requisitos, (2) não devem conter restrições desnecessárias oriundas de outros domínios que influenciam a solução, e (3) não devem conter inconsistências ou ambigüidades para que seja possível aos usuários do artefato utilizar a arquitetura de forma correta. Sendo assim, a abordagem proposta busca identificar defeitos arquiteturais. Esse tipo de defeito está relacionado à falta dessas características em um documento arquitetural.

Visto que não existe uma padronização para se representar arquitetura de software, a abordagem proposta procura avaliar o documento independentemente da forma como as informações são representadas. Contudo, para que ArqCheck possa ser aplicada é necessário que o documento arquitetural contenha algumas informações, como as definidas pelas recomendações da IEEE 1471 que abordam a descrição arquitetural de sistemas de software (IEEE, 2000).

Sendo assim, definimos os seguintes requisitos para o documento arquitetural para que ArqCheck possa ser aplicada na sua avaliação:

- Identificar os elementos arquiteturais que compõem a solução a ser construída, assim como a forma que esses elementos estão organizados;
- Descrever o papel de cada elemento dentro da arquitetura;
- Identificar como cada requisito relevante a nível arquitetural está sendo atendido através da arquitetura documentada. Essa identificação pode ser feita principalmente através do rastreamento de que requisito está sendo atendido e quais requisitos justificam a criação de determinado elemento arquitetural;
- Representar o software através de diferentes perspectivas, como por exemplo, através do uso de visões arquiteturais.

Uma possível abordagem que poderia ser usada para descrever essas informações seria compor o documento com um conjunto de modelos, que representam as diferentes visões arquiteturais. Cada visão seria composta por diagramas gráficos que representariam os elementos arquiteturais sob determinada perspectiva.

Esse documento possuiria também um catálogo de elementos arquiteturais cujo objetivo é descrever o papel de cada elemento representado nos diferentes modelos e identificar a rastreabilidade entre os elementos arquiteturais e os requisitos

especificados. Na Figura 4.2, está representado um exemplo de como seria um documento com essas características.

Descrição da arquitetura

1. Visão Modular

Diagramas

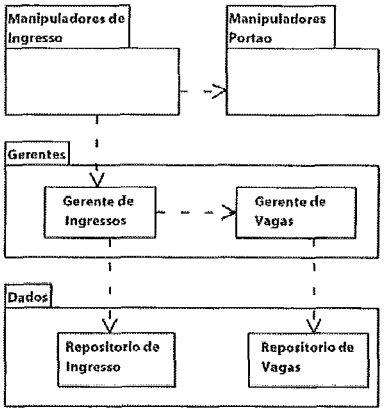


Diagrama Modular 1

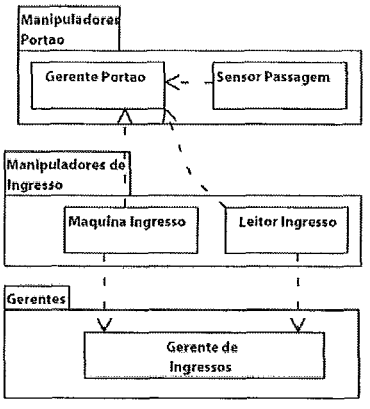


Diagrama Modular 2

Descrição

Diagrama Modular 1: Esse diagrama tem como objetivo representar os principais elementos que formam o sistema e como eles se relacionam

Diagrama Modular 2: Esse diagrama tem como objetivo representar a composição dos clusters Manipuladores Portão e Manipuladores de Ingresso, assim como a forma que eles se relacionam.

Catálogo de Elementos Arquiteturais

Elemento arquitetura	Descrição	Justificativa
Gerente de Ingressos	Módulo criado para gerenciar a disponibilização e a verificação do pagamento dos ingressos	RF16, RF22

Figura 4.2 - Exemplo de documentação de uma visão arquitetural

4.3.2 – Papéis e perfis da equipe de inspeção

Em uma inspeção de software existem principalmente três papéis que são necessários para que a revisão ocorra (KALINOWSKI, 2004):

- Moderador é o responsável por gerenciar a execução do processo. Ele é quem deve garantir que os procedimentos estipulados estão sendo seguidos e que cada membro da equipe assume as responsabilidades de seu papel. FAGAN (1976) afirma que o moderador é o elemento chave para o sucesso de uma inspeção. Para preservar a objetividade e aumentar a integridade da inspeção, normalmente é vantajoso utilizar um moderador de um outro projeto. O moderador deve gerenciar a equipe de inspeção e oferecer liderança. Para obtenção de melhores resultados, este moderador deveria receber um treinamento específico, ainda que este seja breve.
- Autor do documento é uma pessoa ou grupo de pessoas que produziu o artefato em inspeção e é responsável por corrigir os defeitos na fase de correção.

- Inspetor é o responsável por identificar os defeitos no artefato.

Em relação ao número de moderadores e de autores de documento, a realização de uma inspeção de software geralmente necessita somente de uma pessoa alocada para cada um desses papéis. Em relação aos inspetores, visto que são os responsáveis por identificar os defeitos no artefato, em teoria quanto mais pessoas fossem alocadas para esse papel maior seria o número de defeitos identificados. Contudo, o custo da avaliação poderia se tornar muito alto e, além disso, estudos experimentais mostram que não é a quantidade de inspetores que importa mas sim a utilização de profissionais com diferentes níveis de experiência (SVAHNBERG, 2003) e a realização da revisão a partir de diferentes perspectivas que influenciam no tipo e na quantidade de defeitos encontrados (SHULL, 1998).

No contexto de avaliação arquitetural, JOHANSSON *et al.* (2001) identificou que os *stakeholders* tendem a ter diferentes opiniões em relação à importância dos requisitos de qualidade, o que influenciaria na identificação de defeitos durante uma avaliação arquitetural.

Essa diferença de opiniões é um dos principais fatores que influencia na definição do número de *stakeholders* a serem utilizados em avaliações arquiteturais. Abordagens baseadas em execução de cenários, por exemplo, aconselham o uso de diferentes *stakeholders* na construção de cenários, o que pode tornar a abordagem muito custosa dependendo do tamanho do sistema a ser avaliado.

Em relação à otimização de equipes para avaliações arquiteturais, estudos experimentais já foram realizados visando identificar o perfil e a quantidade de pessoas que deveriam ser utilizadas na revisão desse tipo de artefato (SVAHNBERG, 2003).

Os resultados obtidos pelo estudo descrito em (SVAHNBERG, 2003), por exemplo, indica que é possível otimizar o número de inspetores em uma avaliação arquitetural identificando as pessoas que concordam entre si como, por exemplo arquitetos com níveis de experiência similar, por tenderem a encontrar os mesmos defeitos.

A substituição de um deles por pessoas com outros perfis poderia permitir a obtenção de um maior número de defeitos. Uma possibilidade seria alocar para a revisão um gerente, visto que por possuir um conhecimento sobre a maioria dos projetos da empresa, ele consegue identificar defeitos relacionados a *trade-offs* a nível organizacional.

Em relação à abordagem proposta nessa dissertação, nenhum estudo foi realizado visando analisar a quantidade de inspetores e de perfis a ser adotada para que a abordagem seja executada de forma eficiente. Contudo, com base nas observações relatadas por (SVAHNBERG, 2003) e visto que a abordagem proposta avalia o

documento arquitetural sob duas perspectivas, consistência de informações e corretude em relação ao atendimento dos requisitos, a alocação dos inspetores para realizar a abordagem proposta é feita através da divisão dos indivíduos disponíveis em grupos, que ficariam responsáveis pela avaliação de uma dessas perspectivas.

O grupo que avaliaria a consistência das informações poderia ser formado por indivíduos com pouco conhecimento em projeto arquitetural mas que fazem uso de documentos arquiteturais em suas atividades. Um exemplo de tipos de profissionais que possuem essas características são os desenvolvedores, os projetistas ou até mesmo gerentes de projeto.

O grupo responsável por avaliar o atendimento aos requisitos poderia ser formado por pessoas com alguma experiência em projeto arquitetural, pois teriam mais facilidade em identificar se os requisitos foram corretamente atendidos. Um exemplo de um profissional com esse perfil seria o arquiteto de software da organização. Contudo, para que ocorra uma avaliação objetiva do documento, seria interessante que esse arquiteto não tenha participado na construção da arquitetura avaliada.

Além disso, para cada grupo, o ideal seria alocar dois profissionais que possuam diferentes pontos de vista, o que aumentaria a eficiência da inspeção (TRAVASSOS *et al.*, 2002). Contudo, visto que nem todas as organizações dispõem de profissionais suficientes para serem alocadas para esse tipo de tarefa, alocar um indivíduo para cada perspectiva permite a identificação de grande parte dos defeitos.

4.3.3 – Taxonomia de defeitos

A identificação de defeitos é o principal objetivo de uma inspeção. Sendo assim, para se definir uma abordagem de inspeção, um dos passos é determinar quais são os possíveis tipos de defeitos que podem ocorrer no artefato avaliado.

Tipo de defeito não foi um dos componentes identificados em (LAITENBERGER e DEBAUD, 1998) para caracterizar uma abordagem de inspeção. Contudo, a sua definição é de extrema importância durante a construção de uma abordagem dessa natureza, pois auxilia na definição dos questionamentos que auxiliarão os inspetores a identificar defeitos (TRAVASSOS *et al.*, 2002).

Defeitos em artefatos de software podem ser classificados através de diversas taxonomias. A taxonomia que utilizamos como base é a definida por SHULL (1998), composta pelos seguintes tipos: omissão, ambigüidade, inconsistência, fato incorreto e informação estranha. Essa taxonomia é conhecida como sendo uma taxonomia não ortogonal, ou seja, é possível que um determinado defeito pertença a mais de um tipo (SHULL, 1998).

Visto que a descrição de como cada tipo de defeito ocorre é específica ao tipo de artefato inspecionado (TRAVASSOS *et al.*, 1999), além de identificar os tipos de defeitos, é necessário identificar em que consistem. Por isso, a taxonomia de defeitos utilizada foi devidamente interpretada para o contexto de Arquitetura de Software.

Portanto, no contexto de revisão de um documento arquitetural, defeitos arquiteturais podem ser identificados nos seguintes casos: (1) quando não houver uma consistência na representação dos elementos arquiteturais entre os diferentes modelos que compõem o documento, (2) quando os modelos não descreverem o mapeamento das funcionalidades em elementos arquiteturais e (3) quando os elementos arquiteturais que foram definidos, ou a forma como foram organizados, não permitirem atender aos requisitos de qualidade especificados.

Sendo assim, visando auxiliar os inspetores na identificação e categorização dos defeitos encontrados durante a atividade de identificação, a adaptação da taxonomia de defeitos proposta por (SHULL, 1998) gerou uma lista de tipos de defeitos (Tabela 4.1) que podem ser encontrados ao se revisar um documento arquitetural.

Tabela 4.1 - Taxonomia de defeitos

Tipos de defeitos	Descrição
Omissão	1. Quando um elemento arquitetural necessário para o atendimento a um requisito não foi definido;
Ambigüidade	2. Quando a forma como os elementos arquiteturais ou suas responsabilidades foram definidos dificulta ou impossibilita o atendimento a um requisito de qualidade. 3. Quando elementos descritos em visões distintas possuem o mesmo nome, mas responsabilidades diferentes (Homônimo);
Inconsistência	4. Quando elementos descritos em visões distintas possuem mesma responsabilidade, mas nomes distintos (Sinônimo); 5. Quando um elemento arquitetural presente em diagramas das demais visões não foi definido no diagrama avaliado; 6. Quando a representação não condiz com a semântica estabelecida pela abordagem de documentação. 7. Quando um elemento arquitetural é definido com responsabilidades distintas em duas ou mais visões. 8. Quando um elemento é representado de maneira diferente em duas visões.
Fato Incorreto	9. Quando um elemento não foi descrito ou representado de forma correta 10. Quando não é possível mapear um elemento arquitetural para algum elemento descrito em outra visão.
Informação Estranha	11. Quando não é possível determinar o papel de um elemento arquitetural ou de uma de suas responsabilidades no atendimento aos requisitos especificados.

4.3.4 – Técnica de inspeção utilizada

O principal diferencial de uma abordagem de inspeção está na maneira como ela auxilia o inspetor a encontrar os defeitos. O que determina essa maneira é a técnica de detecção de defeitos utilizada.

Para ArqCheck, a técnica de detecção de defeitos utilizada foi o *checklist*. *Checklist* foi escolhido entre as técnicas existentes por ser mais adequado às

características da área de Arquitetura de Software e por permitir minimizar as limitações identificadas nas abordagens de avaliação arquitetural identificadas na literatura (BARCELOS e TRAVASSOS, 2005).

A seguir, mostramos porque a técnica *checklist* foi escolhida em detrimento da técnica *ad-hoc* ou da técnica de leitura. Além disso, mostramos que melhorias foram realizadas no *checklist* visando minimizar as limitações identificadas nas demais abordagens de avaliação arquitetural. Descrevemos também quais os itens de avaliação que compõem esse *checklist* e como eles foram criados.

4.3.4.1 – Checklist versus demais técnicas de inspeção

Diversas técnicas de inspeção de software já foram definidas visando auxiliar na identificação de defeitos. A principal característica que diferencia essas técnicas é o nível de formalidade usado durante a atividade de identificação. Esse nível de formalidade permite as categorizar em três tipos: *Ad-hoc*, *Checklist* e Técnicas de leitura.

Técnicas *ad-hoc* são as técnicas mais simples para identificar defeitos em artefatos de software. Elas não oferecem nenhum tipo de apoio ou procedimento de execução formal e sistemático de inspeção. Sendo assim, os defeitos identificados dependem exclusivamente da capacidade, competência e experiência do inspetor (CHEN *et al.*, 2002).

Um dos pontos negativos em usar técnicas *ad-hoc* é o fato delas não oferecerem auxílio aos inspetores para identificar defeitos. Em um contexto arquitetural, por exemplo, os requisitos de qualidade exercem uma grande influência na definição de uma arquitetura, porém o uso de uma técnica *ad-hoc* para avaliá-la não forneceria nenhum auxílio visando garantir que os inspetores avaliem o atendimento a esses requisitos.

Uma evolução às técnicas *ad-hoc* são as baseadas em *checklist*. Ao se utilizar *checklist* para revisar artefatos de software, é fornecido ao inspetor um conjunto de questionamentos que o auxiliam a identificar em que parte do artefato avaliado ele deve procurar por defeitos (SHULL *et al.*, 2000).

Utilizar esse tipo de técnica em inspeções arquiteturais consiste em definir questionamentos que, por exemplo, indicariam ao inspetor como identificar os elementos arquiteturais que devem ser analisados, visando avaliar o atendimento a um determinado requisito.

Um outro possível tipo de técnica que pode ser usado em inspeção são as técnicas de leitura. Técnicas de leitura são procedimentos que visam guiar individualmente os inspetores no entendimento de um artefato de software e, por conseqüência, na

identificação de discrepâncias (SHULL *et al.*, 2000). Abordagens de avaliação baseadas nessa técnica (SHULL *et al.*, 2000; CONRADI *et al.*, 2003) são mais eficientes na detecção de defeitos quando comparadas a outras técnicas de inspeção, como *checklists*, por exemplo.

Porém, para que esse tipo de técnica seja utilizado, o artefato deve ser representado em uma forma específica e padronizada, característica que ainda não pode ser atingida na Área de Arquitetura de Software por não existir uma padronização para se representar documentos arquiteturais.

Atualmente, se tem notícia de somente uma abordagem baseada em técnicas de leitura para avaliar documentos arquiteturais, intitulada de Architecture Reading Techniques - ARTs (CARVER e LEMON, 2005)[6]. Uma característica dessa abordagem é a necessidade da arquitetura estar documentada seguindo a abordagem apresentada em (CLEMENTS *et al.*, 2004), o que pode dificultar aplicar essa revisão em arquiteturas que não tenham sido documentadas desta forma.

Em suma, a principal diferença entre os três tipos de técnicas de inspeção apresentados está no tipo de auxílio que eles oferecem aos inspetores para identificar defeitos. A técnica *ad-hoc* não oferece nenhum auxílio ao inspetor durante a revisão do artefato visando a identificação de defeitos, a técnica baseada em *checklist* auxilia o inspetor na identificação do que deve ser revisado e a técnica de leitura, além de auxiliar na identificação, auxilia também na forma como o artefato deve ser revisado para que os defeitos sejam identificados. Na Tabela 4.2, é feita uma comparação entre as diferentes técnicas de inspeção usando outros critérios (SHULL *et al.*, 2000; SILVA, 2004) que permitem identificar o contexto em que cada tipo de técnica pode ser usada.

Tabela 4.2 - Características das técnicas de inspeção (SILVA, 2004)

Características	*Ad-hoc*	*Checklist*	Técnicas de leitura
Linguagem: Em que linguagem ou notação os documentos devem estar escritos?	Qualquer	Qualquer	Específica (ex: *Architecture Description Language* - ADL específica)
Sistematização: Os passos específicos do processo de revisão individual são definidos?	Não	Parcialmente	Sim
Foco: Cada revisor deve focar em diferentes aspectos do documento?	Não	Não	Sim
Aprimoramento Contínuo: A partir da realimentação dos revisores, aspectos da técnica podem ser melhorados?	Não	Parcialmente	Sim
Adaptação: A técnica pode ser adaptada para projetos ou organizações específicas?	Não	Sim	Sim
Treinamento: A técnica permite e	Não	Parcialmente	Sim

[6] A abordagem ARTs não foi identificada pela revisão sistemática visto que a abordagem foi publicada no ISESE'05, cujos Anais ainda não tinham sido disponibilizados nas fontes de pesquisa que utilizamos no momento da realização da revisão.

necessita de treinamento?			

Portanto, devido às características da área de Arquitetura de Software, a técnica de *checklist* possui características que a tornam mais adequada para se aplicar em uma inspeção de documento arquitetural.

Várias abordagens baseadas em *checklist* já foram definidas visando a identificação de defeitos em documentos arquiteturais. Contudo, os questionamentos feitos por esses *checklists* ou são específicos ao domínio do software avaliado (HOLLOCKER, 1990; NASA, 1993), o que dificulta a sua aplicação em um contexto diferente do qual foi projetado, ou então são muito específicos à solução (ERICKSON *et al.*, 1993), o que requer que a cada avaliação um novo *checklist* seja criado.

Com isso, algumas melhorias devem ser feitas para que uma técnica de *checklist* possa ser utilizada de forma efetiva e escalável, permitindo assim a sua aplicação em contexto industrial.

4.3.4.2 – Melhorias realizadas no checklist para inspeção arquitetural

ArqCheck objetiva identificar defeitos arquiteturais usando como guia os questionamentos definidos em um *checklist* de avaliação. A nossa hipótese é que o uso de *checklist* como técnica de inspeção e o tipo de conhecimento utilizado para definir esse *checklist* possibilitam identificar defeitos arquiteturais relacionados ao atendimento dos requisitos do software e permitem minimizar as limitações identificadas nas abordagens de avaliação existentes (BARCELOS e TRAVASSOS, 2005).

Para que essa abordagem pudesse utilizar um *checklist* para auxiliar na identificação de defeitos, identificamos alguns requisitos que devem ser atendidos pelo *checklist* para que limitações identificadas nas abordagens de avaliação arquitetural sejam minimizadas. Essas características são:

- Os itens de avaliação que compõem o *checklist* devem usar conceitos oriundos das atividades de especificação arquitetural, levando em consideração principalmente a forma como os requisitos de qualidade são atendidos. Com isso, pretende-se reduzir a subjetividade em relação ao que é avaliado;
- Esses itens devem ser agrupados pelo tipo de requisito de qualidade que se busca avaliar o atendimento. Com isso, é possível definir exatamente que características de qualidade estão sendo avaliadas ao usar o *checklist*;
- Fácil execução da avaliação, sem a necessidade de elaboradas atividades, como as necessárias para a especificação de cenários, por exemplo, permitindo a realização de avaliações com baixos custos.

- Os itens de avaliação devem avaliar as informações contidas no documento arquitetural, ficando assim independente da forma como essas informações foram representadas. Com isso, é possível aplicar a abordagem proposta em documentos que pertencem a diferentes contextos e que utilizam diferentes abordagens de documentação;

4.3.4.3 – Descrição do checklist

O *checklist* que compõem ArqCheck, além de atender aos requisitos descritos na seção anterior, foi criado com base em conceitos presentes na família de técnica de leitura conhecida como *Object Oriented Reading Techniques* - OORTs (TRAVASSOS *et al.*, 2002). Essa família de técnica de leitura foi escolhida devido às similaridades existentes entre arquiteturas de software e documentos de projeto orientado a objetos, artefato avaliado por OORTs.

Após uma análise de como OORTs avalia os documentos de projeto, percebemos que também seria possível utilizar a abordagem de rastreabilidade de informações utilizada por OORTs (TRAVASSOS *et al.*, 1999), na avaliação de documentos arquiteturais.

Ao adaptar essa abordagem de rastreabilidade para o contexto de Arquitetura de Software, definimos que a avaliação do documento arquitetural pode ser feita através (1) do rastreamento das informações que são comuns às diferentes visões que o compõe, visando garantir a consistência do documento, e (2) da análise das informações descritas nesse documento com as que estão presentes na especificação dos requisitos, visando garantir a corretude e a completude da arquitetura avaliada.

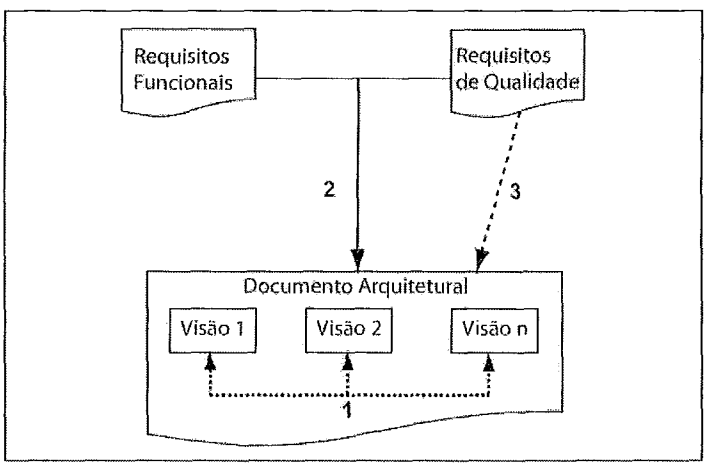
Além disso, devido às características da área de Arquitetura de Software, como a falta de padronização entre abordagens de documentação arquitetural, criamos o *checklist* de forma que ele pudesse ser configurável, para que se adapte à arquitetura avaliada e possa ser utilizado para avaliar documentos arquiteturais oriundos de diversos contextos.

Sendo assim, criamos itens de avaliação para compor o *checklist* com o objetivo de avaliar:

- Se as informações descritas no documento arquitetural estão consistentes entre si. Essa avaliação é feita devido ao uso de diferentes visões arquiteturais para representar um mesmo conjunto de elementos arquiteturais através de diferentes perspectivas. Sendo assim, é necessário avaliar a consistência das informações comuns a essas visões;

- Se todas as funcionalidades especificadas foram atendidas pela arquitetura. Essa avaliação é feita com o objetivo de identificar defeitos relacionados a rastreabilidade entre os requisitos e os elementos arquiteturais;
- Se todos os elementos arquiteturais foram definidos com base nos requisitos especificados. Essa avaliação também é feita com o objetivo de analisar a rastreabilidade requisitos-elementos arquiteturais;
- Se a arquitetura atende aos requisitos de qualidade especificados para o produto. Essa avaliação é feita devido à importância dos requisitos de qualidade na definição da arquitetura.

Com isso, agrupamos os itens de avaliação que criamos em três grupos (Figura 4.3): (1) itens que avaliam a consistência do documento, (2) itens que avaliam o atendimento aos requisitos e (3) itens que avaliam a abordagem utilizada para atender aos requisitos de qualidade.



Legenda

1 - Itens que avaliam a consistência do documento
2 - Itens que avaliam o atendimento aos requisitos
3 - Itens que avaliam a abordagem para atender aos requisitos de qualidade

Figura 4.3 - Grupos de itens de avaliação que compõem o *checklist* proposto

A seguir descrevemos como cada um desses grupos foi criada. No Apêndice B.3, a versão final do *checklist* pode ser encontrada.

Itens que avaliam a consistência do documento

Esse conjunto de itens busca avaliar se todas as visões que compõem o documento arquitetural estão representando a mesma arquitetura.

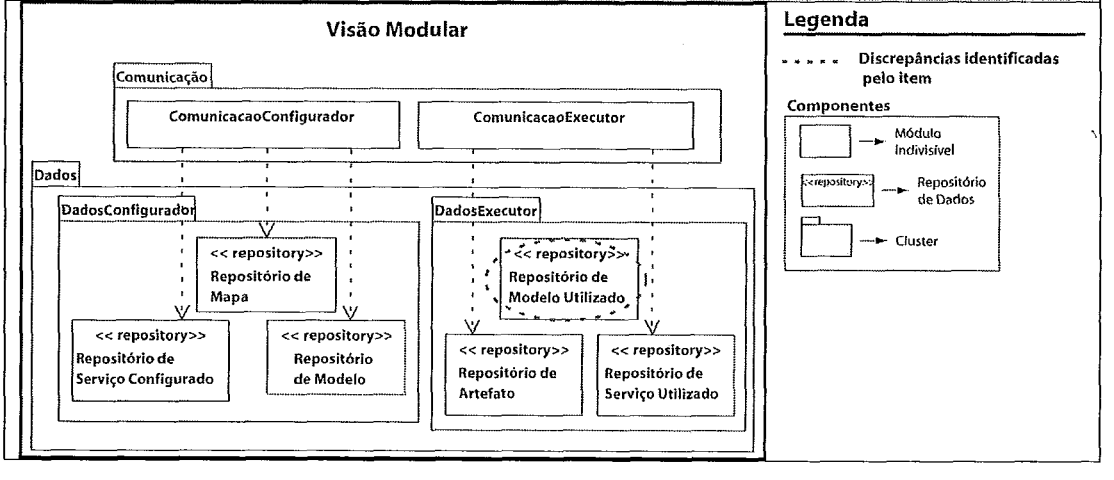
Visto que o papel das visões arquiteturais é representar determinada arquitetura sob diferentes perspectivas, definimos itens para avaliar a consistência das informações de uma visão arquitetural e entre as diferentes visões.

A avaliação realizada por esses itens consiste principalmente em analisar os diagramas gráficos que compõem o documento arquitetural. Visto que esses diagramas são específicos à abordagem de documentação utilizada, ou seja, dependem da semântica atribuída aos símbolos gráficos que descrevem a arquitetura, houve necessidade em se criar itens de avaliação específicos à abordagem de representação gráfica utilizada, formando assim dois grupos de itens.

O primeiro grupo é formado por itens que buscam identificar a consistência em relação a conceitos básicos de especificação arquitetural, o que os deixa independente da representação gráfica utilizada. Um exemplo de avaliação feita por itens que pertencem a esse grupo é a identificação de módulos arquiteturais que nunca se relacionam com outro elemento arquitetural (Tabela 4.3). A presença de um módulo com essa característica indica que possivelmente algum erro foi cometido no projeto arquitetural pois ele não exerce nenhuma função na arquitetura projetada.

Tabela 4.3 - Exemplo de item que avalia consistência do documento de forma independente à representação gráfica utilizada

Item de avaliação de consistência	
Descrição do item	Ao analisar todos os diagramas, foi identificado algum elemento arquitetural que não possua relacionamentos, ficando isolado dos demais?
Resposta Esperada	Não
Tipo de Defeito	Informação Estranha ou Fato Incorreto
Objetivo	Identificar elementos inseridos que foram inseridos na arquitetura sem necessidade ou então identificar erros no projeto da arquitetura
Exemplo	



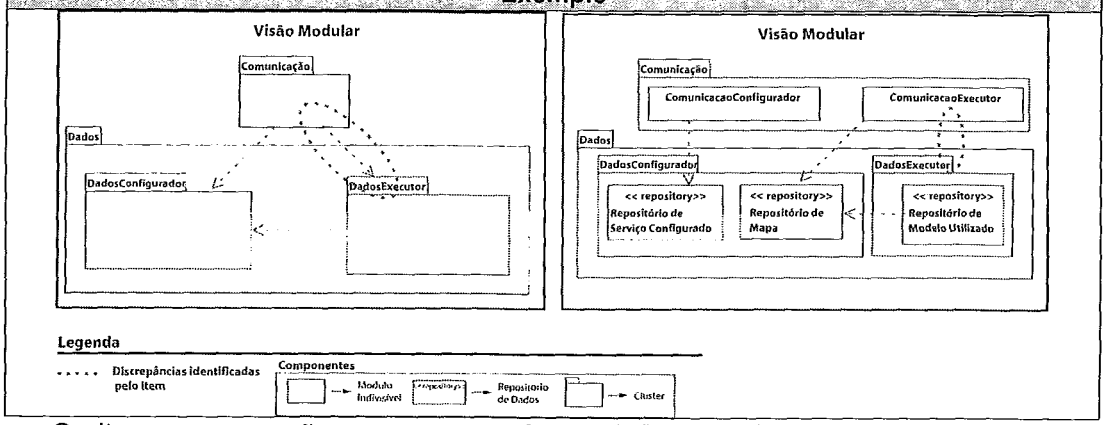
O segundo grupo é formado por itens que são específicos à abordagem de representação utilizada. Esses itens são utilizados para avaliar principalmente a consistência das informações dentro de cada uma das visões. Na Tabela 4.4, descrevemos um item que pertence a esse grupo.

Para cada nova abordagem de documentação arquitetural, é necessário analisar os tipos de visão arquitetural que a abordagem define e também as regras de

consistência pertencentes a cada visão. É com base nessas informações, que os itens de avaliação pertencentes a esse grupo são criados. O principal objetivo deles é levar o inspetor a avaliar se, para cada visão, as regras de consistência estão sendo respeitadas e se as informações estão consistentes entre as diferentes visões.

Tabela 4.4 - Exemplo de item que avalia consistência do documento e que é específico à representação gráfica utilizada

Item de avaliação de consistência	
Descrição do item	Todo relacionamento definido entre dois *clusters* pode ser mapeado para seus respectivos módulos?
Resposta Esperada	Sim
Tipo de Defeito	Inconsistência
Objetivo	Identificar se todos os relacionamentos definidos entre elementos de alto nível estão mapeados para os elementos de baixo nível
Exemplo	



Os itens que compõem esse grupo foram definidos principalmente com base em conceitos relacionados à documentação de visões arquiteturais extraídos de (CLEMENTS *et al.*, 2004). Portanto, esses itens avaliam a consistência entre quatro tipos de visão: modular, dinâmica, de alocação e de contexto geral.

A avaliação realizada por eles é de extrema relevância para a melhoria da qualidade do documento arquitetural pois buscam identificar defeitos que poderiam dificultar a implementação do software a partir da arquitetura. Além disso, eles permitem uma avaliação direcionada, pois não fica a cargo do inspetor determinar o que deve ser avaliado.

Devido à presença de itens específicos à abordagem de representação arquitetural utilizada, é necessário que, antes da avaliação, seja realizada uma análise visando determinar os itens que são aplicáveis ou não para a arquitetura a ser avaliada. Essa análise é feita durante a configuração do *checklist* [7].

Além de avaliar se as diferentes visões que formam um documento arquitetural estão representando a mesma arquitetura, é preciso avaliar também se a arquitetura descrita no documento representa corretamente o sistema descrito pelos requisitos.

[7] A configuração do *checklist* é descrita com mais detalhes na seção 4.3.4 desse capítulo.

Para isso criamos dois grupos de itens, um responsável por avaliar o atendimento aos requisitos e um outro responsável por avaliar abordagem utilizada para atender aos requisitos de qualidade.

Itens que avaliam o atendimento aos requisitos

A principal forma de avaliar o atendimento aos requisitos consiste em verificar se existe rastreabilidade entre os requisitos e os elementos arquiteturais. Por isso, definimos um conjunto de itens de avaliação que avaliam essa rastreabilidade.

Nesses itens, foi dada ênfase principalmente na avaliação do atendimento aos requisitos funcionais, visto que itens específicos também foram criados para avaliar o atendimento a requisitos de qualidade. Sendo assim, esses itens procuram avaliar se algum requisito relevante a nível arquitetural não foi atendido pela arquitetura, devido à falta de atenção por parte do arquiteto, por exemplo.

Além disso, eles procuram identificar também se elementos arquiteturais foram definidos sem haver necessidade de sua existência. Isso ocorre, por exemplo, quando os arquitetos fazem uso de conceitos que não são diretamente relevantes ao domínio do problema e que acabariam introduzindo complexidade desnecessária ao projeto.

Portanto, os itens de avaliação que pertencem a esse grupo permitem a realização de dois tipos de análise: (1) se todas as funcionalidades especificadas foram atendidas pela arquitetura e (2) se todos os elementos arquiteturais foram definidos com base nos requisitos especificados (Tabela 4.5).

Itens que avaliam a abordagem utilizada para atender aos requisitos de qualidade

A avaliação do atendimento a requisitos de qualidade é de grande importância para a garantia da qualidade do documento arquitetural (BASS *et al.*, 2003), com isso, itens de avaliação foram criados especialmente para se a abordagem usada pelos arquitetos realmente atende aos requisitos de qualidade especificados.

Visto que nem todos os requisitos de qualidade podem ser atendidos durante o projeto arquitetural, procuramos identificar, em um primeiro momento, quais são os tipos de requisito de qualidade que são relevantes a nível arquitetural.

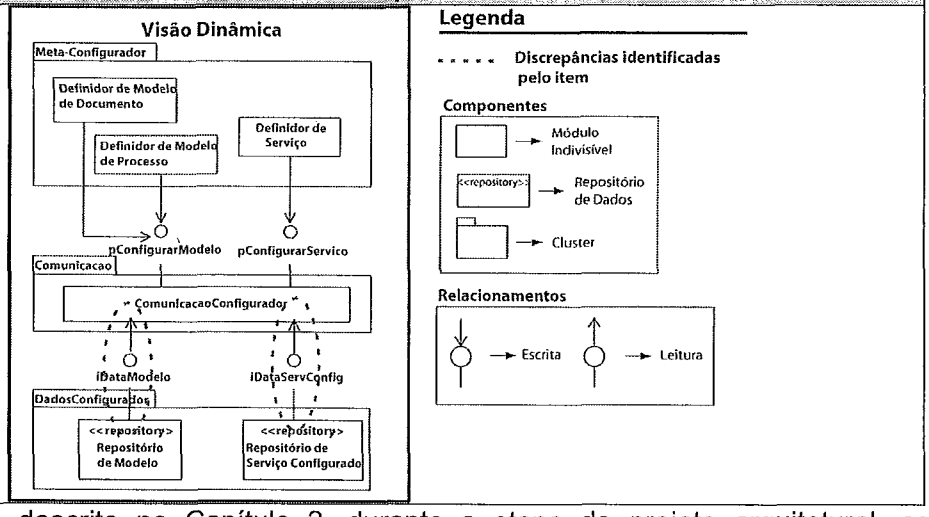
Para isso, foi feita uma análise das taxonomias existentes utilizadas para caracterizar esse tipo de requisito (ISO/IEC, 1998; BASS *et al.*, 2003). Com essa análise, identificamos que a taxonomia definida por BASS *et al.* (2003) identifica os tipos de requisitos de qualidade que são relevantes a esse nível de abstração. Visto isso, a utilizamos como base para o nosso trabalho.

Essa taxonomia nos orientou em relação ao tipo de avaliação que deveríamos fazer. A partir dela, identificamos que para permitir a avaliação do atendimento a requisitos de qualidade, deveríamos avaliar especificamente o atendimento a requisitos de Modificabilidade, Desempenho, Usabilidade, Testabilidade, Disponibilidade e Segurança.

Além de definir que tipo de requisitos de qualidade avaliar, precisávamos definir como avaliar a arquitetura em relação ao atendimento a esses requisitos. Para isso optamos por utilizar conhecimentos provenientes de práticas usadas durante o projeto arquitetural.

Tabela 4.5 - Exemplo de item que avalia o atendimento aos requisitos

Item de avaliação do atendimento aos requisitos	
Requisito Avaliado	A infra-estrutura deve permitir a persistência de todos os modelos e artefatos por ela manipulados
Descrição do Item	As responsabilidades dos elementos arquiteturais estão condizentes com os requisitos que eles atendem?
Resposta Esperada	Sim
Tipo de Defeito	Fato Incorreto
Observação	A simbologia utilizada representa que os dados no repositório podem ser somente acessados e não alterados, o que vai de encontro com o requisito especificado.
Exemplo	



Conforme descrito no Capítulo 2, durante a etapa de projeto arquitetural, os arquitetos utilizam como abordagem para atender aos requisitos de qualidade a aplicação de estilos e táticas arquiteturais.

Visto que as táticas arquiteturais são conhecimentos mais abstratos, simples e são utilizados na criação dos estilos arquiteturais, optamos por utilizá-las como principal fonte de conhecimento para a criação dos itens de avaliação do *checklist* que ficaram responsáveis por avaliar essas características da arquitetura.

As táticas arquiteturais consistem em um conjunto de diretrizes arquiteturais baseados em boas práticas, que auxiliam no atendimento a requisitos de qualidade que influenciam o projeto de uma arquitetura (TVEDT *et al.*, 2002). Algumas pesquisas

mostram indícios que as informações contidas nesse tipo de conhecimento são úteis para apoiar o processo de avaliação arquitetural (ZHU *et al.*, 2004).

As táticas que utilizamos para a criação dos itens de avaliação estão descritas em (BASS *et al.*, 2003). No Anexo B, descrevemos resumidamente as táticas que utilizamos para a criação do *checklist.*

Com base nessas táticas e nos tipos de requisitos de qualidade que elas atendem foi possível criar a Tabela 4.6, que identifica os tipos de requisitos de qualidade que são avaliados pelo *checklist* proposto e as táticas arquiteturais que foram usadas como base na criação dos itens que o compõem.

Tabela 4.6 - Táticas arquiteturais utilizadas na definição do *checklist*

Tipos de requisitos de qualidade atendidos pelos itens	Táticas arquiteturais usadas na definição dos itens
Desempenho	Controle da geração de eventos Controle da freqüência de eventos externos Redução do overhead computacional
Disponibilidade	Ping/echo Heartbeat Redundância Ativa
Modificabilidade	Manter a coerência semântica Isolar serviços comuns Isolar candidatos a alterações Abstrair informações Separar as interfaces da implementação do módulo Usar intermediários
Segurança	Autenticação de usuários Limitar o acesso
Testabilidade	Separar a interface da implementação Interfaces específicas de acesso Monitores
Usabilidade	-

Para cada grupo de itens que avaliam o atendimento aos requisitos de qualidade foi criado uma "Guia de identificação de contexto", que utiliza a descrição do requisito a ser avaliado para identificar os elementos arquiteturais relacionados ao atendimento de determinado requisito de qualidade. Ao identificar esses elementos, o inspetor consegue avaliar através dos itens de avaliação se os elementos arquiteturais foram definidos de forma a permitir o atendimento ao requisito.

A principal contribuição desses itens está no fato do uso de agrupamentos permitir a avaliação de um ou mais requisitos de qualidade, identificados como prioritários pelos *stakeholders*.

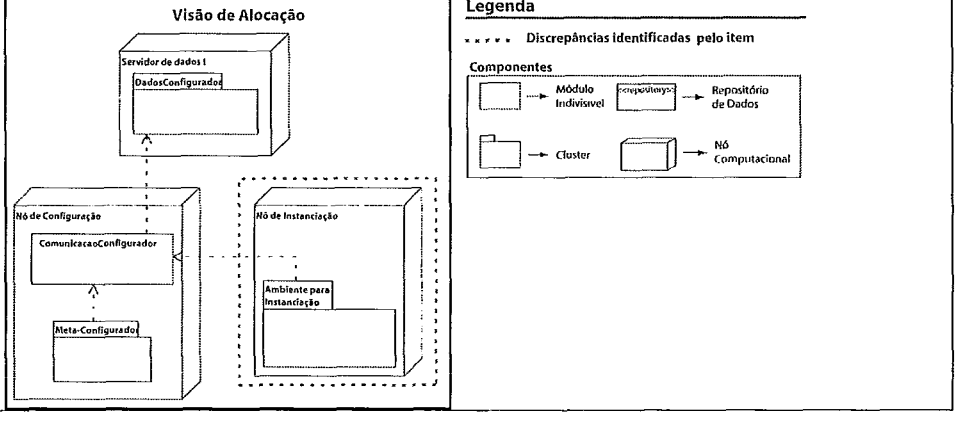
Na Tabela 4.7 pode ser observado um exemplo de item de avaliação de qualidade. Nesse exemplo, um item de desempenho é usado para avaliar um diagrama de acordo com o requisito de desempenho especificado.

4.3.5 – Processo de inspeção utilizado

Um processo de software consiste em uma série de passos envolvendo atividades, restrições e recursos que produzem um resultado esperado (PFLEEGER, 2004).

Abordagens de inspeção são normalmente executadas a partir de um processo que identifica as atividades utilizadas principalmente para detectar defeitos no artefato e que define os papéis que os indivíduos deveram exercer durante a realização dessas atividades.

Tabela 4.7 - Exemplo de item que avalia a abordagem utilizada para atender a um requisito de qualidade

<table>
<tr><th colspan="2">Item de avaliação do atendimento aos requisitos</th></tr>
<tr><td>Requisito de desempenho avaliado</td><td>A infra-estrutura deve permitir a instanciação de um Ambiente para Execução em menos de 10 segundos</td></tr>
<tr><td>Cenário de qualidade</td><td>
<table>
<tr><th colspan="2">Cenário de Desempenho</th></tr>
<tr><td>Fonte do estímulo</td><td>Usuário Engenheiro</td></tr>
<tr><td>Estímulo</td><td>Instanciar Ambiente para Execução</td></tr>
<tr><td>Contexto do sistema</td><td>Durante a execução normal da infra-estrutura</td></tr>
<tr><td>Artefato</td><td>Sistema</td></tr>
<tr><td>Resposta</td><td>Ambiente para Execução instanciado</td></tr>
<tr><td>Medida de resposta</td><td>Menos de 10 segundos</td></tr>
</table>
</td></tr>
<tr><td>Guia de Identificação de Contexto</td><td>De acordo com os requisitos de desempenho selecionados, identifique as seqüências de execução do sistema (Seqüência), assim como os elementos arquiteturais (Elementos Participantes) que a compõem. Essa identificação pode ser feita através da análise do estímulo, do artefato e da resposta descritos nos cenários de desempenho.</td></tr>
<tr><td>Descrição do Item</td><td>Todos os Elementos Participantes foram alocados em um mesmo nó computacional visando aumentar o desempenho da Seqüência?</td></tr>
<tr><td>Resposta Esperada</td><td>Sim</td></tr>
<tr><td>Tipo de Defeito</td><td>Fato Incorreto</td></tr>
<tr><td>Observação</td><td>Para a execução da seqüência de instanciação, os elementos que participam dessa seqüência estão alocados em três nós computacionais diferentes, o que pode atrapalhar o atendimento ao requisito de desempenho especificado.</td></tr>
<tr><th colspan="2">Exemplo</th></tr>
<tr><td colspan="2"> 
</td></tr>
</table>

O processo utilizado por ArqCheck consiste em uma adaptação do processo tradicional de inspeção definido por FAGAN (1976).

Devido às características do *checklist* proposto, algumas modificações tiveram que ser feitas no processo de inspeção adotado. Essas modificações consistem principalmente na introdução de atividades necessárias para realizar as devidas configurações no *checklist* e de atividades que devem ser executadas pelo inspetor durante a identificação de discrepâncias.

A seguir, descrevemos as principais atividades que compõem o processo a ser seguido para realizar a abordagem para inspeção proposta e detalhamos as modificações que foram realizadas quando necessário.

4.3.5.1 – Descrição das atividades do processo de inspeção

O processo de inspeção que utilizamos é composto pelas seguintes atividades (Figura 4.4): Planejamento, Apresentação, Detecção, Reunião de inspeção, Retrabalho e Continuação.

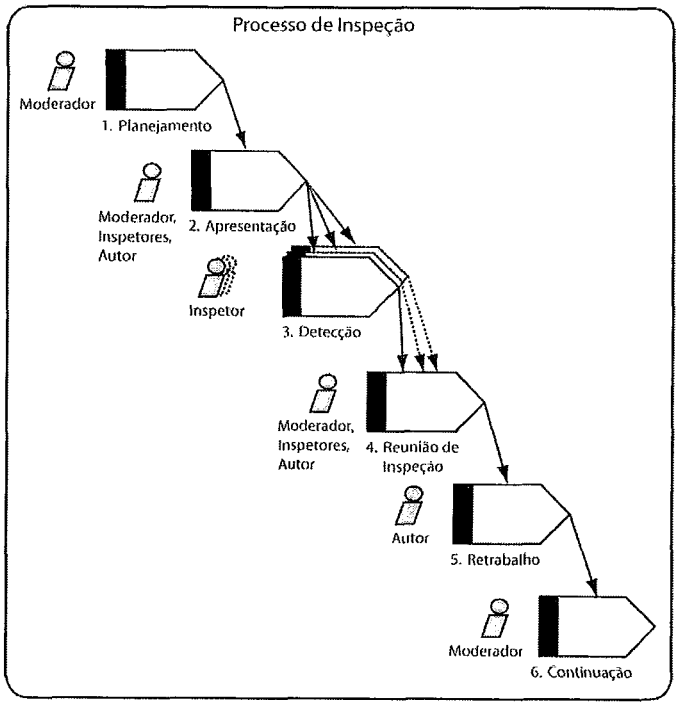


Figura 4.4 - Processo de inspeção utilizado

Planejamento

Nessa atividade, um profissional é identificado como moderador da inspeção. Durante o Planejamento, esse moderador deve definir o contexto em que a inspeção vai ser realizada. Para isso, ele define os participantes, identifica o artefato a ser avaliado e distribui o material aos inspetores.

Além dessas atividades que devem ser realizadas durante o planejamento de uma inspeção, realizamos algumas modificações visando inserir atividades específicas às características da abordagem de inspeção proposta.

A principal característica que motivou essa modificação foi a necessidade de configurar o *checklist* antes de realizar a avaliação do documento arquitetural. Sendo assim, incluímos um conjunto de atividades para permitir a realização dessa configuração (Figura 4.5).

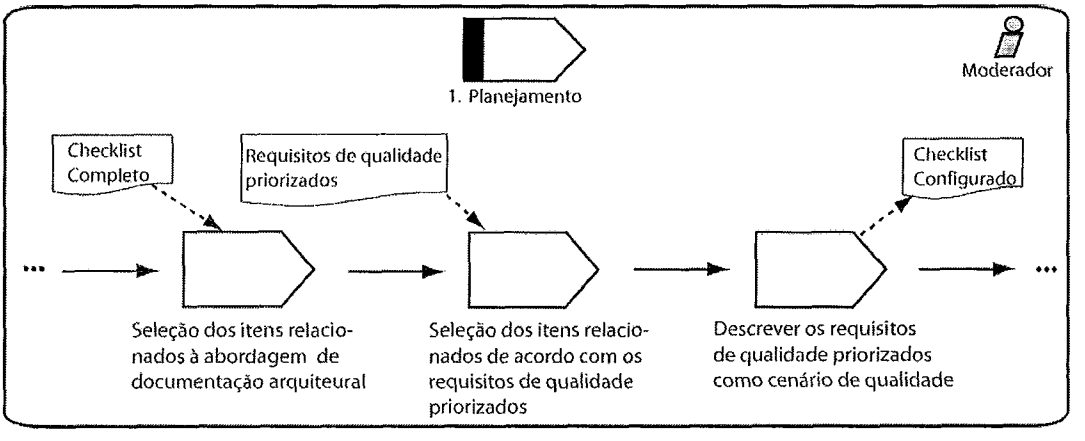


Figura 4.5 – Sub-atividades adicionadas na atividade de planejamento visando configuração do *checklist*

A configuração do *checklist* consiste em selecionar os itens que são aplicáveis ao documento arquitetural a ser avaliado. Itens aplicáveis são itens específicos que são utilizados na inspeção de acordo com a abordagem de representação arquitetural utilizada e de acordo com os tipos de requisitos de qualidade que se deseja avaliar.

Os principais motivos que levaram a criação desses itens aplicáveis são:

- A ausência de padronização na documentação da arquitetura, necessitando a criação de itens genéricos e de itens que sejam específicos à abordagem de documentação utilizada;
- O uso de táticas arquiteturais como base para a criação de itens de avaliação. Visto que existe um conjunto de táticas que auxilia o atendimento a cada um dos tipos de requisitos de qualidade relevantes a nível arquitetural, os itens foram criados seguindo essa mesma política. Contudo, nem toda arquitetura necessita atender a todos esses tipos de requisitos, somente aos que foram especificados, com isso não precisam ser utilizados durante a avaliação.

Sendo assim, durante o planejamento, o moderador deverá também:

- Selecionar os itens de avaliação relacionados à abordagem de documentação arquitetural utilizada;

- Selecionar os itens de avaliação relacionados aos requisitos de qualidade priorizados. Para isso os *stakeholders* devem priorizar os requisitos de qualidade que serão utilizados para avaliar a arquitetura em relação ao seu atendimento;
- Descrever os requisitos de qualidade selecionados como cenários de qualidade. Isso é feito visando complementar a "Guia de identificação de contexto" e auxiliar os inspetores na identificação dos elementos arquiteturais utilizados no atendimento a determinado requisito de qualidade. Para facilitar essa identificação, foi adotado o uso de cenários de qualidade (BASS *et al.*, 2003) que permite caracterizar um requisito de qualidade e facilitar o seu entendimento.

Como principal resultado desse conjunto de atividades obtemos o *checklist* configurado. Na Figura 4.6, é apresentada uma parte de um *checklist* configurado. Nesse exemplo, os itens selecionados são itens que avaliam o atendimento ao requisito de modificabilidade especificado.

<table>
<tr><th>Nº</th><th>Itens relacionados a Modificabilidade</th><th>Sim</th><th>Não</th><th>RE</th></tr>
<tr><td colspan="5">Requisito de Modificabilidade(RNF3)
A infra-estrutura deve ser construída de forma a permitir que os desenvolvedores modifiquem a abordagem de recuperação e persistência dos artefatos manuseados, sem que seja necessário realizar alterações em outras funcionalidades. Essa alteração pode ocorrer durante o desenvolvimento do sistema ou durante a sua manutenção.
<table>
<tr><th colspan="2">Cenário de Modificabilidade</th></tr>
<tr><td>Fonte do estímulo</td><td>Desenvolvedores</td></tr>
<tr><td>Estímulo</td><td>Modificação da abordagem de manipulação de dados</td></tr>
<tr><td>Contexto do sistema</td><td>Desenvolvimento do sistema / Manutenção do sistema</td></tr>
<tr><td>Artefato</td><td>Repositórios de Dados</td></tr>
<tr><td>Resposta</td><td>O sistema deve manipular os dados como antes da modificação</td></tr>
<tr><td>Medida de resposta</td><td>---</td></tr>
</table></td></tr>
<tr><td colspan="5">Guia de identificação de contexto: De acordo com os requisitos de modificabilidade que se deseja avaliar, identificar os elementos arquiteturais que podem ser modificados (Elemento Modificável) e também os elementos que possuem relacionamentos com os Elementos Modificáveis de forma direta (Elemento Relacionado). A identificação dos Elementos Modificáveis é feita principalmente através da análise do estímulo e contextodos requisitos selecionados.</td></tr>
<tr><td>29</td><td>As responsabilidades especificadas para cada Elemento Relacionado pertencem a um mesmo contexto, ou seja, visam atingir um mesmo propósito, manipulam um mesmo tipo de dado ou são utilizadas em uma mesma seqüência de execução?</td><td></td><td></td><td>S</td></tr>
<tr><td>30</td><td>Elementos Relacionados possuem responsabilidades comuns ou similares que podem ser alocadas em um único elemento?</td><td></td><td></td><td>N</td></tr>
<tr><td>31</td><td>Todo Elemento Modificável disponibiliza interfaces para se comunicar com os demais elementos?</td><td></td><td></td><td>S</td></tr>
<tr><td>32</td><td>A comunicação entre um Elemento Relacionado e um Elemento Modificável, e as funcionalidades disponibilizadas pelo conector (interface/porta) que intermedia essa comunicação são realmente necessárias?</td><td></td><td></td><td>S</td></tr>
<tr><td>33</td><td>Elementos intermediários (ex: máquina virtual, servidor de nomes, repositórios) são utilizados como mediadores de comunicação entre o Elemento Modificável e os demais elementos arquiteturais, diminuindo o impacto das alterações realizadas nos Elementos Modificáveis. Existe algum Elemento Relacionado que realize esse papel de intermediário?</td><td></td><td></td><td>S</td></tr>
</table>

Figura 4.6 - Exemplo de itens pertencentes a um *checklist* configurado

Apresentação

O profissional responsável por realizar essa fase é o autor. Neste momento, ele apresenta as características dos artefatos a serem inspecionados. Esta fase pode ser

omitida se os inspetores possuem conhecimento sobre o projeto e os artefatos que devem ser inspecionados.

Além disso, durante essa fase é realizado pelo moderador o treinamento no *checklist* de inspeção arquitetural. Durante esse treinamento, é passado para os inspetores (1) os conceitos presentes no *checklist*, (2) como deve ser feita a identificação e o registro de defeitos e (3) é apresentada a taxonomia de classificação de defeitos que deve ser usada para classificá-los.

Detecção

Nessa atividade, os inspetores individualmente revisam o artefato identificando os defeitos. Devido às características do *checklist* que compõem ArqCheck, sub-atividades foram definidas visando auxiliar o inspetor na identificação de defeitos. Essas sub-atividades determinam a ordem de execução dos itens de avaliação (Figura 4.7).

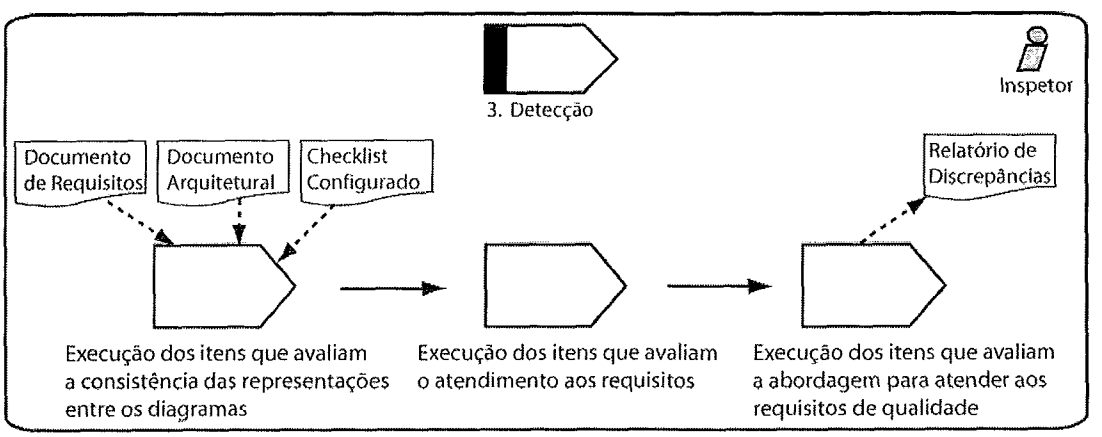


Figura 4.7 – Atividades de detecção de defeitos

O motivo em estabelecer essa ordem está relacionado à ocorrência de inconsistências nas representações que podem influenciar na identificação de possíveis inconsistências relacionadas ao atendimento dos requisitos, por exemplo.

Durante a execução da atividade de Detecção, o inspetor deve registrar as discrepâncias em um relatório de discrepâncias (vide Apêndice C) e classificá-las de acordo com o tipo de defeito que elas podem gerar, usando para isso a taxonomia de defeitos que fora definida.

Reunião de inspeção

Durante essa atividade, uma reunião de equipe ocorre envolvendo o moderador, os inspetores e os autores do documento. Discrepâncias são discutidas e classificadas

como defeitos ou falso positivos. A decisão final sobre a classificação de uma discrepância sendo discutida é do moderador.

A solução dos defeitos não é discutida durante a reunião, por não fazer parte do objetivo dessa reunião e para evitar que ela se estenda demais. Normalmente, uma reunião de inspeção não deve exceder duas horas, visto que após esse tempo a concentração e a capacidade de análise dos inspetores costumam se reduzir drasticamente. No caso em que uma reunião precisa de mais de duas horas, é sugerido que o trabalho de inspeção continue no próximo dia (KALINOWSKI, 2004).

Retrabalho

Nessa fase o autor corrige os defeitos detectados durante a inspeção.

Continuação

O material corrigido pelos autores é repassado para o moderador, que faz uma análise da inspeção como um todo e re-avalia a qualidade do artefato inspecionado. Ele tem a liberdade de decidir se uma nova inspeção deve ocorrer ou não.

4.4 – Conclusão

Ao longo desse capítulo, foi apresentada Arqcheck, a abordagem proposta para inspeção de documentos arquiteturais baseada em *checklist* (Tabela 4.8).

Tabela 4.8 - Caracterização da abordagem proposta seguindo o *framework* definido no Capítulo 3

***Checklist* para inspeção arquitetural**	
Características da abordagem	
Tipo de arquitetura	Tipo de informação: ▪ Descrição usando diferentes perspectivas de representação (visões arquiteturais) ▪ Rastreabilidade com os requisitos Nível de abstração: ▪ Arquitetura inicial
Tipo de técnica	Questionamento (*checklists*)
Processo seguido	1. Planejamento: realiza-se o planejamento da inspeção assim como a configuração do *checklist* para que ele avalie as características de qualidade definidas 2. Apresentação: apresenta-se o documento a ser avaliado e o *checklist* a ser utilizado 3. Detecção: identificação de discrepâncias através do uso dos questionamentos que compõem o *checklist* 4. Reunião de inspeção: consolidação dos defeitos através da análise das discrepâncias 5. Retrabalho 6. Continuação
Artefatos requeridos	▪ Representação arquitetural ▪ Documento de requisitos ▪ Requisitos de qualidade priorizados
Artefatos produzidos	▪ Lista de defeitos ▪ Recomendações de como corrigir os defeitos através

	da indicação de possíveis táticas arquiteturais que poderiam ser aplicadas na arquitetura
Características de qualidade	Múltiplos
Momento da avaliação	Após construção
Apoio ferramental	-
Avaliação	Estudo experimental (estudos de viabilidade)

Essa abordagem foi criada com o objetivo de identificar defeitos em documentos arquiteturais. Além disso, ela tem como principal requisito minimizar as limitações identificadas nas abordagens de avaliação arquitetural existentes (Tabela 4.9), permitindo assim a atender as características por nós identificadas e que motivaram a realização dessa pesquisa e que facilita a aplicação dessa prática em um contexto industrial.

Uma característica de ArqCheck é que o *checklist* que a compõem foi criado com base no conhecimento normalmente utilizado durante as atividades de especificação arquitetural e também com base em conceitos observados em uma família de técnicas de leitura conhecida como OORT's. Contudo, mesmo usando esses elementos como base para a criação da abordagem, não se pode garantir a sua viabilidade e a sua utilidade. Para isso, estudos experimentais devem ser realizados visando (1) avaliar a viabilidade de utilizar essa abordagem na identificação de defeitos e (2) confirmar a relevância dos itens de avaliação do *checklist* na identificação de defeitos arquiteturais.

Tabela 4.9 - Soluções adotadas para minimizar as limitações identificadas

Limitação das abordagens identificadas	Solução para minimizar as limitações
Subjetividade na avaliação	Uso de um checklist composto por itens de avaliação que indicam ao inspetor o que deve ser avaliado, não deixando a cargo de sua experiência essa decisão.
Custo de aplicação	Uso de inspeção de software por não exigir a realização de tarefas complexas, como a definição de cenários. Além disso, ArqCheck não exige uma grande quantidade de profissionais ou especialistas em arquitetura para que a avaliação seja realizada.
Avaliação de múltiplas características de qualidade	Definição de itens no checklist para avaliar as características de qualidade de acordo com os requisitos de qualidade que foram especificados pelo cliente e que possuem relevância em um contexto arquitetural
Contexto de aplicação limitado	Definição de um processo de configuração que permite adaptar o checklist às características do documento arquitetural a ser avaliado e aos objetivos da avaliação.

No capítulo seguinte, intitulado "Estudos de avaliação da abordagem proposta", descrevemos como esses estudos experimentais foram realizados e os resultados que obtivemos.

Capítulo 5 – Avaliando a abordagem proposta

Neste capítulo, detalhamos como foi feita a avaliação de ArqCheck. Para isso, apresentamos uma breve introdução aos conceitos relacionados à Engenharia de Software Experimental e descrevemos os estudos experimentais executados com o intuito de avaliar a viabilidade da abordagem proposta.

5.1 – Introdução

ArqCheck foi criada a partir de conhecimento e boas práticas utilizadas durante as etapas de especificação arquitetural. Uma das principais tarefas que realizamos para definir essa abordagem consistiu na identificação de como esse corpo de conhecimento utilizado na especificação arquitetural poderia ser utilizado na criação de um *checklist* que permitisse avaliar documentos arquiteturais.

Contudo, quando uma tecnologia é desenvolvida com base em experiências pessoais, observações gerais ou boas prática publicadas, não se pode assegurar que ela obtenha os resultados esperados (BABAR *et al.*, 2004). Portanto, além de desenvolver a tecnologia é também importante avaliar se a tecnologia atende aos objetivos inicialmente definidos. O objetivo da abordagem proposta é permitir a melhoria da qualidade de documentos arquiteturais através da identificação de defeitos.

Para que isso seja possível, diversas perguntas devem ser respondidas até que possamos afirmar que ArqCheck realmente atende aos seus objetivos. Entre os questionamentos possíveis, procuramos responder inicialmente aos seguintes: É viável utilizar a abordagem proposta na identificação de defeitos em documentos arquiteturais? Ela identifica esses defeitos de forma eficiente e eficaz?

Nesse sentido, realizamos uma série de estudos experimentais visando à caracterização da abordagem e sua avaliação. Esses estudos objetivaram principalmente identificar se essa tecnologia realmente obtém os resultados esperados, além de identificar problemas e dificuldades que as pessoas tiveram ao utilizá-la na prática (CIOLKOWSKI *et al.*, 2002). Os objetivos de cada estudo e a ordem em que foram executados foram definidos com base na metodologia experimental descrita por SHULL *et al.* (2001).

Nas seções a seguir, identificamos alguns conceitos básicos de experimentação aplicada à engenharia de software, descrevemos a metodologia de transferência

tecnológica que utilizamos como base e detalhamos os estudos que foram realizados até o momento, assim como os resultados obtidos que permitiram uma evolução da abordagem.

5.2 – Conceitos básicos de Experimentação aplicada à engenharia de software

Experimentação é o centro do processo científico. Novos métodos, técnicas, linguagens e ferramentas não deveriam ser apenas sugeridos, publicados ou apresentados para venda sem experimentação e avaliação (AMARAL, 2003).

Estudos experimentais podem ser usados nesses casos para avaliar essas tecnologias e auxiliar a direcionar pesquisas futuras através da identificação dos problemas e dificuldades que as pessoas possuem ao aplicá-las na prática.

As pesquisas realizadas no contexto de Engenharia de Software geralmente buscam o desenvolvimento de novas técnicas, métodos ou ferramentas de desenvolvimento de software. Contudo, os benefícios e as limitações dessas tecnologias são raramente examinados. TRAVASSOS *et al.* (1999) identificaram pesquisas que mostram que 50% de todos os artigos de engenharia de software realizaram pouca ou nenhuma avaliação das teorias por eles propostas. Ou seja, eles se baseiam principalmente em intuição ou utilizam estudos muito simples para demonstrar a validade de suas teorias.

Para permitir um melhor entendimento e avaliação das tecnologias que estão sendo criadas, estudos experimentais têm se mostrado uma abordagem adequada (TRAVASSOS *et al.*, 1999). Um estudo desse tipo consiste basicamente em uma observação sistemática dos efeitos de uma tecnologia em aplicações práticas (WOHLIN *et al.*, 2000).

Em suma, a utilização de métodos experimentais pode trazer os seguintes benefícios (TICHY, 1998; TRAVASSOS *et al.*, 2001):

- Ajudar a construir uma base confiável de conhecimento e desta forma reduzir a incerteza sobre quais teorias, métodos e ferramentas são adequados;
- Poder levar a novas compreensões sobre áreas atualmente pesquisadas;
- Explorar áreas desconhecidas, permitindo estender a fronteira do conhecimento;
- Acelerar o progresso através da rápida eliminação de abordagens não fundamentadas, suposições errôneas e modismos, ajudando a orientar a engenharia e a teoria para direções promissoras;

- Ajudar no processo de formação da Engenharia de Software como ciência, através do crescimento do número de trabalhos científicos com uma avaliação experimental significativa.
- Na demonstração dos beneficio de uma nova tecnologia e no registro de experiências em relação a sua utilidade;

Segundo WOHLIN (2000), os métodos experimentais podem ser classificados como:

- Survey: investigação executada em retrospecto quando, por exemplo, uma ferramenta ou técnica tem sido utilizada em uma empresa e pretende-se avaliá-la sob algum aspecto (AMARAL, 2003). Algumas formas para coleta de dados como entrevistas e questionários são bastante conhecidas e utilizadas.
- Estudos de caso: usados para monitorar projetos, atividades ou exercícios objetivando rastrear um atributo específico, ou estabelecer relacionamentos entre diferentes atributos, sem um controle muito formal sobre as atividades relacionadas ao método experimental.
- Experimentos: atividade com o propósito de descobrir algo desconhecido ou de testar uma hipótese envolvendo uma investigação de coleta de dados e de execução de uma análise para determinar o significado dos dados (BASILI *et al.*, 1999). Experimentos são apropriados para confirmar teorias, o conhecimento convencional, explorar os relacionamentos, avaliar a predição dos modelos ou validar as medidas. A suas principais características são de (1) permitir o controle total sobre processo e variáveis e (2) de ser possível repeti-lo.

A escolha de qual método de pesquisa experimental utilizar depende dos pré-requisitos da investigação, do propósito, da disponibilidade de recursos e de como se pretende analisar os dados coletados (WOHLIN *et al.*, 2000).

No contexto da nossa pesquisa, estudos experimentais foram realizados para avaliar a abordagem proposta para inspeção de documentos arquiteturais. Para isso, utilizamos principalmente estudos de casos devido (1) aos tipos de avaliação que se deseja realizar (avaliar a viabilidade, eficiência e eficaz da abordagem proposta), (2) às características desse tipo de estudo (fácil de planejar) e (3) ao fato de não termos o controle necessário sobre diversas variáveis que compõem o contexto em que os estudos serão executados (variabilidade na escolha dos participantes, controle sobre os defeitos dos documentos arquiteturais) o que impossibilitaria a realização de experimentos formais, por exemplo.

Além disso, uma das expectativas em relação a ArqCheck é de introduzi-la em um contexto industrial, portanto, optamos por realizar os estudos experimentais de forma que além de avaliar a abordagem proposta eles fornecessem um auxílio na realização

dessa transferência tecnológica. Para isso, utilizamos uma metodologia que auxilia na definição de que estudos devem ser realizados visando minimizar os riscos ao introduzir essa tecnologia em um ambiente industrial. A metodologia utilizada foi definida em (SHULL *et al.*, 2001).

5.3 – Metodologia experimental utilizada

O MCT/SEPIN, órgão federal brasileiro responsável pela política de informática, vem realizando pesquisas anuais em relação à qualidade e à produtividade no setor de software no país (MCT/SEPIN, 2005).

Com base nos resultados obtidos com essa pesquisa, é possível observar um aumento, por parte das empresas brasileiras, pela procura por abordagens que permitam tanto a melhoria de seus processos quanto a de seus produtos. Essas abordagens consistem principalmente na implementação de modelos de maturidades de software, como CMMi (SEI, 2000) e MPS.Br (WEBER *et al.*, 2004), ou então no uso de conceitos como os relacionados a validação e verificação.

Devido a essa procura, durante a construção e avaliação de ArqCheck, procuramos levar em consideração características que facilitem a transferência dessa tecnologia e a sua utilização em um contexto industrial, principalmente o das indústrias de software brasileiras.

Para isso, a estratégia utilizada neste trabalho foi de realizar estudos inicialmente num ambiente acadêmico para então depois realizá-los na indústria. A realização de estudos experimentais em ambientes acadêmicos é um passo necessário por reduzirem o risco de transferência de uma tecnologia imatura.

É importante lembrar que a transferência tecnológica não é uma tarefa trivial. A inserção de uma tecnologia em um ambiente industrial envolve diversos riscos e custos que devem ser minimizados para que ela possa ser aplicada com sucesso.

A transferência de uma tecnologia imatura diretamente em um ambiente industrial apresenta riscos que nem sempre são possíveis de serem mitigados. Com isso, em caso de fracasso não seria possível amadurecer a tecnologia devido à impossibilidade de se identificar quais foram os fatores responsáveis por esse resultado (SHULL *et al.*, 2001), deixando assim de responder a várias dúvidas, como por exemplo: A idéia básica por traz da tecnologia está correta? A tecnologia não é compatível com as características do ambiente industrial? A tecnologia foi aplicada de forma correta?

Além do mais, os custos relacionados ao fracasso de um estudo quando realizado em um contexto industrial são altos, o que justifica a realização de uma série de estudos em um ambiente acadêmico, menos custoso. O principal objetivo dos estudos realizados no contexto é o de identificar e solucionar problemas observados para que

a tecnologia em estudo se torne mais madura e seus resultados não sejam oriundos de variações humanas ou erros experimentais (PORTER *et al.*, 1995).

Com base nesse contexto, adotamos uma metodologia experimental que objetiva desenvolver novas tecnologias e auxiliar a sua transferência para um contexto industrial. Essa metodologia de transferência tecnológica foi definida por SHULL *et al.* (2001) e busca lidar com os principais problemas de uma tecnologia ou processo antes que eles sejam inseridos em um contexto industrial.

As principais motivações que nos influenciaram a utilizar essa metodologia estão no fato (1) dela já ter sido utilizada com sucesso para avaliar outras abordagens de inspeção (TRAVASSOS *et al.*, 1999) e (2) dela permitir mostrar aos profissionais da área de software os possíveis benefícios das tecnologias desenvolvidas na academia.

A aplicação dessa metodologia consiste na avaliação da tecnologia a ser introduzida através de uma série de estudos: de viabilidade, de observação, de caso aplicado em um ciclo de vida real e de caso aplicado em um ambiente industrial (Figura 5.1). Os estudos sugeridos para cada uma das fases estão diretamente relacionados às variáveis que se desejam controlar e os riscos possíveis de serem assumidos na fase em questão.

A seguir, descrevemos o que foi realizado no contexto dessa pesquisa para cada um desses estudos.

5.4 – Estudo de viabilidade

Estudos de viabilidade costumam ser os primeiros a serem conduzidos quando se deseja avaliar uma tecnologia recém criada. Através de um estudo com esse objetivo, procura-se identificar se a tecnologia realmente faz o que ela se propõe a fazer e se é viável continuar a despender recursos para desenvolvê-la. SHULL *et al.* (2001) destacam que revisões nestas questões provocam as maiores alterações na tecnologia, por isso elas devem ser tratadas no início do processo de avaliação.

5.4.1 – Definição do estudo

No contexto da avaliação da abordagem proposta, foi realizado um estudo de viabilidade visando identificar a sua capacidade em produzir resultados práticos no que se refere à detecção de defeitos em um documento arquitetural. Portanto, procurou-se avaliar principalmente se o *checklist* que compõem ArqCheck permite que um inspetor realmente identifique defeitos em um documento arquitetural.

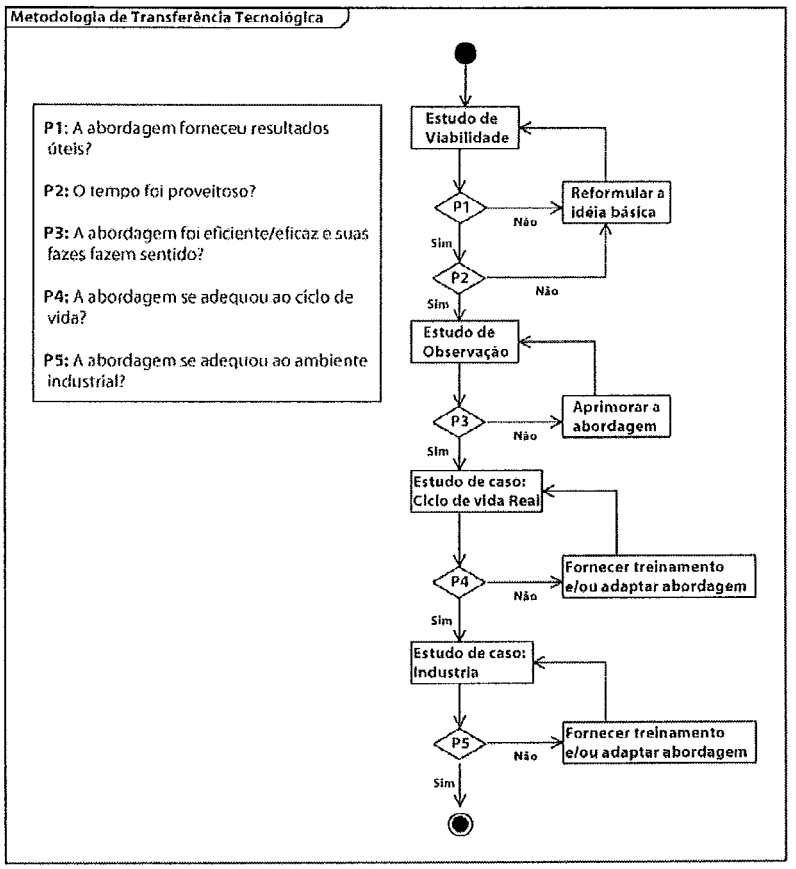


Figura 5.1 - Visão geral da metodologia experimental utilizada. Adaptado de (SHULL *et al.*, 2001)

Uma descrição mais detalhada do objetivo desse estudo está descrito na Tabela 5.1.

Tabela 5.1 - Objetivo do estudo de viabilidade seguindo o método GQM (BASILI *et al.*, 1994)

Analisar o *checklist* para avaliar documentos arquiteturais
Com o propósito de caracterizar
Com respeito à clareza e aplicabilidade em identificar defeitos
Do ponto de vista de inspetores de artefatos de software
No contexto de uma inspeção de um documento arquitetural cujos defeitos são conhecidos

Visando avaliar esse objetivo, buscou-se responder às seguintes perguntas:

- **Q1**: Os itens de avaliação que compõem o *checklist* foram totalmente compreendidos?

  Métricas:

  - Para cada item, quantidade de participantes que não o respondeu da forma planejada;

- **Q2**: Os itens de avaliação que compõem o *checklist* auxiliaram os inspetores na identificação de defeitos?

  Métricas:

  - Para cada item, quantidade de participantes que o utilizou para identificar defeitos;

As respostas obtidas por esses questionamentos foram então utilizadas visando obter algum tipo de conclusão em relação à seguinte hipótese nula:

- **H0**: Não é possível identificar, em um documento arquitetural, defeitos arquiteturais relacionados ao atendimento dos requisitos de software utilizando uma abordagem para inspeção arquitetural baseada em *checklist*.

5.4.2 – Planejamento e Execução do estudo

Para a realização desse estudo, foi utilizado o documento que representa a arquitetura do Ambiente de Experimentação eSEE (NETO *et al.*, 2004), que está sendo construído pelos membros do grupo de pesquisa ESE/COPPE/UFRJ.

Esse documento foi criado de acordo com as recomendações da IEEE 1471 e, portanto atende aos requisitos, descritos no Capítulo 4, que definimos como essenciais para que a abordagem para inspeção proposta possa ser aplicada.

Em relação à corretude desse documento, foi utilizada nessa avaliação uma versão que possui 18 defeitos conhecidos, que foram identificados anteriormente no processo de desenvolvimento através de inspeção *ad-hoc,* pelo grupo que está desenvolvendo o projeto eSEE.

Além disso, visto que o objetivo do estudo é avaliar se os itens que compõem o *checklist* realmente detectam defeitos, foram inseridos defeitos adicionais no documento arquitetural. Essa inserção ocorreu de forma que os defeitos possam ser encontrados pelos itens de questionamento, o que permite avaliar se os itens do *checklist* estão claros e que realmente, ao menos do ponto de vista do pesquisador, auxiliam na identificação de defeitos.

A versão do *checklist* utilizada é composta por 42 itens de avaliação. Entre esses itens, existem os que são aplicáveis e os que não são aplicáveis.

Os itens aplicáveis que se destacam nessa avaliação são os responsáveis por avaliar as características de qualidade designadas pelos *stakeholders*. No contexto desse estudo, o ambiente eSEE possui requisitos de Modificabilidade, Segurança e Usabilidade que devem ter o seu atendimento avaliado. Por isso, é necessário que itens relacionados a esses tipos de requisitos sejam aplicados ao avaliar o documento arquitetural.

Em relação aos itens não aplicáveis, eles são formados principalmente por itens que avaliam características do documento arquitetural que não estão sendo abordadas nesse estudo, como testabilidade, por exemplo, e, portanto não deveriam fazer parte do *checklist*.

Contudo, como outra forma de avaliar a clareza dos itens, optamos por deixar esses itens no *checklist* a ser entregue para os participantes. Com isso, além das opções "Sim" e "Não", o *checklist* utilizado nesse estudo disponibiliza uma terceira opção, "NA - Não Aplicável" (Figura 5.2), que deve ser marcada caso o participante identifique que o item não pode ser usado para avaliar o documento arquitetural que lhe fora passado.

Itens de avaliação da consistência das representações entre os diagramas (específicos à abordagem de documentação arquitetural utilizada)				
Nº	**Visão Modular**	**Sim**	**Não**	**NA**
3	Os módulos internos de cada cluster foram descritos em algum diagrama da visão Modular?			
4	Todo relacionamento definido com um cluster foi devidamente mapeado para um de seus módulos internos?			
Nº	**Visão Dinâmica**	**Sim**	**Não**	**NA**
5	Toda porta/interface possui um nome, é utilizada com um único propósito e de forma única?			
6	Os fluxos de execução, descritos na visão Dinâmica, alocam todos os módulos definidos na visão Modular?			
7	Todo módulo/cluster representado na visão Dinâmica foi descrito na visão Modular?			
8	Todo fluxo entre dois elementos arquiteturais pode ser mapeado para algum relacionamento da visão Modular?			
9	Todo relacionamento, descrito na visão Modular, pode ser mapeado para algum fluxo de comunicação, de dados ou de controle da visão Dinâmica?			
Nº	**Visão de Alocação**	**Sim**	**Não**	**NA**
10	Todo módulo/cluster, representado na visão de Alocação, foi descrito na visão Modular?			
11	Toda dependência, representada na visão de Alocação, pode ser mapeada para um ou mais relacionamentos da visão Modular?			
12	Dado os módulos/clusters representados na visão de Alocação, todos os relacionamentos definidos entre eles na visão Modular também foram representados na visão de Alocação?			

Figura 5.2 - Exemplo de itens do *checklist* utilizado no estudo de viabilidade

Ao fornecer ao participante a possibilidade de marcar um item como não aplicável, não desejamos avaliar se determinado grupo de itens realmente é aplicável ou não. O pesquisador já conhece quais são esses itens pois eles dependem das características de qualidade que se deseja avaliar. Com essa atitude, o que buscamos é identificar se os inspetores conseguem entender o propósito do item e identificar se o item é aplicável ou não aplicável. Partimos do princípio que se o participante marcar um item aplicável como não aplicável ou vice-versa, isso poderia ser um indício de que ele não entendeu o propósito do item, o que categoriza o item como não claro.

Após as alterações no *checklist*, o pesquisador aplicou ArqCheck no documento arquitetural a ser distribuído aos participantes. Essa tarefa foi realizada com o objetivo de caracterizar o *checklist* em relação aos defeitos do documento arquitetural avaliado. Essa caracterização permitiu, a partir do ponto de vista do pesquisador, (1) definir que itens do *checklist* levariam a identificar os defeitos, conhecidos como "identificadores de defeitos", (2) definir se todos os defeitos conhecidos até o momento seriam identificados pelos itens, (4) avaliar se haveria novos defeitos que a inspeção *ad-hoc* não teria identificado e (5) identificar os itens que não seriam aplicáveis.

Com base nos resultados obtidos através dessa caracterização (Tabela 5.2), foi possível identificar, durante a análise dos dados obtidos através do estudo, se os itens identificados pelo pesquisador realmente auxiliam os participantes a detectar defeitos.

Sabemos da limitação dessa análise, principalmente devido ao viés que pode ocorrer devido ao fato de ser o pesquisador quem realizou essa caracterização do *checklist*. Contudo, o objetivo dessa comparação é obter algum indício sobre a clareza dos itens, para que sejam melhorados, e os resultados obtidos por essa caracterização permitem a identificação desses indícios.

Para isso, partimos do princípio de que se um participante não identificar um defeito a partir de um item marcado como tal, isso pode significar que o item não está claro o suficiente e que deve sofrer melhorias.

Tabela 5.2 - Caracterização dos itens do *checklist*

Item	Identifica defeitos?	Item	Identifica defeitos?	Item	Identifica defeitos?
1	Não	15	Não	29	Não
2	Sim	16	Sim	30	Sim
3	Sim	17	Não	31	Não
4	Sim	18	Não	32	Não
5	Não	19	Sim	33	Não
6	Não	20	Não	34	Não
7	Sim	21	NA	35	Não
8	Sim	22	NA	36	Não
9	Não	23	NA	37	Sim
10	Não	24	NA	38	Sim
11	Não	25	NA	39	NA
12	Sim	26	NA	40	NA
13	Não	27	NA	41	Não
14	Sim	28	NA	42	Não

Após esse planejamento, o estudo foi executado em outubro de 2005 com 4 participantes, escolhidos por conveniência. Estes participantes são alunos de pós-graduação e possuem conhecimento no domínio do problema e na aplicação de técnicas de inspeção.

O fato dos participantes possuírem esse perfil de conhecimento não oferece risco em relação à validade dos resultados que podem ser obtidos com a condução desse estudo. Na verdade, é soba perspectiva destes participantes que se pretende avaliar ArqCheck de acordo com a descrição GQM realizada. Isso se justifica visto que o objeto de estudo é a abordagem ArqCheck, além do mais, pelos participantes possuírem conhecimento em técnicas de inspeção, por exemplo, é possível que o retorno fornecido por eles seja de extrema relevância para a melhoria da abordagem avaliada.

Devido à falta de uma abordagem padrão para representação arquitetural, foi realizado, antes da execução do estudo, um treinamento com os participantes sobre a abordagem de representação que foi utilizada para descrever o documento arquitetural a ser avaliado. O principal objetivo na realização desse treinamento é de evitar problemas de entendimento dos participantes em relação à simbologia utilizada, o que poderia influenciar na tarefa de identificação de defeitos.

Após isso, os participantes assinaram um formulário de consentimento de participação do experimento e preencheram um questionário de caracterização. Para esse estudo, os dados relacionados à experiência dos participantes não foram utilizados para agrupamento por termos poucos participantes, o que não possibilitava a divisão em diferentes grupos, por exemplo.

Feito isso, os artefatos necessários para que a inspeção fosse realizada foram distribuídos. Por não estar no objetivo do estudo observar os participantes durante a execução da inspeção, a inspeção foi conduzida em apenas uma sessão de forma remota, ou seja, os participantes não realizaram a inspeção ao mesmo tempo e nem no mesmo lugar. Contudo, eles foram informados que comentários entre eles deveriam ser evitados para não prejudicar a validade do estudo.

Para cada participante, foi distribuído o *checklist* de avaliação, o documento arquitetural, o documento de requisitos, o relatório de discrepâncias e o questionário de pós-experimento, utilizado para obter informações de ordem qualitativa. Como retorno, eles deveriam retornar ao pesquisador o relatório de discrepâncias preenchido, indicando o tempo que levaram para realizar a inspeção, o *checklist* com os itens selecionados e o questionário de pós-experimento preenchido.

Nesse estudo não foi necessário realizar a etapa de Reunião de Inspeção, onde as discrepâncias identificadas pelos inspetores são discutidas com o autor do documento visando identificar quais dela são defeitos ou falso-positivos. A principal razão por não ter sido necessária a realização dessa reunião foi o fato dos pesquisadores conhecerem de antemão os defeitos contidos no documento.

5.4.3 – Análise dos dados e resultados obtidos

Durante o planejamento desse estudo foi definido como objetivo principal avaliar a viabilidade de ArqCheck. Essa viabilidade, conforme os questionamentos identificados, seria avaliada através da análise da compreensão dos itens de avaliação do *checklist* pelos participantes e através do auxílio que cada item ofereceu a identificação dos defeitos. A seguir, descrevemos que resultados preliminares foram obtidos para cada um desses questionamentos.

Os itens de avaliação que compõem o checklist foram totalmente compreendidos?

A avaliação da compreensão dos itens que compõem o *checklist* consistiu em comparar o valor atribuído pelos participantes a cada item de avaliação (S, N, NA) em relação aos valores identificados durante a caracterização do *checklist* (Tabela 5.2), que fora realizada pelo pesquisador. Nesse momento, não foi levado em consideração o fato do item ter auxiliado os participantes na identificação de defeitos. Para definir se um item foi compreendido ou não, partimos do princípio que se o participante não atribuísse o valor esperado ao item de avaliação, isso seria um possível indício de que o item pode não ter sido compreendido.

Sendo assim, se um participante tiver identificado um item aplicável como não aplicável ou vice-versa, ou então se o item identificar um defeito, mas ele não tiver marcado de tal forma, é considerado que o participante não entendeu corretamente o propósito do item.

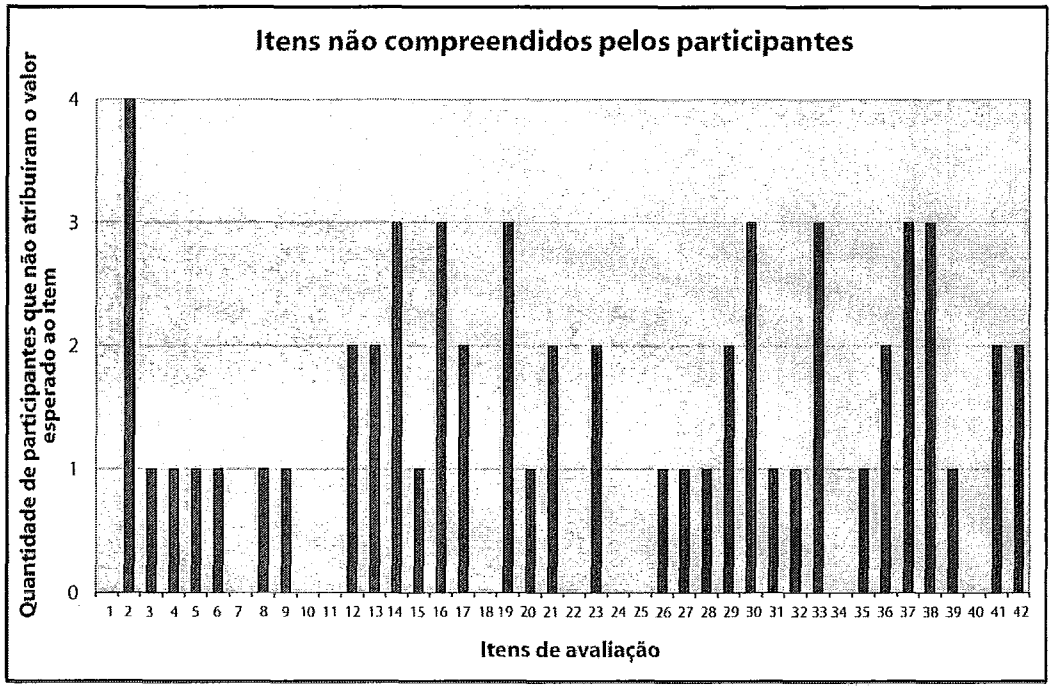


Gráfico 5.1 - Compreensão dos itens de avaliação

Portanto, com base nos dados coletados, foi identificado para cada item a quantidade de participantes que atribuíram erroneamente valores para ele. Com isso, foi observado que 8 itens de avaliação não foram compreendidos por pelo menos 3 participantes e por conseqüência precisam ser analisados e possivelmente sofrer melhorias (Gráfico 5.1). Esses itens foram: 2, 14, 16, 19, 30, 33, 37, 38 (vide Apêndice B.1).

Com base nesses dados, foi possível observar também uma outra característica da abordagem proposta: a necessidade das atividades de configuração. Ao analisar os dados em relação à compreensão, foi identificado que 80% dos itens não-aplicáveis foram marcados como tal (Gráfico 5.2), justificando assim a realização das atividades de configuração.

Além disso, pode ser observado que alguns participantes indicaram alguns itens aplicáveis como não-aplicáveis. Esses itens, dependendo da quantidade de participantes que o marcaram, serão analisados, pois podem apresentar problemas como, por exemplo, falta de clareza ou não são realmente aplicáveis a uma avaliação arquitetural.

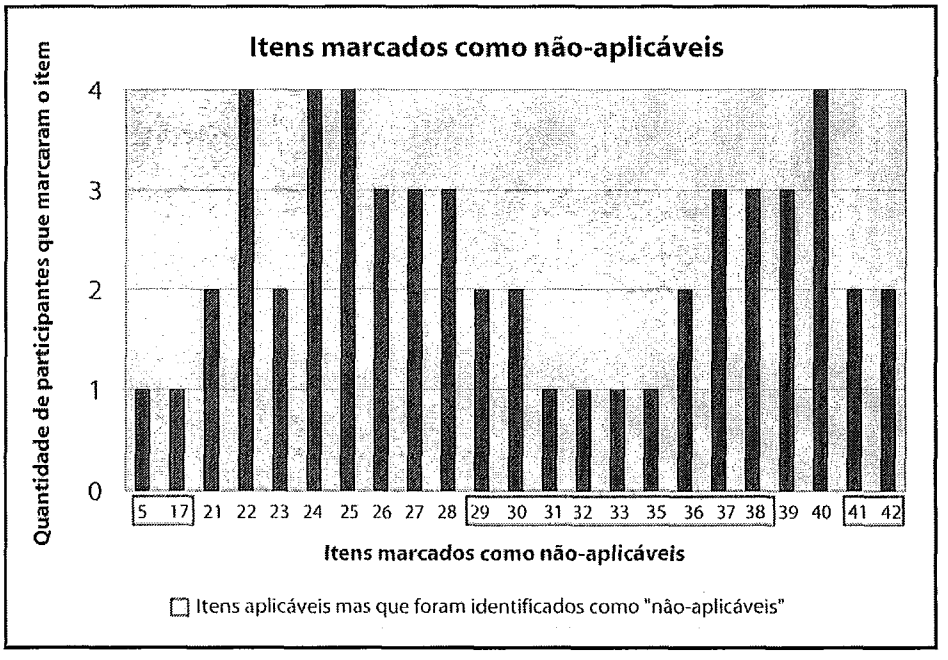


Gráfico 5.2 - Itens não-aplicáveis

Os itens de avaliação que compõem o checklist auxiliaram os inspetores na identificação de defeitos?

Para avaliar se os itens realmente auxiliaram o inspetor na identificação de defeitos, além das perguntas realizadas através do questionário de pós-experimento, foi realizada uma análise dos relatórios de discrepância de cada participante. Essa análise identificou que itens levaram cada participante a detectar defeitos e permitiu a

comparação desses dados com os obtidos pela caracterização realizada pelo pesquisador.

Sabe-se das limitações em obter conclusões a partir desse tipo de comparação, uma vez que não se pode afirmar que determinado defeito só pode ser detectado pelo item indicado pelo pesquisador. Contudo, essa análise fornece indícios de itens de avaliação que deveriam ser revistos, principalmente em relação a sua clareza.

Além do mais, visto que não foi possível realizar agrupamentos dos participantes em relação aos seus níveis de experiência, devido à pequena quantidade de participantes, a análise dos dados em relação a essa perspectiva apresenta fatores de confusão[8]. Esses fatores ocorrem visto que não é possível determinar se os defeitos são identificados graças ao conhecimento contido no item de avaliação ou ao conhecimento do inspetor obtido através de sua experiência. Contudo, mesmo com a presença desse fator, os indícios fornecidos em relação aos itens que auxiliaram na identificação de defeitos são relevantes para a evolução do *checklist*.

Portanto, após essa análise (Gráfico 5.3), foi possível observar que alguns dos itens que o pesquisador caracterizou como "identificador de defeitos" não auxiliaram os participantes. Os principais itens onde isso ocorreu foram nos identificados com o número 2, 12, 14, 19, 30, 37 e 38. Acreditamos que o principal motivo seja a falta de clareza desses itens.

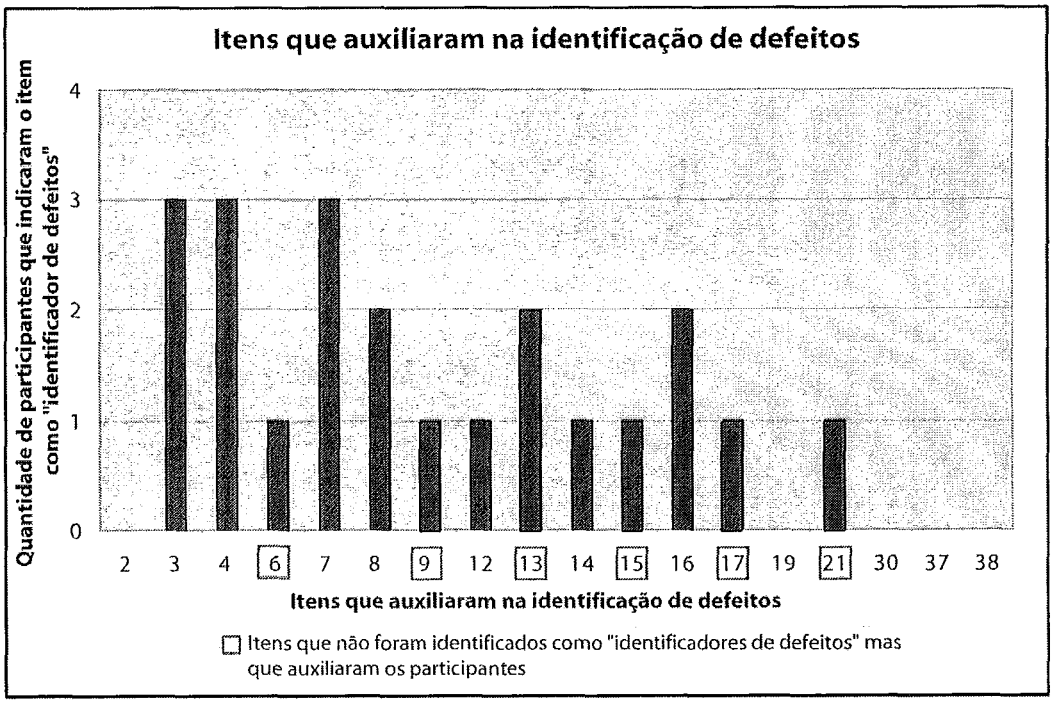


Gráfico 5.3 - Itens que auxiliaram na identificação de defeitos

[8] *Confounding factors*

Um outro fato que foi observado ao analisar os resultados obtidos em relação à detecção de defeitos foi o elevado tempo de execução, em média 3 horas para a realização da inspeção. Mesmo sendo um documento arquitetural que representava uma arquitetura com certa complexidade, os participantes já possuíam conhecimento em relação ao domínio do problema. Sendo assim, acreditamos que o elevado tempo de inspeção foi ocasionado por itens redundantes, que faziam com que os participantes re-avaliassem o documento buscando defeitos que já haviam sido identificados, e também pela presença de itens não-aplicáveis, que faziam com que os participantes gastassem uma determinada quantidade de tempo para identificar se o item era realmente não-aplicável.

Os itens redundantes foram classificados como tal por terem sido originalmente definidos com o propósito de detectar determinado tipo de defeito, mas que na prática identificaram os mesmos tipos de defeitos que outros itens. Os principais grupos de itens redundantes identificado foram: os itens 2, 13 e 14 e os itens 3, 4, e 9. Esses itens foram analisados, e com base nisso foram reescritos ou excluídos do *checklist*.

5.4.4 – Considerações em relação à hipótese nula

Ao analisar os relatórios de discrepâncias devolvidos pelos participantes do estudo, observou-se que, em média, cada participante identificou 30% dos defeitos (Gráfico 5.4).

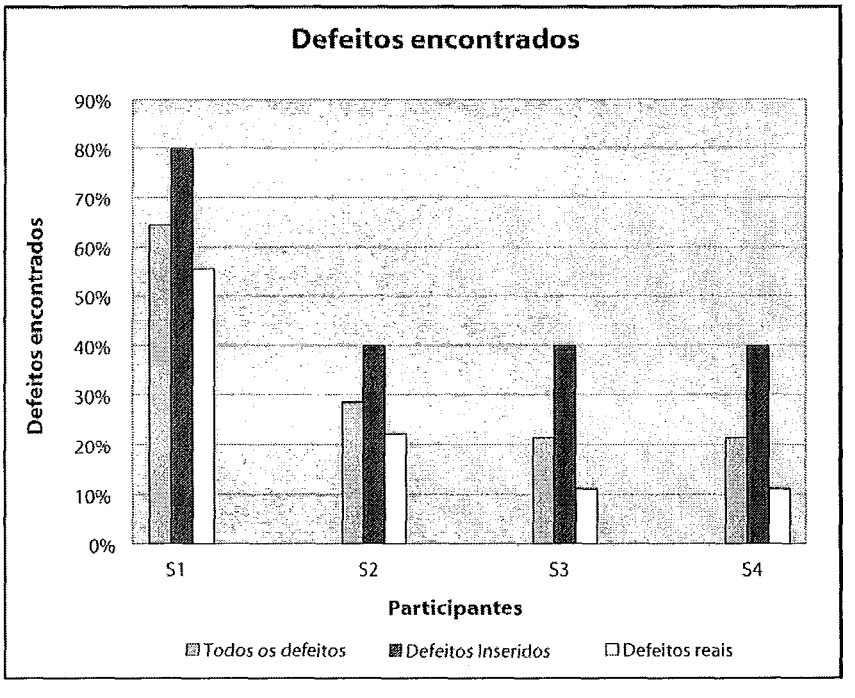


Gráfico 5.4 - Defeitos encontrados

Ao analisar as porcentagens individuais dos defeitos inseridos e defeitos reais, percebemos que a porcentagem de defeitos inseridos encontrados por participante é

bem maior que a de reais. Uma das razões dessa diferença de valores está no fato dos defeitos inseridos terem sido enxertados no documento com o objetivo de avaliar a clareza dos itens, ou seja, os defeitos foram colocados de forma que haveria uma grande possibilidade de eles serem facilmente identificados através dos itens.

Mesmo não sendo o objetivo desse estudo avaliar a quantidade de defeitos que o *checklist* auxilia identificar, ao consolidar os relatórios de discrepâncias, identificou-se que 75% dos defeitos conhecidos foram encontrados pela abordagem. A diferença entre a porcentagem de defeitos encontrados individualmente e a de defeitos encontrados no total indica que cada participante encontrou poucos defeitos no geral,, mas cada um encontrou defeitos diferentes entre si, o que aumentou a cobertura dos defeitos encontrados. Esses resultados nos fornecem dois indícios: (1) de que a experiência dos inspetores em arquitetura de software e no domínio da solução influencia nos defeitos detectados, a dependência da experiência do inspetor é uma das principais característica de uma abordagem de inspeção baseada em *checklist* (SHULL *et al.*, 2000), e (2) dos ganhos que podem ser obtidos ao se alocar mais de um inspetor para esse tipo de avaliação.

Sendo assim, a realização desse estudo não permite refutar totalmente a hipótese de que ArqCheck não permite identificar defeitos arquiteturais, uma vez que não é possível obter resultados relevantes a nível estatístico, principalmente devido à pequena quantidade de participantes. Contudo, esses resultados preliminares fornecem alguns indícios que nos levam a crer nessa possibilidade, principalmente, se for levado em consideração que inspeções de software encontram, em média, 60% dos defeitos contidos no artefato avaliado (BOEHM, 1981) e que nesse estudo 75% dos defeitos conhecidos foram encontrados.

5.4.5 – Lições aprendidas

Com base na análise das discrepâncias identificadas e das respostas do questionário de pós-experimento, foi possível identificar:

- Os itens que necessitavam serem reescritos, por não auxiliarem na identificação de defeitos ou não estarem suficientemente compreensíveis;
- Os itens que poderiam ser excluídos do *checklist* por estar avaliando uma mesma característica várias vezes.

A partir do questionário de pós-experimento, os participantes opinaram em relação à relevância do *checklist* na identificação de defeitos arquiteturais. Como resultado, todos concordaram com o auxílio que esse *checklist* oferece para essa tarefa.

Além do mais, com base nessas informações foi identificada a necessidade de:

- Treinamento na utilização do *checklist*, principalmente relacionado ao uso das "Guias de identificação de contexto", utilizadas durante a avaliação do atendimento aos requisitos de qualidade;
- Descrição mais detalhada da taxonomia de defeitos utilizada, fornecendo assim maior auxílio ao inspetor durante a classificação do tipo de defeitos identificado;
- Definição de um glossário visando auxiliar na compreensão de conceitos utilizados pelos itens de avaliação;
- Um mecanismo que permita informar ao inspetor que a atribuição de determinado valor para um item implica na presença de um defeito no documento avaliado. Para isso, foi adicionado à nova versão do *checklist* o campo "RE- Resultado esperado";
- Alocar mais de um inspetor, quando possível, para participar desse tipo de avaliação.

Após a evolução da abordagem com base nos resultados obtidos, o passo seguinte deveria ser a re-execução desse estudo, utilizando o mesmo documento arquitetural, mas utilizando participantes diferentes. O principal objetivo dele seria de avaliar se as modificações permitiram a identificação de uma quantidade maior de defeitos. Contudo, por esse estudo fazer parte de uma pesquisa de mestrado, que deve ser finalizada em um determinado espaço de tempo, e ter sido executado em um contexto acadêmico, onde existe escassez de participantes, seria necessário esperar o início do próximo semestre para possibilitar essa re-execução, o que seria inviável devido ao prazo de defesa de mestrado. Por isso, assumimos os riscos e realizamos o segundo estudo definido pela metodologia: o estudo de observação.

5.5 – Estudo de observação

Um estudo de observação ocorre em um ambiente onde indivíduos desempenham tarefas enquanto são observados pelos pesquisadores. O propósito deste estudo é coletar dados sobre como essas tarefas são executadas. Desta forma, os pesquisadores podem adquirir uma compreensão mais refinada sobre como a nova tecnologia é aplicada na prática, presenciando eventuais dificuldades que os indivíduos podem enfrentar (SILVA, 2004).

A coleta de dados num estudo de observação pode se dar de duas formas: observação e investigação (SHULL *et al.*, 2001). Dados de observação são coletados enquanto a abordagem está sendo executada, sem a interferência do pesquisador. Dados de investigação, por sua vez, são coletados ao término de determinada etapa

da abordagem, quando os indivíduos respondem a perguntas pré-definidas, abordando questões sobre as quais eles, normalmente, não pensariam a respeito.

No contexto dessa pesquisa, tratamos esse estudo de observação como um pré-experimento, pois um único objeto foi avaliado através de uma comparação com uma *baseline,* também conhecido como elemento de controle (AMARAL, 2003). Nesse estudo, ArqCheck foi aplicada para inspecionar um documento arquitetural que já havia sido previamente inspecionado. Em posse dos defeitos detectados por ArqCheck e os detectados pela abordagem previamente aplicada, desejamos realizar uma comparação desses resultados, visando avaliar a abordagem proposta.

5.5.1 – Contexto

A principal característica desse estudo de observação, foi que a avaliação de ArqCheck utilizou um documento arquitetural real, que fora produzido em um projeto de desenvolvimento de software de uma empresa.

Para permitir essa troca de informações academia-indústria, um acordo foi firmado entre o grupo de pesquisa ESE/COPPE/UFRJ e a SIEMENS-AM.

A empresa SIEMENS-AM[9] é um instituto de pesquisa e de desenvolvimento de software da SIEMENS Mobile. Esse instituto possui uma indicação de maturidade em desenvolvimento de software (CMMi nível 2 e aplica práticas do CMMi nível 3) e por isso possui consciência na importância em validar e verificar os artefatos que produzem durante o processo de desenvolvimento de software. Para a avaliação de seus documentos de projeto, por exemplo, eles aplicam uma abordagem ad-hoc de avaliação, que chamaremos no contexto dessa dissertação de abordagem SIEMENS.

Nesse instituto são desenvolvidas aplicações voltadas para a área de telefonia celular que devem lidar com todas as limitações presentes nessa plataforma de desenvolvimento.

O acordo firmado entre essas duas instituições, com cláusulas de confidencialidade, previa a troca de informações visando à realização desse estudo, o que possibilitou o acesso dos pesquisadores da COPPE aos documentos de requisitos e de projeto de alguns dos projetos da SIEMENS-AM.

O projeto escolhido para ter seus documentos avaliados foi o projeto "Application Menu X85". Esse projeto foi escolhido por seus documentos de projeto já terem sido avaliados por abordagens próprias da SIEMENS-AM, o que possibilitava usar os dados dessa avaliação como controle, e foi escolhido também por possuir

---

[9] SIEMENS-AM: Empresa filial da SIEMENS Mobile que atua em Manaus/AM no desenvolvimento de software voltado para aparelhos celulares. Durante a realização desse estudo, essa filial foi adquirida pela BENQ, mas continuaram no mesmo ramo de atuação.

características de qualidade que obrigaram os arquitetos a utilizarem diversas práticas arquiteturais para projetá-las.

O propósito desse projeto é desenvolver um software que atue como gerenciador do menu de serviços de um aparelho celular. Esse gerenciador deve disponibilizar itens de menu, que quando selecionados executam a aplicação correspondente, e deve ser altamente configurável, para permitir as diferentes customizações que possam ser requisitadas pelas operadoras de telefonia, o principal cliente desse software.

Durante a realização desse estudo, esse projeto já estava em fase de conclusão, com isso o documento de projeto a ser avaliado por ArqCheck já tinha sido avaliado através da abordagem SIEMENS.

Em relação aos defeitos presentes no documento de projeto, fomos informados somente que o documento que nos foi passado apresenta defeitos. Contudo, para minimizar possíveis vieses, não foi informado quais são esses defeitos ou de que tipos são. Esses dados foram repassados somente durante a etapa de análise dos dados do estudo.

Uma outra característica especial que compõem o contexto onde esse estudo foi executado é o fato de haver restrições, principalmente de confidencialidade impostas pelo acordo firmado com a SIEMENS-AM, que implicaram em algumas restrições, como por exemplo, a limitação do número de participantes.

Devido a essa limitação, foi possível utilizar participantes somente na fase de detecção de defeitos. Para as demais fases, principalmente as que envolviam a atuação do moderador, foi o próprio pesquisador que as executou.

5.5.2 – Definição do estudo

Com base nos resultados do estudo previamente realizado, foram identificados indícios que nos levam a crer na viabilidade de ArqCheck em detectar defeitos em um documento arquitetural.

Visando amadurecer a abordagem e buscar obter mais indícios que confirmem o que já foi observado até o momento, realizamos um estudo de observação. Nesse estudo, os resultados obtidos pela aplicação de ArqCheck foram comparados com os defeitos detectados pela abordagem SIEMENS, uma abordagem utilizada no contexto industrial em que a arquitetura avaliada pertence.

Entretanto, a comparação entre as duas abordagens não era o nosso objetivo principal, ou seja, não visamos identificar qual a melhor abordagem de avaliação. A determinação da melhor abordagem não é possível devido ao fato dos resultados obtidos com a aplicação dessas abordagens em um único estudo de caso não serem

suficientes para tirarmos esse tipo de conclusão, e também devido ao fato de não termos informações suficientes para caracterizar, em um primeiro momento, a abordagem SIEMENS.

Portanto, o objetivo dessa comparação é caracterizar ArqCheck em relação a um *baseline*. Para esse estudo, o *baseline* escolhido foi a abordagem SIEMENS por ser a abordagem que de fato está sendo utilizada nesse contexto industrial e por atualmente ser a abordagem da SIEMENS-AM para a melhoria da qualidade de documentos de projetos.

Portanto, com esse estudo de observação buscamos criar um corpo de conhecimento que nos permita avaliar a aplicação da abordagem proposta. Além disso, esperamos que os resultados obtidos, e o corpo de conhecimento construído decorrente de sua condução, nos forneçam subsídios que permitam evoluir a abordagem para otimizar a sua aplicação tanto em relação ao esforço necessário para executá-la quanto à quantidade e tipo de defeitos que ela permite identificar.

Para realizar a comparação entre essas duas abordagens, analisamos principalmente a eficiência e eficácia das duas abordagens. A Tabela 5.3 descreve os objetivos desse estudo de forma mais completa.

Tabela 5.3 - Objetivos do estudo de observação seguindo o método GQM (BASILI *et al.*, 1994)

Analisar a abordagem proposta para inspeção de documentos arquiteturais (ArqCheck)
Com o propósito de caracterizar
Com respeito à eficácia e custo/eficiência em identificar defeitos
Do ponto de vista dos pesquisadores
No contexto de uma inspeção de documento arquitetural que fora criado em um contexto industrial por inspetores com experiência em inspeção

Por eficácia, entendemos a quantidade de defeitos em relação ao número total de discrepâncias encontradas. Por eficiência, entendemos a quantidade de defeitos identificados por unidade de tempo de inspeção.

Com base nesses objetivos, definimos os seguintes questionamentos que procuramos responder durante a execução do estudo:

- **Q1**: Qual é a eficácia de ArqCheck no que diz respeito à detecção de defeitos?
  Métrica:
  - Eficácia da abordagem = Quantidade de defeitos detectados por ArqCheck / Quantidade de discrepâncias identificadas.

- Eficácia 2 da abordagem = Quantidade de defeitos detectados por ArqCheck e comuns a abordagem SIEMENS/ Quantidade de discrepâncias identificadas pela abordagem SIEMENS.

- **Q2**: Qual é o custo/eficiência de ArqCheck no que diz respeito à detecção de defeitos?

  Métricas:

  - Custo/Eficiência da abordagem (ArqCheck) = Quantidade de defeitos detectados por participante / Tempo de inspeção do participante
  - Custo/Eficiência da abordagem (SIEMENS) = Quantidade de defeitos detectados / Tempo total de inspeção

Com base nas respostas obtidas por esses questionamentos, procuramos refutar através da execução desse estudo a seguinte hipótese nula:

- **H0**: Não existe diferença entre os resultados encontrados pela abordagem ArqCheck e a abordagem SIEMENS em relação à detecção de defeitos arquiteturais.

Informações adicionais relacionadas à definição do estudo, ao seu planejamento e à sua execução podem ser encontradas no Plano do Experimento, descrito no Apêndice D dessa dissertação.

5.5.3 – Planejamento e Execução

A realização desse estudo permitiu a observação da aplicação da abordagem proposta em um documento construído em um contexto real. Visto que ele já havia sido inspecionado, foi possível também comparar os resultados obtidos pelas duas abordagens e caracterizar ArqCheck em relação aos resultados obtidos pela abordagem SIEMENS.

Contudo, durante o planejamento do estudo, algumas tarefas de adaptação tiveram que ser realizadas. Essas tarefas consistiram principalmente na adaptação do documento de projeto fornecido pela SIEMENS-AM às características que devem estar presentes em um documento arquitetural para que ele possa ser avaliado pela abordagem ArqCheck. Além disso, visto que a arquitetura descrita nesse documento foi representada através da notação UML, itens de avaliação tiveram que ser criados para compor o *checklist* e permitir a avaliação desse tipo de representação.

Adaptações necessárias

O documento de projeto fornecido pela SIEMENS-AM[10] é composto tanto por informações referentes ao projeto de alto nível, representando principalmente informações a nível arquitetural, quanto por informações de projeto de baixo nível, representados principalmente por diagramas orientados a objetos.

Portanto, devido à presença de informações que não são relevantes a nível arquitetural e que poderiam atrapalhar os inspetores durante a detecção de defeitos, o pesquisador teve que realizar uma análise desse documento visando identificar as informações necessárias para aplicar a abordagem de inspeção.

Essas informações, depois de identificadas, foram extraídas desse documento e utilizadas na criação de um documento arquitetural[11] que foi utilizada na etapa de detecção de defeitos desse estudo.

A escolha de que informações extrair do documento de projeto original foi baseada nas características que um documento arquitetural deve conter e que são essenciais para permitir a aplicação da abordagem ArqCheck. Sendo assim, procuramos principalmente por:

- Informações que permitissem representar o software através de diferentes perspectivas (diagramas de subsistemas, diagramas de classe de alto-nível e diagramas de seqüência de alto nível);
- Informações que justificassem a construção dos elementos arquiteturais e sua organização, ou seja, uma rastreabilidade entre os elementos arquiteturais e os requisitos do software (Tabela de rastreabilidade com os requisitos);
- Descrição do papel de cada elemento dentro da arquitetura (Descrição textual dos elementos arquiteturais);

Vale lembrar que esse esforço foi necessário devido à falta de consenso presente na comunidade de Arquitetura de Software em relação ao que é arquitetura e que tipo de informação é relevante para representá-la. Além do mais, essas informações estavam presentes no documento disponibilizado pelos arquitetos da SIEMENS-AM, foi necessário somente extrair as informações relevantes a nível arquitetural e criar o documento arquitetural do projeto.

Portanto, com base nas informações extraídas foi possível descrever a arquitetura desse software através de três visões arquiteturais: a visão de contexto geral, a modular e a dinâmica. A definição da visão de alocação não foi necessária pois todos

---

[10] Visto que esse documento representa informações reais relacionadas a um dos projetos da SIEMENS-AM, ele não pode ser apresentado aqui por motivos de confidencialidade.
[11] O documento arquitetural do projeto AppMenu X85 contém 31 páginas.

os elementos arquiteturais estavam alocados em um mesmo nó computacional. A seguir descrevemos o que continha cada uma dessas informações:

- **Visão de contexto geral**: Composta por informações que descrevem a decomposição da aplicação em subsistemas e como os sistemas externos se relacionam com eles;
- **Visão Modular**: Composta por informações que descrevem os elementos arquiteturais que compõem a solução e como eles se relacionam
- **Visão Dinâmica**: Composta por informações que descrevem o comportamento dos elementos arquiteturais durante a execução do sistema. Para a criação dessa visão foram utilizadas principalmente informações oriundas dos diagramas de caso de uso e de seqüência descritos no documento de projeto.

A segunda atividade de adaptação que foi realizada foi a criação de itens adicionais do *checklist* para avaliar as características específicas da representação arquitetural utilizada pela SIEMENS-AM (notação UML).

Essa atividade não consiste na configuração do *checklist*, onde os itens existentes são escolhidos de acordo com as características da inspeção e que é uma das atividades executadas durante o processo de inspeção. Ela consiste na criação de itens para avaliar especificamente documentos arquiteturais que são representados de através da notação UML. Tivemos que realizar essa atividade pois não tínhamos aplicado ArqCheck para avaliar arquiteturas que utilizassem esse tipo de representação.

Os itens que tiveram que ser criados foram principalmente itens utilizados para avaliar a consistência do documento, uma vez que esses tipos de itens, conforme descrito no Capítulo 4, são específicos à abordagem de representação utilizada.

O processo de criação foi divido em duas atividades. Em um primeiro momento procuramos adaptar os itens existentes aos conceitos presentes na notação UML. Após isso, analisamos a notação UML e procuramos identificar as regras de consistência semânticas existentes entre os diferentes diagramas que a compõem.

Essa análise foi baseada principalmente nas heurísticas de rastreabilidade presentes nas técnicas para leitura horizontal da família de técnicas de leitura OORT´s (TRAVASSOS *et al.*, 2002), pois um de seus objetivos é avaliar a consistência entre diagramas UML de um projeto.

Como resultado desse processo de adaptação do *checklist* 12 novos itens de avaliação foram criados. Contudo, eles são utilizados somente quando se deseja avaliar um documento representado através da notação UML. Essa escolha é feita durante o planejamento da inspeção, através das atividades de configuração do *checklist*.

Execução do estudo

A proposta desse estudo é de realizar a execução de todo processo de inspeção que compõem a abordagem proposta. Contudo, devido às limitações impostas pelo contexto em que esse estudo foi realizado, nem todas as atividades puderam ser realizadas, principalmente as que exigiam a presença dos autores do documento, pela impossibilidade de utilizar profissionais da SIEMENS-AM durante sua execução.

Sendo assim, as atividades que puderam ter certo controle nas suas execuções, permitindo a realização de algum tipo de observação e investigação foram as atividades de planejamento e detecção de defeitos. A atividade de planejamento foi realizada pelo próprio pesquisador. Somente na atividade de detecção de defeitos que os participantes foram acionados.

Esse estudo de observação foi conduzido em Fevereiro de 2006, seguindo o processo de inspeção definido por ArqCheck, onde a primeira atividade a ser realizada é a de planejamento da inspeção. Durante esse planejamento, os inspetores foram escolhidos e o *checklist* foi configurado para atender às características do documento arquitetural e atender aos objetivos da inspeção.

Visto que o estudo foi realizado em um ambiente acadêmico, os inspetores escolhidos foram estudantes de pós-graduação em Engenharia de Software. Contudo, devido às restrições impostas pelo acordo de confidencialidade estabelecido com a SIEMENS-AM, o número de inspetores se limitou a dois indivíduos.

Esses inspetores já haviam aplicado ArqCheck, durante a realização do estudo de viabilidade, portanto eles já possuíam conhecimento da abordagem de inspeção utilizada. Além do mais, conforme pode ser visto na Tabela 5.4, eles possuem conhecimento em inspeção de software, em arquitetura de software, mas nunca atuaram no domínio de desenvolvimento de software para aparelhos móveis.

Em relação a detecção de defeitos, além dos participantes, o pesquisador também aplicou ArqCheck no documento visando obter dados de observação para serem utilizados durante a análise dos resultados. Contudo, os defeitos por ele identificados foram isolados e não serão utilizados durante a análise dos dados do estudo para evitar possíveis vieses nos resultados da avaliação.

Em relação à configuração do *checklist*, visto que o documento arquitetural foi descrito através de três visões arquiteturais (contexto geral, modular e dinâmica) e que os diagramas que descrevem essas visões foram representadas através da UML, somente os itens referentes a essas visões e a essa abordagem de descrição foram selecionados.

Tabela 5.4 - Conhecimento e habilidades dos inspetores

Experiências	Inspetor S1	Inspetor S2	Pesquisador
Com desenvolvimento de software	Trabalhou 3 anos como programador em uma empresa que desenvolvia software para web e vem desenvolvendo software a 5 anos no meio acadêmico.	Trabalhou por 13 anos como programador, analista de sistemas e coordenador de projetos na indústria	Trabalhou dois anos com desenvolvimento de software para desktop
Em projeto arquitetural	Praticou em 1 projeto em sala de aula	Usou em vários projetos na indústria	Usou em vários projetos na indústria
Em leitura de documento arquitetural	Praticou em 1 projeto em sala de aula	Praticou em 1 projeto em sala de aula	Praticou em 1 projeto em sala de aula
Em desenvolver projetos a partir de requisitos e casos de uso	Praticou em 1 projeto em sala de aula	Usou em vários projetos na indústria	Usou em vários projetos na indústria
Em projetos Orientados a Objetos	Praticou em projetos em sala de aula	Usou em vários projetos na indústria	Usou em vários projetos na indústria
Com UML	Usou em 1 projeto na indústria	Usou em vários projetos na indústria	Usou em vários projetos na indústria
Com inspeções de software	Praticou em 1 projeto em sala de aula	Praticou em 1 projeto em sala de aula	Praticou em 1 projeto em sala de aula
Em desenvolvimento de software para celular	Nenhum	Nenhum	Estudou em aula ou em livro

Em relação à avaliação dos requisitos de qualidade, havia principalmente um requisito de modificabilidade que os profissionais da SIEMENS-AM desejavam garantir o correto atendimento, por isso somente itens que avaliam esse tipo de requisito foram selecionados para compor o *checklist* a ser utilizado durante a inspeção (Figura 5.3). Com isso, o *checklist* utilizado nesse estudo possuía 22 itens de avaliação.

A atividade de Detecção foi conduzida em apenas uma sessão de forma remota, ou seja, os participantes não realizaram a inspeção ao mesmo tempo e nem no mesmo lugar.

Nessa sessão, os candidatos responderam ao Formulário de Caracterização visando identificar a competência e a experiência relacionada ao propósito do estudo, além de se buscar identificar o conhecimento do participante no domínio do problema ao qual o artefato inspecionado pertence.

Após a caracterização, os participantes receberam uma descrição sobre a abordagem ArqCheck, onde os principais conceitos relacionados à aplicação dessa abordagem para a detecção de defeitos estavam descritos.

Além disso, foi distribuído o documento arquitetural a ser avaliado, o documento de requisitos utilizado durante a inspeção, o *checklist* de avaliação configurado e um relatório de discrepâncias que deveria ser preenchido a medida que o participante

identificar discrepâncias no artefato avaliado. Não existe um limite de tempo para que essa avaliação seja realizada, mas solicitamos que os participantes indicassem quanto tempo levaram para realizar a atividade de detecção.

Itens de avaliação da consistência do documento				
Nº	Visão Modular	Sim	Não	NE
4	Para todo componente, foi identificado em al gum diagrama da visão Modular as classes ou sub-componentes que o compõem?			S
5	Todo relacionamento definido entre dois componentes pode ser mapeado para relacionamentos realizados entre classes/sub-componentes que o compõem?			S
6	Todo relacionamento realizado entre classes/sub-componentes alocados em "componentes pais" distintos foi definido como uma dependência entre esses "componentes pais" em algum outro diagrama?			S
7	Toda interface é disponibilizada por um único componente/classe?			S
Nº	Visão de Contexto Geral	Sim	Não	NE
8	Todo relacionamento entre os componentes. descrito na visão de Contexto Geral, pode ser mapeado para os relacionamentos descritos na visão Modular?			S
9	Todo elemento externo que se relaciona diretamente com um componente da arquitetura foi representado na visão Modular?			S
10	Todo relacionamento existente entre um elemento externo e um componente da arquitetura foi mapeado para alguma classe representada na visão Modular?			S
11	Toda interface disponibilizada/implementada pel os componentes descritos na visão de Contexto Geral podem ser mapeadas para um a classe que a disponibiliza/implementa na visão Modular?			S
Nº	Visão Dinâmica	Sim	Não	NE
12	Ao analisar as seqüências de execução, descritas na visão Dinâmica, pode-se dizer que todas as classes definidas na visão Modular foram alocadas em ao menos uma dessas seqüências?			S
13	Toda mensagem trocada entre duas cl asses/componentes, descritas na visão Dinâmica, pode ser mapeada para algum relacionamento definido entre essas classes/componentes da visão Modular?			S
14	Todo relacionamento, descrito na visão Modular, pode ser mapeado para alguma mensagem de comunicação descrita na visão Dinâmica?			S
15	Todo componente externo representado na visão Dinâmica foi representado em algum diagrama da visão Modular?			S

Figura 5.3 – Exemplo de itens do *checklist* utilizado no estudo de observação

Após a conclusão do estudo, foi pedido a cada indivíduo que preenchesse o Questionário de Avaliação Pós-Experimento, para que obtenhamos a sua avaliação qualitativa a respeito da aplicação da abordagem ArqCheck, e dos procedimentos do estudo experimental.

De posse dos Relatórios de Discrepância gerados por cada participante, o pesquisador consolidou os dados e enviou aos profissionais da SIEMENS-AM para que eles nos retornassem quais eram realmente defeitos e uma lista dos defeitos encontrados pela abordagem SIEMENS. Com base nesses dados, pudemos realizar uma análise visando refutar, ou não, a hipótese definida nesse estudo e identificar lições aprendidas.

5.5.4 – Análise dos dados e resultados obtidos

A análise descrita nessa seção foi realizada com base em dados: (1) disponibilizados pela SIEMENS-AM (dados de controle) referente à utilização da

abordagem SIEMENS na avaliação do documento de projeto, e (2) obtidos através do estudo de observação (dados de observação e de investigação), onde o documento arquitetural criado a partir desse documento de projeto foi inspecionado através da abordagem ArqCheck.

Com base nesses dados, procuramos realizar uma análise de ordem quantitativa e qualitativa da abordagem ArqCheck, e também uma caracterização da abordagem proposta em relação à abordagem SIEMENS.

- **Análise quantitativa**

Originalmente, eram esperadas como retorno da SIEMENS-AM informações relacionadas aos resultados das revisões que foram realizadas no documento de projeto que nos foi passado.

Contudo, não foi possível obtermos tais informações. Como retorno, obtivemos somente resultados de revisões que foram realizadas em versões anteriores do artefato que avaliamos. Sendo assim, esses dados disponibilizados criaram o seguinte cenário: durante o estudo de observação, ArqCheck avaliou um documento que foi melhorado através da correção dos defeitos encontrados pela abordagem SIEMENS.

Esse fato fez com que alguns dos questionamentos que definimos durante o planejamento do estudo não pudessem ser respondidos e inviabilizou também refutar a hipótese nula da forma como foi definida.

Contudo, semelhante a um estudo experimental onde cenário similar foi utilizado (BERLING e THELIN, 2004), com base nos dados que foram coletados durante o estudo e os resultados fornecidos pela SIEMENS-AM, ainda foi possível realizar algumas considerações.

A seguir os dados obtidos durante os diferentes tipos de coletas estão descritos e a análise sobre eles foi realizada.

Dados de controle

Os dados de controle consistem em dados reais obtidos através da realização de duas revisões no contexto do projeto "AppMenu X85" e que foram disponibilizadas pela SIEMENS-AM.

A primeira revisão consiste em um *walkthrough* realizado por uma equipe de 4 profissionais (um autor, um moderador e dois especialistas no domínio da solução). A avaliação realizada através desse *walkthrough* foi realizada durante uma única sessão, onde o documento avaliado foi apresentado pelo autor. Após essa apresentação, os especialistas realizaram uma análise do documento de projeto em relação aos requisitos especificados.

A segunda revisão foi realizada com o objetivo de comparar o que estava especificado no documento de projeto e a forma como a solução foi realmente implementada, visando deixar a arquitetura coerente com essa solução. Contudo, as características e objetivos dessa revisão são diferentes em relação aos de ArqCheck, que busca avaliar o documento arquitetural em relação aos requisitos que foram especificados. Portanto, optamos por não utilizar na comparação entre as abordagens os dados obtidos nessa segunda revisão.

Com isso, a partir das informações do *walkthrough* foi possível extrair medidas para as seguintes métricas:

- Quantidade de inspetores que participaram nas revisões;
- Tempo gasto com a revisão arquitetural;
- Quantidade de defeitos encontrados no documento;

Vale ressaltar que o documento de projeto avaliado pelo *walkthrough* é composto tanto por informações arquiteturais quanto por informações de projeto de baixo-nível, tipo de informação que não é avaliado por ArqCheck.

Portanto, para ser possível utilizar esses dados referentes a abordagem SIEMENS como dados de controle, foi necessário a realização de uma análise visando identificar quais dos defeitos detectados são defeitos arquiteturais. Baseado no custo/eficiência da abordagem (quantidade de defeitos encontrados por tempo de inspeção), foi possível estimar a duração total da revisão se ela tivesse avaliado somente as informações arquiteturais (Tabela 5.5).

Tabela 5.5 - Dados de controle (baseado nos resultados obtidos com a abordagem SIEMENS)

Quantidade de inspetores	2
Tempo gasto com a revisão	4 horas
Quantidade de defeitos detectados	5
Custo/eficiência	1,25 defeitos/hora

Dados de observação

Durante a inspeção, cada inspetor preencheu um relatório com as discrepâncias por eles identificadas. Em posse desses dados, o pesquisador fez uma consolidação deles, buscando identificar discrepâncias comuns aos inspetores ou então discrepâncias que com certeza seriam falsos positivos.

As discrepâncias que puderam ser classificadas pelo pesquisador como falso-positivo estavam relacionadas principalmente a informações de projeto de baixo nível que ficaram documento arquitetural, após a adaptação do documento de projeto, e que acabaram sendo identificadas pelo *checklist*, mas que não faziam parte do escopo da avaliação.

Portanto, após a consolidação dessas discrepâncias, esse relatório consolidado foi enviado para os profissionais da SIEMENS-AM que identificaram as discrepâncias que são consideradas defeitos ou não.

Com base no retorno fornecido pela empresa, podemos extrair medidas para as seguintes métricas definidas durante o planejamento do estudo:

- Quantidade de discrepâncias identificadas por participante;
- Quantidade de defeitos detectados por participante;
- Tempo gasto por cada participante com a avaliação do documento;
- Quantidade de defeitos detectados por ArqCheck.

A etapa de detecção de defeitos com ArqCheck foi realizada por dois participantes (Tabela 5.6) e pelo pesquisador (Tabela 5.7). Esses indivíduos realizaram o papel de inspetores.

Tabela 5.6 - Dados obtidos durante a inspeção realizada pelos inspetores

Inspetores	S1	S2
Discrepâncias	72	45
Defeitos	45	26
Falsos positivos	27	19
Tempo gasto	6 horas	3,75 horas
Custo/Eficiência	7,5 defeitos/hora	6,93 defeitos/hora
Quantidade total de defeitos	47	

Após a analise desses dados, percebemos que mesmo o documento avaliado por ArqCheck tendo já sido corrigido, com base nos defeitos encontrados pela abordagem SIEMENS, uma grande quantidade de defeitos ainda foi encontrada.

Tabela 5.7 - Dados obtidos durante a inspeção realizada pelo pesquisador

Inspetores	Pesquisador
Discrepâncias	41
Falsos positivos	14
Defeitos	2
Tempo gasto	2,3 horas
Custo/Eficiência	11,7 defeitos/hora

Contudo, existem algumas limitações em se comparar os dados obtidos através da aplicação da abordagem SIEMENS e a abordagem ArqCheck, uma vez que avaliaram versões diferentes do documento. Contudo, a análise e a comparação desses dados, permitiram a observação de alguns fatos.

1. Viabilidade da abordagem ArqCheck em identificar defeitos

No primeiro estudo experimental, realizado para avaliar a viabilidade da abordagem ArqCheck, um dos pontos investigados foi a capacidade da abordagem em detectar defeitos.

Mesmo não sendo o objetivo desse estudo, os resultados obtidos justificaram essa viabilidade uma vez que os próprios profissionais da SIEMENS-AM identificaram que muitas das discrepâncias identificadas eram defeitos.

2. Eficácia de ArqCheck

Em relação à eficácia de ArqCheck, os participantes tiveram uma eficácia em média de 60% (Gráfico 5.5). Esses resultados não são satisfatórios uma vez que um esforço relativamente alto foi gasto na identificação das discrepâncias mas 40%, em média, não auxiliaram na melhoria da qualidade do documento avaliado.

Um dos principais motivos que levou a esse resultado foi a adaptação do documento de projeto às características exigidas para a aplicação de ArqCheck. Mesmo após a realização desse processo, muitas informações que não possuíam relevância a nível arquitetural continuaram no documento, o que levou os inspetores a identificar um excessivo número de discrepâncias relacionadas a informações contidas no documento que não pertenciam ao escopo da avaliação.

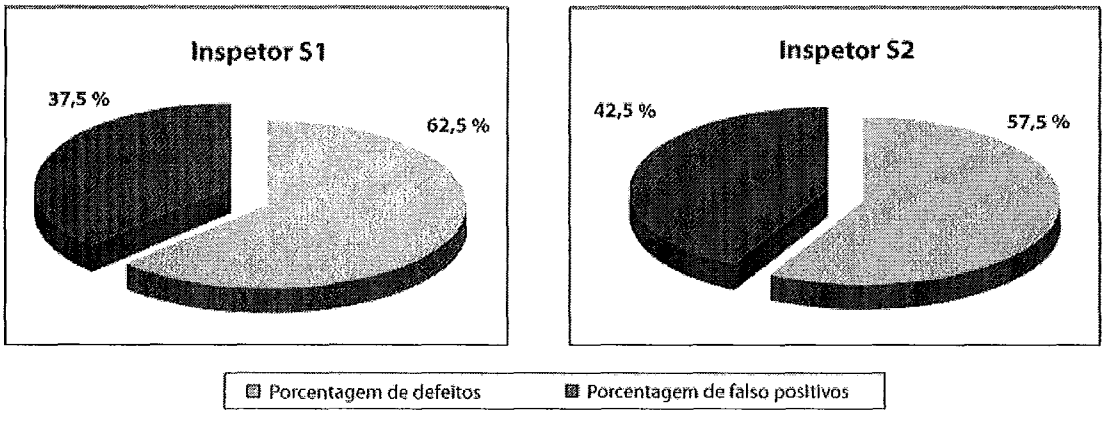


Gráfico 5.5 - Eficácia (defeitos/discrepâncias) de cada participante

Além disso, quando os defeitos encontrados por cada participante são comparados entre si (Figura 5.4). Identificamos uma grande disparidade nos dados obtidos pelo participante S1, principalmente em relação ao número de defeitos exclusivamente encontrados por ele. Ao realizarmos uma comparação entre os defeitos encontrados pelos participantes e pelo pesquisador (Figura 5.5), essa disparidade se mantém.

Ao analisarmos os demais dados relacionados principalmente ao tempo de inspeção e realizarmos uma entrevista onde questionamos como esse participante realizou esse processo, observamos que devido a diversos motivos ele não pode realizar a detecção de defeitos de uma vez.

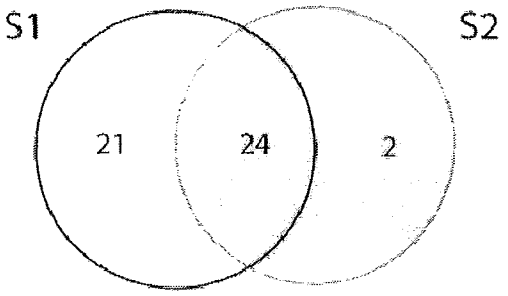


Figura 5.4 - Interseção entre os defeitos encontrados pelos participantes

Sendo assim, devido à necessidade de interromper e retomar a realização dessa atividade por diversas vezes, ele acabou tendo que reavaliar várias vezes determinadas partes do documento e além do mais a realização de inspeções curtas diminuiu os efeitos negativos da fadiga, observados quando se realiza uma inspeção por um longo período de tempo.

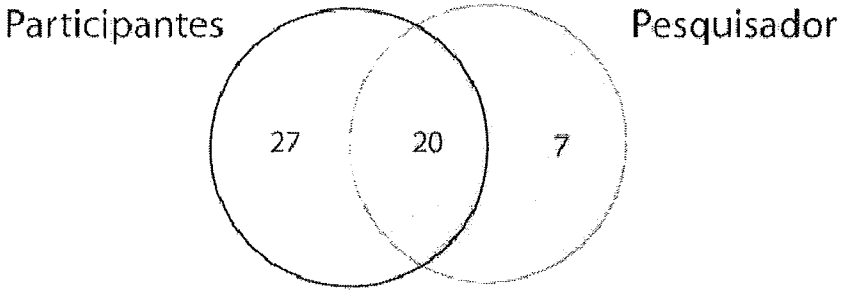


Figura 5.5 - Interseção entre os defeitos encontrados pelos participantes e pelo pesquisador

3. Custo/Eficiência de ArqCheck em relação à abordagem SIEMENS

Embora o documento avaliado por ArqCheck tenha sido previamente corrigido com base nos defeitos encontrados pela abordagem SIEMENS-AM, uma grande quantidade de defeitos foi detectada durante essa segunda avaliação. Além disso, um outro dado que chama a atenção está quando se compara o custo/eficiência das duas abordagens (SIEMENS = 1,25 defeitos/hora; ArqCheck = 7 defeitos/hora), sendo que o tempo gasto por cada uma delas se encontra em uma mesma ordem de grandeza.

Uma das explicações para essa disparidade está no apoio que cada uma das abordagens oferece para a tarefa de detecção de defeitos. A abordagem SIEMENS consiste basicamente em uma abordagem ad-hoc, onde nenhum apoio é fornecido aos inspetores, o que torna os resultados dependentes da experiência do inspetor tanto no domínio da solução quanto na realização desse tipo de revisão. Contudo, a abordagem ArqCheck apóia a detecção de defeitos através de um *checklist*, que fornece uma direção aos inspetores em relação ao que avaliar.

Com isso, uma indicação que podemos ressaltar em relação a esses dados é que a abordagem SIEMENS, embora esteja contribuindo para a melhoria da qualidade, pode precisar ser ajustada para a realização dessa tarefa ou então não esteja sendo aplicada com tempo ou treinamento suficientes. Além do mais, o fato das duas abordagens consumirem uma quantidade de recursos semelhante, tanto em relação ao tempo de inspeção quanto em relação à quantidade de pessoas envolvidas no processo, fornece alguns indícios que apontam para a viabilidade em se implantar ArqCheck no contexto industrial avaliado.

Dados de investigação

Os dados de investigação são utilizados principalmente durante a análise qualitativa do objeto em estudo. Esses dados foram coletados a partir de três fontes: pesquisador, participantes e profissionais da SIEMENS-AM.

O pesquisador forneceu dados tanto em relação à aplicação do processo de configuração do *checklist*, quanto em relação à detecção de defeitos.

Os participantes forneceram informações relacionadas à aplicação da abordagem na detecção de defeitos. Para coletá-las, utilizamos como base o questionário de avaliação pós-experimento (Apêndice E). Nesse questionário buscou-se identificar:

- Os aspectos que facilitam ou dificultam o uso da abordagem;
- A opinião dos participantes em relação a utilidade da abordagem e se eles a utilizariam em uma avaliação arquitetural;

Os profissionais da SIEMENS-AM forneceram informações principalmente relacionadas aos resultados obtidos e as expectativas que eles tinham em relação à aplicação da abordagem ArqCheck.

Com isso, após realizarmos uma consolidação desses resultados, identificamos que os inspetores, por exemplo, consideram mediana a utilização da abordagem ArqCheck.

- Eles a consideraram como tal devido principalmente à dificuldade de se utilizar essa abordagem na avaliação de documentos arquiteturais grandes. Essa dificuldade está relacionada ao fato do *checklist* que compõem ArqCheck possuir itens de avaliação que avaliam a consistência das informações entre as diferentes visões arquiteturais, o que dependendo do tamanho do documento implica em várias verificações que devem ser realizadas, o que acaba tornando a inspeção cansativa e pode influenciar negativamente o resultado final da avaliação.
- Contudo em relação às facilidades, os seguintes aspectos por eles foram apontados em relação a abordagem:

- Permite um mapeamento automático/direto para o tipo do erro, não deixando duvidas em relação a isto;
- É de fácil entendimento, pois os itens deixam claro o escopo que deve ser analisado
- Possui uma boa organização, e pode ser utilizado com um passo a passo, pois normalmente o conhecimento utilizado em um item, auxilia na avaliação realizada no item seguinte, o que facilita a realização de *cross-checking.*
- Em relação aos comentários fornecidos pelos profissionais da SIEMENS-AM, o que se destaca foi o fato da quantidade de defeitos que foram encontrados, principalmente devido ao fato do documento já ter sido avaliado.

5.5.5 – Considerações em relação à hipótese nula

Devido ao fato de não termos dados que permitam uma comparação direta entre as abordagens ArqcCheck e SIEMENS, não foi possível tecer algum tipo de conclusão em relação à hipótese nula, definida durante o planejamento do estudo de observação.

Contudo, os dados coletados durante esse estudo e as análises que puderam ser realizadas não diminuem a importância que foi a sua realização pois com base neles foi possível obter indícios em relação à viabilidade de aplicar ArqCheck no contexto industrial avaliado, identificar pontos de melhorias permitindo a evolução da abordagem e também a oportunidade de se observar a utilização da abordagem proposta para avaliar um documento arquitetural criado em um contexto industrial.

5.5.6 – Lições aprendidas

As lições aprendidas podem ser divididas principalmente em duas categorias. A primeira consiste em informações que auxiliaram na evolução da abordagem.

Uma das melhorias identificadas foi a necessidade da descrição do passo-a-passo a ser seguido durante a configuração do *checklist.* Durante o estudo, percebemos que tal tarefa está muito dependente do conhecimento tácito do pesquisador em realizar essa tarefa e que a descrição desse passo-a-passo facilitaria a realização dessa atividade por outros.

Outra melhoria identificada está relacionada ao "Guia de identificação de contexto" utilizado pelo inspetor como auxílio para a avaliação do documento em relação ao atendimento dos requisitos de qualidade pela arquitetura. Esses guias não estavam sendo totalmente compreendidos, o que dificultou os inspetores na realização desse tipo de avaliação. Como solução para melhorar esse entendimento, definimos que durante a configuração do *checklist,* o moderador da inspeção, com o auxilio do autor do documento, deve realizar a identificação do contexto a ser avaliado, deixando a

cargo do inspetor somente a avaliação da conformidade dos elementos arquiteturais em relação aos requisitos de qualidade indicados.

- Além dessas melhorias, evoluímos os itens de avaliação 8, 9, 10, 11 e 18 (vide Apêndice B.2) pois não estavam sendo totalmente compreendidos.

A segunda categoria de lições aprendidas consiste principalmente de observações que foram realizadas durante esse estudo e que indicam próximos passos a serem seguidos para se ter uma caracterização mais completa de ArqCheck.

Um dessas observações está na dificuldade apresentada pelos inspetores na avaliação de um documento arquitetural maior. Devido a grande quantidade de verificações a serem feitas com a utilização da abordagem, acreditamos que uma forma mais eficiente de utilizar a abordagem seria alocando inspetores, de acordo com os níveis de experiência, para avaliar a arquitetura utilizando somente uma parte do *checklist*. Acreditamos que indivíduos que possuam mais experiência em leitura de documentos arquiteturais poderiam avaliar a consistência das informações entre as diferentes visões arquiteturais e indivíduos com mais experiência em projeto arquitetural poderia ser alocada para avaliar como a arquitetura está atendendo os requisitos especificados pelo cliente.

- Uma outra observação que foi feita em relação aos resultados obtidos foi a quantidade de defeitos detectados pelo participante que teve que interromper a avaliação por diversas vezes e, por conseqüência, avaliar diversas vezes determinadas partes do documento. Uma hipótese inicial do que tenha influenciado esse resultado seja o fato dele não ter sido vítima do desgaste observado quando se inspeciona um documento por um grande período de tempo.
- Contudo estudos experimentais deveriam ser realizados para procurar responder a esses questionamentos.

5.6 – Conclusão

O desenvolvimento de tecnologias de software apoiadas por experimentação tem demonstrado ser uma abordagem adequada, podendo dar um diferencial de qualidade e crédito às tecnologias que vêm sendo desenvolvidas na academia.

Por esse motivo que para avaliar a abordagem proposta, descrita nesse capítulo, realizamos estudos experimentais, utilizando principalmente como base conceitos de uma metodologia para a transferência de tecnologias para a indústria.

A transferência de tecnologias criadas em um contexto acadêmico para um contexto industrial não é uma tarefa trivial. Grande parte de seu sucesso dependerá da cooperação entre as diferentes instituições.

Esse tipo de cooperação, como a existente entre o grupo ESE/COPPE/UFRJ e a empresa multinacional SIEMENS, representada através de seu instituto de pesquisa instalado em Manaus/AM, possibilita que os pesquisadores tenham acesso e realizem estudos usando informações produzidas em um ambiente real e não acadêmico. Esse fato permite a pesquisa por tecnologias que atendam às necessidades da indústria e diminua o *gap* existente entre estado da arte e o estado da prática.

Portanto, através dos estudos que foram realizados até o momento conseguimos identificar a viabilidade da abordagem na identificação de defeitos e também a sua eficácia e custo/eficiência, uma vez que permite identificar defeitos no mínimo de forma tão eficiente e eficaz que uma abordagem utilizada no estado da prática.

Contudo acreditamos, mesmo com todos os avanços, melhorias e maturidade obtidos através dos estudos realizados, ser aconselhável a realização dos dois estudos restantes, de acordo com a metodologia de transferência utilizada como base, para que se avalie a utilização da abordagem como um todo dentro de um processo de desenvolvimento, visando à identificação de melhorias adicionais e o seu completo amadurecimento. O planejamento desses estudos de caso deverá considerar principalmente as características de ciclo de vida presentes em um contexto industrial. Como exemplo, poderia se utilizar as características presentes no ambiente para desenvolvimento de software da SIEMENS-AM.

Desta forma, o primeiro estudo deve avaliar principalmente o impacto da introdução de um processo completo de inspeção no contexto do processo de desenvolvimento e o segundo consistiria de um estudo piloto na indústria onde ArqCheck seria avaliada de forma on-line, para um projeto em andamento.

Capítulo 6 – Conclusões

Nesse capítulo são apresentadas as conclusões deste trabalho, relatando as suas contribuições, limitações e perspectivas futuras.

6.1 – Considerações finais

Com o aumento da complexidade dos sistemas, a avaliação arquitetural está sendo cada vez mais usada visando principalmente a identificação e correção de defeitos o mais cedo possível dentro do processo de desenvolvimento de software. Nos últimos anos, temos observado que essa prática não se restringe mais somente ao contexto acadêmico, ela está sendo cada vez mais adotada em ambientes industriais (MARANZANO *et al.*, 2005).

Contudo, são poucas as abordagens de avaliação arquitetural que estão sendo realmente aplicadas, o que identificamos são principalmente técnicas *ad-hoc* criadas com base nas características específicas do contexto em que são aplicadas.

Durante a pesquisa que resultou nessa dissertação identificamos, através de revisão sistemática, abordagens de avaliação arquitetural descritas na literatura, caracterizamos essas abordagens visando identificar seus benefícios e limitações. Com base nesses dados, sugerimos uma abordagem de avaliação arquitetural que permitisse minimizar essas limitações e avaliamos essa abordagem através de uma série de estudos experimentais visando, principalmente, a sua transferência para um contexto industrial.

A realização dessa pesquisa se destaca pela forma como foi realizada. Utilizando vários conceitos oriundos da Engenharia de Software Experimental, fizemos uso de revisões sistemáticas para identificar as abordagens existentes de forma objetiva e realizamos estudos experimentais para avaliar a abordagem proposta. Além disso, fizemos uso de conhecimentos identificados em diversas abordagens de projeto e documento arquitetural na construção da abordagem de avaliação arquitetural proposta.

Com isso, construímos uma abordagem que visa a inspeção de documentos arquiteturais através do uso de um *checklist*. Essa abordagem foi criada após a observação de que as abordagens descritas na literatura não atendiam a requisitos essenciais para o sucesso de uma tecnologia em um ambiente industrial.

Esses requisitos foram identificados com base em condições normalmente presentes em ambientes industriais e que podem influenciar na adoção de uma determinada tecnologia dentro desse contexto (SHULL *et al.*, 2001).

Com isso, ArqCheck foi desenvolvida visando:

- **Generalidade:** Por generalidade entendemos a possibilidade da abordagem ser aplicada em diferentes contextos de avaliação. Em relação à abordagem proposta, devido ao fato da abordagem proposta de permitir avaliar principalmente a informação contida no documento arquitetural e não a abordagem de representação usada para descrever essas informações, consideramos que ela é genérica o suficiente para poder ser utilizada em diferentes contextos.
- **Adaptabilidade:** Além de permitir uma avaliação genérica, a abordagem proposta pode ser adaptada às características especificas do documento. Essa adaptação é feita principalmente através das atividades de configuração que permitem configurar o *checklist* de acordo com as características do documento a ser avaliado;
- **Simplicidade**: A abordagem proposta não necessita da execução de atividades elaboradas atividades para sua aplicação, da necessidade de tipos de conhecimento específicos ou da alocação de grandes quantidades de recursos, sendo assim aparentemente simples de ser aplicada em um projeto;
- **Extensibilidade**: Por extensibilidade entendemos a necessidade de uma abordagem permitir que novos conceitos, que não tinham sido previstos durante a criação da abordagem, sejam avaliados. No contexto da abordagem proposta, ela possui essa extensibilidade, que é feita a partir da adição de novos itens de avaliação;

6.2 – Contribuições

Com base no que foi realizado durante essa pesquisa, identificamos que as principais contribuições que ela pode oferecer à comunidade de Arquitetura de Software são:

- **Identificação das principais abordagens de avaliação arquitetural descritas na literatura.** Através da realização de uma revisão sistemática nas principais fontes de publicações disponíveis, identificamos as 27 principais abordagens de avaliação arquitetural, diferente das publicações com o mesmo propósito na literatura que não descrevem mais de 8 abordagens. Devido ao tipo abordagem de pesquisa que utilizamos, é possível também que qualquer pesquisador repita essa pesquisa, utilizando o protocolo de revisão sistemática criado (Apêndice A);

- **Caracterização das abordagens.** Com base nas principais características que compõem uma abordagem de inspeção de software, definimos um *framework* de caracterização. Através desse *framework* e das informações coletadas através da revisão sistemática foi possível caracterizar as abordagens identificadas, permitindo um maior entendimento de cada uma delas e a identificação de seus benefícios e limitações;
- **Abordagem para inspeção de documentos arquiteturais baseada em *checklist*.** Visando a criação de ArqCheck principalmente três estudos foram realizados; (1) análise das principais limitações presentes nas abordagens de avaliação arquitetural descritas na literatura, (2) análise do processo de especificação arquitetural e das principais abordagens que podem ser utilizadas para realizá-lo e (3) estudo sobre as abordagens de inspeção de software. Com base nos resultados obtidos após esses estudos, criamos uma abordagem para inspeção arquitetural que visa avaliar documentos arquiteturais. Essa avaliação é feita através de um *checklist* que teve seus itens criados a partir dos tipos de conhecimentos utilizados durante a especificação arquitetural.
- **Planejamento e execução de estudos experimentais visando a avaliação e a transferência dessa tecnologia para a industria.** Além da criação da abordagem, planejamos e executamos estudos experimentais para avaliar a viabilidade e o custo/eficiência da abordagem proposta e transferir essa tecnologia para um contexto industrial. Em relação a esses estudos, eles confirmaram tanto a viabilidade quanto ao custo/eficiência da abordagem. Vale lembrar também, que além de documentos arquiteturais criados em contexto acadêmico, utilizamos também durante esses estudos documentos arquiteturais reais, construídos no contexto de projetos da empresa SIEMENS-AM, o que nos forneceu resultados preliminares sobre a adaptabilidade e extensibilidade da abordagem.

6.3 – Limitações

Ao analisar a abordagem proposta identificamos algumas limitações. Uma delas se relaciona ao tipo de conhecimento utilizado como base para a sua criação, as táticas arquiteturais.

Além de usarmos somente esse tipo de conhecimento para avaliar o atendimento aos requisitos de qualidade, o que pode ser uma avaliação limitada dependendo do contexto em que ela for usada, não foi possível criar itens de avaliação para todas as táticas descritas em (BASS *et al.*, 2003). Esses itens não foram criados, pois muitas das táticas não eram relevantes para a arquitetura do software, mas sim para a

arquitetura do sistema, o que envolvia a necessidade de solução via hardware, que estava fora do escopo da pesquisa.

Mesmo assim, por causa disso não é possível assegurar que todas as características da arquitetura serão avaliadas e que a abordagem permite identificar todos os tipos defeitos. Contudo essa limitação pode ser superada com a criação de novos itens utilizando principalmente o conhecimento de especialistas da área.

Além disso, os estudos realizados forneceram somente resultados preliminares, e não conclusivos, sobre a abordagem avaliada. Isso ocorreu por não termos um controle adequado em grande parte das variáveis envolvidas nos estudos. Como por exemplo, nem todos os itens que compõem o *checklist* puderam ser avaliados pela impossibilidade de se obter arquiteturas de software que possuíssem características relacionadas à disponibilidade ou à testabilidade e que poderia ser usadas em experimentos que avaliaram a aplicabilidade dos itens relacionados. Além disso, a quantidade de participantes nos estudos não permitiu a obtenção de resultados relevantes a nível estatístico.

Ainda em relação aos estudos experimentais, a metodologia experimental de transferência tecnológica não foi totalmente executada e, como resultado, os estudos que previam um maior controle sobre as variáveis envolvidas não foram realizados.

Em relação à aplicação da abordagem proposta no estado da prática identificamos dois pontos que podem limitar o contexto onde ela pode ser utilizada:

- Acreditamos que para a sua aplicação, seja necessário que antes de tudo a empresa sinta a necessidade de realizar esse tipo de avaliação. Para isso, a empresa já deve ter a maturidade requerida para a realização de atividades relacionadas a validação e verificação dos artefatos produzidos, características comuns a empresas que possuem CMMi nível 3 / MpsBr nível C.
- A necessidade de descrever a arquitetura através de um documento arquitetural que atenda as necessidades requeridas por ArqCheck. Como vimos no estudo de viabilidade, muitas vezes essas informações estão juntas com informações de projeto de baixo-nível e criar um documento somente com informações relevantes a nível arquitetural pode exigir recursos adicionais e defeitos podem ser inseridos durante essa criação.

6.4 – Trabalhos Futuros

Em curto prazo, as próximas atividades de pesquisa relacionadas a ArqCheck serão realizadas visando principalmente a execução das etapas restantes da metodologia experimental proposta por (SHULL *et al.*, 2001). Esse direcionamento foi

definido visando principalmente amadurecer a abordagem proposta e introduzi-la em um contexto industrial.

Contudo, em longo prazo pensamos principalmente na melhoria dos itens de avaliação, utilizando principalmente conhecimento oriundo de especialistas em diferentes áreas da qualidade de software, como usabilidade e desempenho, por exemplo.

Além disso, um outro trabalho que pode ser feito é relacionada à construção de uma ferramenta de apoio às atividades do processo de inspeção que compõem a abordagem proposta. Uma possível solução seria utilizar a ferramenta ISPIS (KALINOWSKI, 2004) visto que ela apóia grande parte dessas atividades. Contudo, seria necessário fazer uma análise dessa ferramenta visando identificar como ela poderia apoiar computacionalmente as atividades específicas da abordagem proposta, como por exemplo, a configuração do *checklist*.

Referências Bibliográficas

ABOWD, G., PITKOW, J., KAZMAN, R., 1996, "Analyzing differences between Internet information system software architectures". In: *Proceedings of the International Conference on Communications*, v. 1, pp. 203-207, Dallas, TX, June.

ABOWD, G., BASS, L., CLEMENTS, P., KAZMAN, R., NORTHROP, L., ZAREMSKI, A., 1997, *Recommended Best Industrial Practice for Software Architecture Evaluation*, SEI, Carnegie Mellon University, Relatório Técnico, CMU/SEI-96-TR-025.

ABOWD, G.D., 1995, "Defining reference models and software architectural styles for cooperative systems", *ACM SIGOIS Bulletin*, v. 15, n. 3 (Abril), pp. 4-5.

ACM, 2004, "Association for Computing Machinery (ACM) Digital Library". In: http://portal.acm.org/dl.cfm Acessado em Dezembro / 2005.

AMARAL, E.A.G., 2003, *Empacotamento de Experimentos em Engenharia de Software*, Dissertação de Mestrado, Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ, Rio de Janeiro.

AURUM, A., PETERSSON, H., WOHLIN, C., 2002, "State-of-the-art: software inspections after 25 years", *Software Testing, Verification and Reliability*, v. 12, n. 3, pp. 133-154.

BABAR, M.A., GORTON, I., 2004, "Comparison of scenario-based software architecture evaluation methods". In: *Proceedings of the Asia-Pacific Software Engineering Conference*, pp. 600-607, Busan, South Korea, November.

BABAR, M.A., ZHU, L., JEFFERY, R., 2004, "A framework for classifying and comparing software architecture evaluation methods". In: *Proceedings of the Australian Software Engineering Conference*, pp. 309-318, Melbourne, Australia, April.

BACHMANN, F., BASS, L., CHASTEK, G., DONOHOE, P., PERUZZI, F., 2000, *The Architecture Based Design Method*, CMU/SEI, Relatório Técnico, CMU/SEI-2000-TR-001.

BAHSOON, R., 2003, "Evaluating Software Architectures for Stability: A Real Options Approach". In: *Proceedings of the Doctral Symposium of the International Conference on Software Engineering*, pp. 765, Portland, USA.

BAHSOON, R., EMMERICH, W., 2003, "Evaluating software architectures: development, stability, and evolution". In: *Book of Abstracts of the ACS/IEEE International Conference on Computer Systems and Applications*, pp. 47, Tunis, Tunisia, July.

BARBER, K.S., GRASER, T., HOLT, J., 2001, "Providing early feedback in the development cycle through automated application of model checking to software architectures". In: *16th Annual International Conference on Automated Software Engineering*, pp. 341-345, Coronado Island, San Diego, CA, USA, November.

BARCELOS, R.F., TRAVASSOS, G.H., 2005, "Avaliando documentos arquiteturais através de um checklist baseado em atributos de qualidade". In: *Proceedings of Workshop de Teses e Dissertação de Engenharia de Software (WTES) - SBES*, Uberlândia, MG, Brasil, Outubro.

BARCELOS, R.F., TRAVASSOS, G.H., 2006a, "ArqCheck: Uma abordagem para inspeção de documentos arquiteturais baseada em checklist". In: *Proceedings of the Simpósio Brasileiro de Qualidade de Software*, Vilha Velha, ES, Brasil, Junho.

BARCELOS, R.F., TRAVASSOS, G.H., 2006b, "Evaluation Approaches for Software Architectural Documents: a Systematic Review". In: *Proceedings of the Ibero-American Workshop on Requirements Engineering and Software Environments (IDEAS)*, Buenos Aires, Argentina, Abril.

BASILI, V., CALDIEIRA, G., ROMBACH, H., 1994, "Goal Question Metrics Paradigm". In: MARCINIAK, J.J. (eds), *Encyclopedia of Software Engineering*, Wiley.

BASILI, V.R., SELBY, R.W., 1987, "Comparing the Effectiveness of Software Testing Strategies", *IEEE Transactions on Software Engineering*, v. 13 (Dezembro), pp. 1278-1296.

BASILI, V.R., SHULL, F., LANUBILE, F., 1999, "Building Knowledge Through Families of Experiments", *IEEE Transactions on Software Engineering*, v. 25, n. 4 (July), pp. 456 - 473.

BASS, L., CLEMENTS, P., KAZMAN, R., 2003, *Software Architecture in Practice, Second Edition*, Addison Wesley.

BENARIF, S., CHERIF, A.R., LEVY, N., LOSAVIO, F., 2004, "Intelligent tool based-agent for software architecture evaluation". In: *Proceedings of the 4th International Conference on Quality Software*, pp. 126-133, Braunschweig, Germany, Setembro.

BENGTSSON, O., BOSCH, J., 1999, "Architecture Level Prediction of Software Maintenance". In: *Proceedings of the Third European Conference on Software Maintenance and Reengineering*, pp. 139, Amsterdam, The Netherlands, March.

BENGTSSON, P., BOSCH, J., 1998, "Scenario-Based Software Architecture Reengineering". In: *Proceedings of the 5th International Conference on Software Reuse*, pp. 308, Victoria, B.C, Canada, June.

BENGTSSON, P., LASSING, N., BOSCH, J., VAN VLIET, H., 2004, "Architecture-Level Modifiability Analysis (ALMA)", *Journal of Systems and Software*, v. 69, n. 1-2 (Janeiro), pp. 129-147.

BENGTSSON, P.O., 2002, *Architecture-Level Modifiability Analysis*, Tese de Doutorado, Blekinge Institute of Technology.

BERLING, T., THELIN, T., 2004, "A Case Study of Reading Techniques in a Software Company". In: *Proceedings of the International Symposium on Empirical Software Engineering*, pp. 229-238, Redondo Beach, CA, USA, August.

BIOLCHINI, J., MIAN, P.G., NATALI, A.C.C., TRAVASSOS, G.H., 2005, *Systematic Review in Software Engineering*, COPPE / UFRJ, Relatório Técnico, ES-679/05.

BOEHM, B.W., 1981, *Software Engineering Economics*, Prentice-Hall.

BOEHM, B.W., ABTS, C., BROWN, A.W., CHULANI, S., CLARK, B.K., HOROWITZ, E., MADACHY, R., REIFER, D., STEECE, B., 2000, *Software Cost Estimation with COCOMO II*, Prentice Hall.

BOEHM, B.W., BASILI, V.R., 2001, "Software Defect Reduction Top 10 List", *IEEE Computer*, v. 34, n. 1 (Janeiro), pp. 135-137.

BOSCH, J., MOLIN, P., 1999, "Software Architecture Design: Evaluation and Transformation". In: *Proceedings of the IEEE Engineering of Computer Based Systems Symposium (ECBS'99)*, pp. 4, Nashville, TN, USA, March.

BOSCH, J., 2000, *Design and use of software architectures: adopting and evolving a product-line approach*, ACM Press/Addison-Wesley Publishing Co.

BUSCHMANN, F., MEUNIER, R., ROHNERT, H., SOMMERLAD, P., STAL, M., 1996, *Pattern-Oriented Software Architecture: A System of Patterns*, Jon Wiley and Sons.

CARRIERE, S.J., KAZMAN, R., WOODS, S.G., 1999, "Assessing and maintaining architectural quality". In: *Proceedings of the Third European Conference on Software Maintenance and Reengineering*, pp. 22-30, Amsterdam, The Netherlands, March.

CARVER, J., LEMON, K., 2005, "Architecture Reading Techniques: A Feasibility Study". In: *Proceedings of the International Symposium on Empirical Software Engineering*, pp. 17-20, Noosa Heads, Australia, Novembro.

CHEN, T.Y., POON, P.L., TANG, S.F., 2002, "Towards a Problem-Driven Approach to Perspective-Based Reading". In: *Proceedings of the 7th IEEE International Symposium on High Assurance Systems Engineering (HASE'02)*, pp. 221-229, Washington, DC, USA, October.

CHOI, H., YEOM, K., 2002, "An approach to software architecture evaluation with the 4+1 view model of architecture". In: *Proceedings of the Asia-Pacific Software*

Engineering Conference, pp. 286-293, Gold Coast, Queensland, Australia, December.

CIOLKOWSKI, M., SHULL, F., BIFFL, S., 2002, "A Family of Experiments to Investigate the Influence of Context on the Effect of Inspection Techniques". In: *Proceedings of the 6th International Conference on Empirical Assesment in Software Engineering (EASE)*, pp. 48-60, Keele, UK, April.

CLEMENTS, P., KAZMAN, R., KLEIN, M., 2002, *Evaluating Software Architectures: Methods and Case Studies*, Addison-Wesley.

CLEMENTS, P., BACHMANN, F., BASS, L., GARLAN, D., IVERS, J., LITTLE, R., NORD, R., STAFFORD, J., 2004, *Documenting Software Architectures*, Addison-Wesley.

CLEMENTS, P.C., 2000, *Active Reviews for Intermediate Designs*, CMU/SEI, Relatório Técnico, CMU/SEI-2000-TN-009.

CONRADI, R., MOHAGHEGHI, P., ARIF, T., 2003, "Object-Oriented Reading Techniques for Inspection of UML Models - An Industrial Experiment". In: *Proceedings of the European Conference on Object-Oriented Programming*, pp. 483-500, Darmstadt, Germany, July.

DIAS, M.S., VIEIRA, M.E.R., 2000, "Software architecture analysis based on statechart semantics". In: *International Workshop on Software Specification and Design*, pp. 133-137, Washington, DC, USA.

DOBRICA, L., NIEMELA, E., 2002, "A survey on software architecture analysis methods", *IEEE Transactions on Software Engineering*, v. 28, n. 7, pp. 638-653.

DUENAS, J.C., DE OLIVEIRA, W.L., DE LA PUENTE, J.A., 1998, "A Software Architecture Evaluation Model". In: *Procedure of the Second Int'l ESPRIT ARES Workshop*, pp. 148-157, Feb.

EGYED, A., MEDVIDOVIC, N., 1999, "Extending Architectural Representation in UML with View Integration". In: *Proceedings of the 2nd International Conference on the Unified Modeling Language. Beyond the Standard (UML'99)*, v. 1723, pp. 2-16, Fort Collins, USA, October.

EICKELMANN, N.S., RICHARDSON, D.J., 1996, "An evaluation of software test environment architectures". In: *Proceedings of the International conference on Software Engineering (ICSE)*, pp. 353-364, Washington, DC, USA.

ERICKSON, R.L., GRIFFETH, N.D., LAI, M.Y., WANG, S.Y., 1993, "Software architecture review for telecommunications software improvement". In: *IEEE International Conference on Communications*, v. 2, pp. 616-620 vol.2.

FAGAN, M.E., 1976, "Design and code inspection to reduce Errors in Program Development", *IBM Systems Journal*, v. 15, n. 3, pp. 182-211.

FALBO, R., 1998, *Integração de Conhecimento em um Ambiente de Desenvolvimento de Software*, Tese de Doutorado, Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ.

FOLMER, E., V. GURP, J., BOSCH, J., 2003, "A framework for capturing the relationship between usability and software architecture". In: *Software Process: Improvement and Practice*, pp. 67-87.

FOLMER, E., BOSCH, J., 2005, "Case studies on Analyzing Software Architectures for Usability". In: *Proceedings of the EUROMICRO Conference on Software Engineering and Advanced Applications*, pp. 206-213.

FOLMER, E., 2005, *Software Architecture Analysis of Usability*, Tese de Doutorado, RuG.

GACEK, C., 1995, *On the Definition of Software System Architecture*, University of Southern California, Relatório Técnico, USC/CSE-95-TR-500.

GAMMA, E., HELM, R., JOHNSON, R., VLISSIDES, J., 1995, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley.

GANNOD, G.C., LUTZ, R.R., 2000, "An approach to architectural analysis of product lines". In: *Proceedings of the International conference on Software Engineering (ICSE)*, pp. 548-557.

GARLAN, D., SHAW, M., 1993, "An Introduction to Software Architecture". In: *Advances in Software Engineering and Knowledge Engineering*, v. I.

GARLAN, D., PERRY, D., 1995, "Introduction to the Special Issue on Software Architecture". In: *IEEE Transactions on Software Engineering*, v. 21, April.

GARLAN, D., 2000, "Software architecture: a roadmap". In: *Proceedings of The Conference on The Future of Software Engineering*, pp. 91-101.

GLOGER, M., JOCKUSCH, S., WEBER, N., 1996, "Assessment and optimization of system architectures-experience fromindustrial applications at Siemens". In: *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems*, pp. 400-407.

GRAAF, B., VAN DIJK, H., VAN DEURSEN, A., 2005, "Evaluating an Embedded Software Reference Architecture: Industrial Experience Report". In: *Proceedings of the European Conference on Software Maintenance and Reengineering*, pp. 354-363.

HOFMEISTER, C., NORD, R.L., SONI, D., 2000, *A study on agreement between participants in an architecture assessment*, Addison-Wesley.

HOLLOCKER, C.P., 1990, *Software Reviews and Audits Handbook*, New York, John Wiley \& Sons, Inc.

IEEE, 1988, "IEEE Standard for Software Reviews and Audits - IEEE Standard 1028-1988", *Institute of Electrical and Eletronics Engineers.*

IEEE, 2000, "IEEE Recommended Practice For Architectural Description Of Software-Intensive Systems - IEEE Standard 1471-2000", *Institute of Electrical and Electronics Engineers.*

IEEE, 2004, "IEEE Xplore". In: http://ieeexplore.ieee.org/Xplore/guesthome.jsp Acessado em Dezembro / 2005.

ISO/IEC, 1998, "International Technology - Software Product Evaluation - ISO/IEC 9126 Part 1: Quality Model".

IVEZIC, N., BARBACCI, M., LIBES, D., POTOK, T., ROBERT, J., 2000, "An analysis of a supply chain management agent architecture". In: *Proceedings of the International Conference on MultiAgent Systems*, pp. 401-402.

JAZAYERI, M., 2002, "On Architectural Stability and Evolution". In: *Proceedings of the Ada-Europe International Conference on Reliable Software Technologies*, pp. 13-23, London, UK.

JOHANSSON, E., HÖST, M., WESSLÉN, A., BRATTHAL, L., 2001, "The Importance of Quality Requirements in Software Platform Development - A Survey". In: *Proceedings of HICSS-34*, Maui, Hawaii, January.

JONES, C., 1991, *Applied Software Measurement*, McGraw Hill.

KALINOWSKI, M., 2004, *Infra-Estrutura Computacional de Apoio ao Processo de Inspeção de Software*, Dissertação de Mestrado, Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ, Rio de Janeiro.

KAZMAN, R., BASS, L., ABOWD, G., WEBB, M., 1994, "SAAM: a method for analyzing the properties of software architectures". In: *Proceedings of the International conference on Software Engineering (ICSE)*, pp. 81-90.

KAZMAN, R., ABOWD, G., BASS, L., CLEMENTS, P., 1996, "Scenario-based analysis of software architecture", *IEEE Software*, v. 13, n. 6, pp. 47-55.

KAZMAN, R., KLEIN, M., CLEMENTS, P., 2000, *ATAM: Method for Architecture Evaluation*, CMU/SEI, Relatório Técnico, CMU/SEI-2000-TR-004.

KAZMAN, R., 2001, "Handbook of Software Engineering and Knowledge Engineering". In: CHANG, S.K. (eds), World Scientific Publishing.

KAZMAN, R., ASUNDI, J., KLEIN, M., 2001, "Quantifying the costs and benefits of architectural decisions". In: *Proceedings of the International conference on Software Engineering (ICSE)*, pp. 297-306.

KAZMAN, R., BASS, L., 2002, "Making architecture reviews work in the real world", *IEEE Software*, v. 19, n. 1, pp. 67-73.

KITCHENHAM, B., 2004, *Procedures for Performing Systematic Reviews*, Keele University, Relatório Técnico.

KITCHENHAM, B.A., PFLEEGER, S.L., PICKARD, L.M., JONES, P.W., HOAGLIN, D.C., EMAM, K.E., ROSENBERG, J., 2002, "Preliminary Guidelines for Empirical Research in Software Engineering", *IEEE Transactions on Software Engineering*, v. 28, n. 8 (Agosto), pp. 721-734.

KRUCHTEN, P., 1995, "Architectural Blueprints - The "4+1" View Model of Software Architecture". In: *IEEE Software*, v. 12, pp. 42-50, November.

LAITENBERGER, O., DEBAUD, J., 1998, *Scenarios, Quality Attributes, and Patterns: Capturing and Using their Synergistic Relationships for Product Line Architectures*, Fraunhofer Institute Experimental Software Engineering, Relatório Técnico, ISERN-98-32.

LAITENBERGER, O., ATKINSON, C., 1999, "Generalizing Perspective-based Inspection to handle Object-Oriented Development Artifacts". In: *Proceedings of the International conference on Software Engineering (ICSE)*.

LAND, R., 2002, "Improving Quality Attributes of a Complex system Through Architectural analysis - A Case Study". In: *Proceedings of the 9th IEEE Conference on engineering of Computer-Based Systems*, pp. 167-174.

LASSING, N., RIJSENBRIJ, D., VAN VLIET, H., 1999a, "On Software Architecture Analysis of Flexibility - Complexity of Changes: Size isn't Everything". In: *the Second Nordic Workshop on Software Architecture (NOSA99)*.

LASSING, N., RIJSENBRIJ, D., VAN VLIET, H., 1999b, "Towards a broader view on software architecture analysis of flexibility". In: *Sixth Asia Pacific Software Engineering Conference*, pp. 238-245.

LATTANZE, A.J., 2005, *The Architecture Centric Development Method*, Carnegie Mellon University, Relatório Técnico.

LEE, J., HA, S., KANG, K.C., CHOO, Y., HONG, Y., HWANG, H., 2001, "Quality requirement elicitation for the architecture evaluation of process computer systems". In: *Procedings of the Eighth Asia-Pacific Software Engineering Conference (APSEC)*, pp. 335-340.

LEE, J.E., CHOI, K., DUTT, N.D., 2003, "Evaluating memory architectures for media applications on coarse-grained reconfigurable architectures". In: *Proceedings of the IEEE International Conference on Application-Specific Systems*.

LEE, Y., CHOI, H.-J., 2005, "Experience of combining qualitative and quantitative analysis methods for evaluating software architecture". In: *Fourth Annual ACIS International Conference on Computer and Information Science*, pp. 152-157.

LICHTNER, K., ALENCAR, P., COWAN, D., 2000, "A framework for software architecture verification". In: *Australian Software Engineering Conference*, pp. 149-157.

LUNG, C.-H., BOT, S., KALAICHELVAN, K., KAZMAN, R., 1997, "An approach to software architecture analysis for evolution and reusability". In: *Proceedings of the 1997 Conference of the Centre For Advanced Studies on Collaborative Research*, pp. 15.

MAFRA, S.N., TRAVASSOS, G.H., 2005, "Técnicas de Leitura de Software: Uma revisão Sistemática". In: *Proceedings of the Simpósio Brasileiro de Engenharia de Software*, Uberlândia, MG, Brasil, Outubro.

MAFRA, S.N., TRAVASSOS, G.H., 2006, *Estudos primários e secundários apoiando a busca por evidência em engenharia de software*, COPPE / UFRJ, 687/06.

MARANZANO, J.F., ROZSYPAL, S.A., ZIMMERMAN, G.H., WARNKEN, G.W., WIRTH, P.E., WEISS, D.M., 2005, "Architecture reviews: practice and experience", *IEEE Software*, v. 22, n. 2, pp. 34-43.

MCCRICKARD, D.S., ABOWD, G.D., 1996, "Assessing the impact of changes at the architectural level: a case study on graphical debuggers". In: *Proceedings of the International Conference on Software Maintenance*.

MCT/SEPIN, 2005, "Qualidade e Produtividade no Setor de Software". In: http://www.mct.gov.br/sepin/Dsi/Software/Menu_Qualidade.htm Acessado em Fevereiro / 2006.

MEDVIDOVIC, N., ROSENBLUM, D., TAYLOR, R., 1999, "A Language and Environment for Architecture-Based Software Development and Evolution". In: *Proceedings of the International conference on Software Engineering (ICSE)*, Los Angeles, CA, Maio.

MELO, W., SHULL, F., TRAVASSOS, G.H., 2001, *Software Review Guidelines*, COPPE/UFRJ, Relatório Técnico, ES-556/01.

MIAN, P., CHAPETTA, W., SANTOS, P.S., JR, C.R.M., NATALI, A.C.C., BIOLCHINI, J., ROCHA, A.C.R., TRAVASSOS, G., ROCHA, A.R., 2005, "eSEE: an Infrastructure for Supporting Experimental Software Engineering". In: *Proceedings the 4th IEEE/ACM International Symposium on Empirical Software Engineering (ISESE) - Late Breaking Paper*, Australia, November.

MOLTER, G., 1999, "Integrating SAAM in Domain-Centric and Reuse-based Development Processes". In: *2nd Nordic Workshop on Software Architecture*.

NASA, 1993, *Software Formal Inspection Guidebook*, NASA, Relatório Técnico, NASA-STD-2202-9, Disponível em: citeseer.ist.psu.edu/aeronautics93software.html

NETO, A.C.D., BARCELOS, R.F., CHAPETTA, W.A., SANTOS, P.S.M., MAFRA, S.N., TRAVASSOS, G.H., 2004, "Infrastructure for Software Engineering Experiments Definition and Planning". In: *Proceedings of the Experimental Software Engineering Latin American Workshop*, Brasilia.

NETO, A.C.D., TRAVASSOS, G.H., 2005, "Uma infra-estrutura Computacional para Apoio ao Planejamento e Controle de Teste de Software". In: *Proceedings of Workshop de Teses e Dissertação de Qualidade de Software (WTQS) - SBQS*, Porto Alegre, RS, Brasil.

PEREZ, M., GRIMAN, A., LOSAVIO, F., 2002, "Simulation-based architectural evaluation for collaborative systems". In: *Proceedings of the 22nd International Conference of the Chilean Computer Science Society*, pp. 204 - 213.

PFLEEGER, S.L., 2004, *Engenharia de Software – Teoria e Prática*, Prentice Hall.

PORTER, A., VOTTA, J.R., BASILI, V., 1995, "Comparing Detection Methods for Software Requirements Inspection: A Replicted Experiment", *IEEE Transactions on Software Engineering*, v. 21, n. 6, pp. 563-575.

PRESSMAN, R.S., 2001, *Software Engineering: A Practitioner's Approach*, Fifth ed., McGraw Hill.

PURHONEN, A., 2004, "Performance optimization of embedded software architecture - a case study". In: *Proceedings of the Working IEEE/IFIP Conference on Software Architecture (WICSA)*, pp. 112-121.

SEI, 2000, *CMMI-SE/SW: Capability Maturity Model – Integrated for Systems Engineering/Software Engineering. Version 1.0 staged representation*, Software Engineering Institute, Carnegie Mellon University, Relatório Técnico, 2000-TR-012.

SEI, 2004, "SEI Reports". In: http://www.sei.cmu.edu/publications/index.html Acessado em Dezembro / 2005.

SHAW, M., 1995, "Some Patterns for Software Architectures".

SHAW, M., GARLAN, D., 1996, *Software Architecture - Perspectives on an Emerging Discipline*, Prentice Hall.

SHULL, F., RUS, I., BASILI, V., 2000, "How perspective-based reading can improve requirements inspections", *IEEE Computer*, v. 33, n. 7, pp. 73-79.

SHULL, F., CARVER, J., TRAVASSOS, G.H., 2001, "An Empirical Methodology for Introducing Software Processes". In: *Proceedings of European Software Engineering Conference*, pp. 288-296, September.

SHULL, F.J., 1998, *Developing techniques for using software documents: a series of empirical studies*, Tese de Doutorado, University of Mariland.

SILVA, L.F.S., 2004, *Uma abordagem com apoio ferramental para aplicação de técnicas de leitura baseada em perspectiva*, Dissertação de Mestrado, Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ.

SMITH, D., MERSON, P., 2003, "Using architecture evaluation to prepare a large Web based system for evolution". In: *Proceedings of the IEEE International Workshop on Web Site Evolution*, pp. 85-92.

SVAHNBERG, M., WOHLIN, C., LUNDBERG, L., MATTSSON, M., 2002, "A Method for Understanding Quality Attributes in Software Architecture Structures". In: *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE 2002)*, pp. 819-826, New York NY.

SVAHNBERG, M., 2003, "A study on agreement between participants in an architecture assessment". In: *Proceedings of the International Symposium on Empirical Software Engineering (ISESE)*, pp. 61-70.

TEKINERDOGAN, B., 2004, "ASAAM: aspectual software architecture analysis method". In: *Proceedings of the Fourth Working IEEE/IFIP Conference on Software Architecture*, pp. 5-14.

TICHY, W.F., 1998, "Should Computer Scientists Experiment More?" *IEEE Software*, v. 31, n. 5, pp. 32-39.

TRAVASSOS, G.H., SHULL, F., FREDERICKS, M., BASILI, V.R., 1999, "Detecting Defects In Object-Oriented Designs: Using Reading Techniques To Increase Software Quality". In: *Proceedings of the Conference on Object-Oriented Programming Systems, Languages \& Applications.*

TRAVASSOS, G.H., SHULL, F., CARVER, J., 2001, "Working with UML: A Software Design Process Based on Inspections for the Unified Modeling Language". In: *Advances in Computer*, v. 54, pp. 35-97.

TRAVASSOS, G.H., SHULL, F., CARVER, J., BASILI, V.R., 2002, *Reading Techniques for OO Design Inspections*, Programa de Engenharia de Software - COPPE/UFRJ, Relatório Técnico.

TVEDT, R.T., COSTA, P., LINDVALL, M., 2002, "Does the code match the design? A process for architecture evaluation". In: *Proceedings of the International Conference on Software Maintenance*, pp. 393 - 401, October.

VILLELA, K., 2004, *Definição e construção de ambientes de software orientados à organização*, Tese de Doutorado, Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ.

WEBER, K.C., ROCHA, A.R., ROUILLER, A.C.K., 2004, "Uma Estratégia para Melhoria de Processo de Software nas Empresas Brasileiras". In: *Proceedings of the 5th Conference for Quality in Information and Communications.*

WEINBERG, G.M., FREEDMAN, D.P., 1984, "Reviews, walk-Through and Inspections", *IEEE Transactions on Software Engineering*, v. 10, n. 5, pp. 68-72.

WILLIAMS, L.G., SMITH, C.U., 1998, "Performance evaluation of software architectures". In: *Proceedings of the first international workshop on Software and performance (WOSP)*, pp. 164-177.

WOHLIN, C., RUNESON, P., HÖST, M., OHLSSON, M.C., REGNELL, B., WESSLÉN, A., 2000, "Experimentation in Software Engineering: an Introduction", Massachusetts.

XAVIER, J.R., 2001, *Criação e Instanciação de Arquiteturas de Software Específicas de Domínio no Contexto de uma Infra-estrutura de Reutilização*, Dissertação de Mestrado, Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ.

YOUNESSI, H., 2002, *Object-oriented Defect Management of Software*, Prentice Hall.

ZHU, L., BABAR, M.A., JEFFERY, R., 2004, "Mining patterns to support software architecture evaluation". In: *Proceedings of the 4th Working IEEE/IFIP Conference on Software Architecture*, pp. 25 - 34, June.

Apêndice A – Protocolo de Revisão Sistemática

A.1 – Formulação da pergunta

A.1.1 – Foco da pergunta de pesquisa

Identificar e caracterizar abordagens utilizadas para avaliar a qualidade de documentos arquiteturais;

A.1.2 – Qualidade e Amplitude da pergunta

Contexto: Essas abordagens de avaliação devem ter sido executadas em projetos de desenvolvimento de software que (1) possuam seus requisitos bem definidos, (2) possuam atividades que apóiem a construção da arquitetura do software a ser implementado e (3) utilizam abordagens de avaliação para determinar se os atributos de qualidade desejados foram respeitados.

Pergunta: De que forma uma arquitetura de software ou um documento arquitetural são avaliados? Quais as abordagens e em que contexto elas são utilizadas?

Palavras-chave: "software architecture", "software modularity", "architectural model", "high level design", "high level model", "evaluate", "assurance", "review", "inspection", "verification", "analysis";

Intervenção: abordagens de avaliação de arquiteturas de software

Efeito: Caracterização

Controle: (Não possui por ser uma caracterização)

Medida de desfecho: número de abordagens encontradas

População: estudos primários que abordam avaliação arquitetural.

Aplicação: projetistas de software

Projeto Experimental: nenhum método estatístico será aplicado

A.2 – Seleção de fontes de pesquisa

A.2.1 – Critérios de seleção de fontes

Disponibilidade de consulta através da web; possuírem um nível de avaliação Qualis Capes superior à C;

A.2.2 – Linguagem dos estudos

Inglês

A.2.3 – Identificação das fontes

Métodos de busca nas fontes: Leitura dos artigos e, quando for máquina de busca, busca através dos mecanismos de busca disponibilizados

Listagem de fontes: Portal IEEE (IEEE Xplore)(IEEE, 2004), Portal da ACM (ACM, 2004), Relatórios técnicos do SEI(SEI, 2004), Livros da área (Documenting Software Architecture (CLEMENTS *et al.*, 2004), Evaluating Software Architecture (CLEMENTS *et al.*, 2002) e Software Architecture in Practice, Second Edition (BASS *et al.*, 2003)).

Strings de busca:

IEEE Xplore
(software <and> <not>(hardware,synthesize,circuit) <and> architecture <phrase> (<or> (evaluat*,assurance*,review*,inspection,verification, analysis))) <in> metadata
(software <and> <not>(hardware,synthesize,circuit) <and> high <and> level <and> design <phrase> (<or> (evaluat*,assurance*,review*,inspection,verification, analysis))) <in> metadata
(software <and> <not>(hardware,synthesize,circuit) <and> high <and> level <and> model <phrase> (<or> (evaluat*,assurance*,review*,inspection,verification, analysis))) <in> metadata
(software <and> <not>(hardware,synthesize,circuit) <and> modularity <phrase> (<or>(evaluat*,assurance*,review*,inspection,verification, analysis))) <in> metadata
(software <and> <not>(hardware,synthesize,circuit) <and> component <phrase> (<or> (evaluat*,assurance*,review*,inspection,verification, analysis))) <in> metadata
ACM
+abstract:software +abstract:architecture abstract:evaluate abstract:assurance abstract:review abstract:inspection abstract:verification abstract:analysis -abstract:hardware -abstract:synthesize -abstract:circuit -"natural language" -CAD -"memory optimization" -processors -microprocessors -java -"human computer interaction" -memory -kernel -gcc -nlp -holonic -"hypermedia systems" -olap -networking -"Programming Environments" -"Software process models"
+abstract:software +abstract:"high level design" abstract:evaluate abstract:assurance abstract:review abstract:inspection abstract:verification abstract:analysis -abstract:hardware -abstract:synthesize -abstract:circuit -"natural language" -CAD -"memory optimization" -processors -microprocessors -java -"human computer interaction" -memory -kernel -gcc -nlp -holonic -"hypermedia systems" -olap -networking -"Programming Environments" -"Software process models"
+abstract:software +abstract:"high level model" abstract:evaluate abstract:assurance abstract:review abstract:inspection abstract:verification abstract:analysis -abstract:hardware -abstract:synthesize -abstract:circuit -"natural language" -CAD -"memory optimization" -processors -microprocessors -java -"human computer interaction" -memory -kernel -gcc -nlp -holonic -"hypermedia systems" -olap -networking -"Programming Environments" -"Software process models"

Fontes selecionadas após avaliação: a priori, todas as listadas satisfazem os critérios de qualidade.

Checagem de referências: Todas foram aprovadas

A.3 – Seleção dos estudos

A.3.1 – Definição dos estudos

Critérios de inclusão e exclusão dos estudos: Os artigos devem: estar disponíveis; descrever ou apresentar algum método que avalie a arquitetura de um software ou de seus modelos;

Definição dos tipos de estudos: Todos os estudos relacionados ao tópico de pesquisa serão selecionados.

Procedimentos de seleção dos estudos: As strings de buscas são executadas nas fontes de pesquisa selecionadas. Após isso, para a seleção de um conjunto inicial de estudos, os critérios de inclusão e exclusão são aplicados através das informações obtidas com a leitura dos abstracts de cada estudo. Para os estudos restantes, eles são lidos na integra com o objetivo de extrais as informações desejadas.

A.3.2 – Execução da seleção

Lista dos estudos selecionados: a lista pode ser encontrada no capítulo 3 dessa dissertação.

A.4 – Extração das informações

A.4.1 – Critérios de inclusão e exclusão de informações

Permita descrever uma abordagem de avaliação arquitetural sem a necessidade de consultar o estudo que a originou

A.4.2 – Formulário de extração de dados

Os dados a serem extraídos devem possibilitar a caracterização da abordagem através do *framework* descrito no capítulo 3 dessa dissertação

A.5 – Sumarização dos resultados

Devido à grande quantidade de informações coletadas não foi possível fazer um sumário das abordagens identificadas. O capítulo 3 dessa dissertação descreve os resultados encontrados ao executar esse protocolo.

Apêndice B – *Checklist* para inspeção arquitetural

Este apêndice descreve as diferentes versões do checklist utilizados por ArqCheck desde a sua criação até a última evolução, realizada após o estudo de viabiilidade.

B.1 – Versão do *checklist* utilizado no estudo de viabilidade (versão 1.0)

Checklist para Inspeção de Documento Arquitetural

⇒ **Instruções**

- Avalie a documentação arquitetural através dos itens de avaliação abaixo;
- A opção NA (Não Aplicável) deve ser marcada se considerar que não é possível aplicar o item para avaliar a documentação arquitetural;
- Para os itens de avaliação dos requisitos de qualidade (21 - 42), a documentação arquitetural deve ser avaliada em relação aos requisitos de qualidade identificados na "Guia de identificação de contexto". Caso os requisitos não tiverem sido identificados, o inspetor deve avaliar somente se o item é aplicável ou não.

Equipe/Revisor #: _____

Nº	Itens de avaliação da consistência das representações entre os diagramas	Sim	Não	NA
1	Nos diagramas, existe algum módulo/cluster que não possui relacionamentos, ficando isolado dos demais?			
2	Todos elementos arquiteturais, identificados através de seu nome, foram representados através da mesma abstração nos diferentes diagramas?			
	Itens de avaliação da consistência das representações entre os diagramas (específicos à abordagem de documentação arquitetural utilizada)			
Nº	**Visão Modular**	**Sim**	**Não**	**NA**
3	Os módulos internos de cada cluster foram descritos em algum diagrama da visão Modular?			
4	Todo relacionamento definido com um cluster foi devidamente mapeado para um de seus módulos internos?			
Nº	**Visão Dinâmica**	**Sim**	**Não**	**NA**
5	Toda porta/interface possui um nome, é utilizada com um único propósito e de forma única?			
6	Os fluxos de execução, descritos na visão Dinâmica, alocam todos os módulos definidos na visão Modular?			
7	Todo módulo/cluster representado na visão Dinâmica foi descrito na visão Modular?			

8	Todo fluxo entre dois elementos arquiteturais pode ser mapeado para algum relacionamento da visão Modular?			
9	Todo relacionamento, descrito na visão Modular, pode ser mapeado para algum fluxo de comunicação, de dados ou de controle da visão Dinâmica?			
Nº	**Visão de Alocação**	**Sim**	**Não**	**NA**
10	Todo módulo/cluster, representado na visão de Alocação, foi descrito na visão Modular?			
11	Toda dependência, representada na visão de Alocação, pode ser mapeada para um ou mais relacionamentos da visão Modular?			
12	Dado os módulos/clusters representados na visão de Alocação, todos os relacionamentos definidos entre eles na visão Modular também foram representados na visão de Alocação?			
Nº	**Visão de Contexto Geral**	**Sim**	**Não**	**NA**
13	Todo módulo/cluster representado na visão de Contexto Geral foi descrito na visão Modular?			
14	Todo relacionamento entre elementos arquiteturais pode ser mapeado para os fluxos de comunicação, descritos na visão Dinâmica?			
Nº	**Itens de avaliação do atendimento aos requisitos**	**Sim**	**Não**	**NA**
15	Todo elemento arquitetural tem a sua presença na arquitetura justificada por um conjunto de requisitos?			
16	Todo requisito funcional ou de qualidade ou adicional, criado pelas decisões arquiteturais, foi atendido por algum elemento arquitetural?			
17	As responsabilidades atribuídas a um elemento arquitetural estão relacionadas aos requisitos que ele atende?			
Nº	**Itens de avaliação do atendimento aos requisitos**	**Sim**	**Não**	**NA**
18	Nos diagramas da Visão Dinâmica, todo Fluxo de Execução é justificado por um conjunto de requisitos?			
19	Nos diagramas da Visão Dinâmica, todos os requisitos que justificam um Fluxo de Execução são atendidos pelos elementos arquiteturais que compõe esse fluxo?			
20	Nos diagramas da Visão de Contexto Geral, os relacionamentos entre os elementos externos e os elementos arquiteturais foram justificados por um conjunto de requisitos?			
Itens de avaliação da abordagem de atendimento aos requisitos de qualidade				
Nº	**Itens relacionados a Desempenho**	**Sim**	**Não**	**NA**
Guia de identificação de contexto: De acordo com os requisitos de desempenho que se deseja avaliar, identificar os fluxos de execução do sistema (Fluxo) relacionados ao atendimento dos requisitos selecionados e os elementos que participam desses fluxos (Elementos Participantes). Essa identificação pode ser feita através da análise do estímulo, do artefato e da resposta descritos nos requisitos selecionados.				
21	Todos os *Elementos Participantes* foram alocados em um mesmo nó computacional visando aumentar o desempenho do *Fluxo*?			
22	Quando possível, os fluxos de comunicação que compõem o *Fluxo* são executados concorrentemente?			

23	Nesse contexto, elementos intermediários (ex: máquina virtual, servidor de nomes, repositórios) são elementos arquiteturais cuja responsabilidade é realizar a comunicação entre dois *Elementos Participantes*. Existe algum *Elemento Relacionado* que executa o papel de elemento intermediário dentro do *Fluxo*?			
24	Todo *Elemento Participante* pode ter a comunicação com outros *Elementos Participantes* ou a freqüência de execução das suas funcionalidades otimizada?			
25	Para todo *Elemento Participante*, a forma como seus serviços são requisitados pode ser otimizada?			
Nº	**Itens relacionados a Disponibilidade**	**Sim**	**Não**	**NA**
***Guia de identificação de contexto*:** De acordo com os requisitos de disponibilidade selecionados, identifique os elementos arquiteturais que devem possuir alta disponibilidade (*Elemento Principal*). A identificação pode ser feita a partir do *artefato* e da *resposta* descritos nos cenários de disponibilidade				
26	Existe algum elemento arquitetural responsável por verificar periodicamente a disponibilidade dos *Elementos Principais*?			
27	Todo *Elemento Principal* possui redundâncias e algum elemento arquitetural que gerencia a sua disponibilidade?			
28	Todo *Elemento Principal* e suas respectivas redundâncias possuem alguma forma de realizar a sincronização de seus dados e possíveis estados?			
Nº	**Itens relacionados a Modificabilidade**	**Sim**	**Não**	**NA**

Requisito RNF3

A infra-estrutura deve ser construída de forma a permitir que os desenvolvedores modifiquem a abordagem de recuperação e persistência dos artefatos manuseados, sem que seja necessário realizar alterações em outras funcionalidades. Essa alteração pode ocorrer durante o desenvolvimento do sistema ou durante a sua manutenção.

Cenário de Modificabilidade	
Fonte do estímulo	Desenvolvedores
Estímulo	Modificação da abordagem de manipulação de dados
Contexto do sistema	Desenvolvimento do sistema / Manutenção do sistema
Artefato	Repositórios de Dados
Resposta	O sistema deve manipular os dados como antes da modificação
Medida de resposta	---

Guia de identificação de contexto: De acordo com os requisitos de modificabilidade que se deseja avaliar, identificar os elementos arquiteturais que podem ser modificados (Elemento Modificável) e também os elementos que possuem relacionamentos com os Elementos Modificáveis de forma direta (Elemento Relacionado). A identificação dos Elementos Modificáveis é feita principalmente através da análise do estimulo e contexto dos requisitos selecionados.				
29	As responsabilidades de um Elemento Relacionado pertencem a um mesmo contexto, ou seja, visam atingir um mesmo propósito, manipulam um mesmo tipo de dado ou são utilizadas em um mesmo fluxo de execução?			
30	As funcionalidades dos Elementos Relacionados possuem uma similaridade que, para reduzir os esforços de modificação, poderiam ser agrupadas em um único elemento?			
31	Todo Elemento Modificável disponibiliza interfaces para realizar os fluxos de comunicação?			
32	A comunicação entre um Elemento Relacionado e um Elemento Modificável, e as funcionalidades disponibilizadas pelo conector (interface/porta) que intermedia essa comunicação são realmente necessárias?			
33	Nesse contexto, elementos intermediários (ex: máquina virtual, servidor de nomes, repositórios) são elementos arquiteturais cuja responsabilidade é permitir que outros elementos acessem as funcionalidades ou dados do Elemento Modificável. Existe			

	algum Elemento Relacionado que realize o papel de elemento intermediário?			
Nº	**Itens relacionados a Segurança**	**Sim**	**Não**	**NA**

Requisito RNF4

A infra-estrutura deve permitir, durante a execução de um processo instanciado, que o Usuário Executor tenha acesso somente às atividades relacionadas aos papéis que lhe foram alocados pelo Usuário Engenheiro.

Cenário de Segurança	
Fonte do estímulo	Usuário Executor
Estímulo	Acesso a funcionalidades
Contexto do sistema	Durante a execução normal da infra-estrutura
Artefato	Ambiente para Execução
Resposta	Acesso somente às funcionalidades alocadas pelo Usuário Engenheiro ao Usuário Executor
Medida de resposta	---

Guia de identificação de contexto: De acordo com os requisitos de segurança que se deseja avaliar, três tipos de elementos devem ser identificados: os elementos arquiteturais seguros (Elemento Seguro) que são acessados de forma restrita, os elementos que possuem relacionamentos com os Elementos Seguros (Elemento Relacionado) e os elementos que buscam garantir o acesso seguro ao Elemento Seguro (Elemento Gerenciador). Para que o Elemento Seguro seja identificado, o estímulo e o artefato dos requisitos selecionados devem ser analisados.

34	Todo *Elemento Seguro* implementa alguma funcionalidade ou possui algum relacionamento com um *Elemento Gerenciador* que autentique o acesso aos seus dados ou funcionalidades?			
35	Todo *Elemento Relacionado* realiza alguma autenticação para acessar as funcionalidades de *Elemento Seguro*?			
36	Todo *Elemento Seguro* disponibiliza somente os serviços relacionados aos direitos de acesso do *Elemento Relacionado* requisitante?			
37	Todo *Elemento Seguro* e os *Elementos Relacionados* possuem funcionalidades que permitem a codificação e a decodificação dos dados trocados entre eles?			
38	Todo *Elemento Seguro* e os *Elementos Relacionados* possuem funcionalidades que permitem checar a integridade dos dados?			
Nº	**Itens relacionados a Testabilidade**	**Sim**	**Não**	**NA**

***Guia de identificação de contexto*:** De acordo com os requisitos de testabilidade que se deseja avaliar, identificar os elementos que participam do atendimento às funcionalidades que se deseja permitir testabilidade (*Elemento Testável*). Essa identificação pode ser feita através de uma análise do *estímulo* e *artefato* dos requisitos selecionados.

39	Todo *Elemento Testável* disponibiliza interfaces de comunicação que permitem a sua substituição?			
40	Todo *Elemento Testável* possui interfaces específicas ou funcionalidades que visam à sua monitoração e à realização de testes?			

<table>
<tr><th>Nº</th><th>Itens relacionados a Usabilidade</th><th>Sim</th><th>Não</th><th>NA</th></tr>
<tr><td colspan="5">RequisitoRNF2

A infra-estrutura deve permitir que a execução de um processo ocorra de forma desacoplada das atividades de configuração e instanciação de modelos.

<table>
<tr><th colspan="2">Cenário de Usabilidade</th></tr>
<tr><td>Fonte do estímulo</td><td>Usuário Engenheiro</td></tr>
<tr><td>Estímulo</td><td>Modelar ou Instanciar processos</td></tr>
<tr><td>Contexto do sistema</td><td>Durante a execução normal da infra-estrutura</td></tr>
<tr><td>Artefato</td><td>Ambientes para Instanciação e Configuração</td></tr>
<tr><td>Resposta</td><td>Novos modelos ou ambientes instanciados</td></tr>
<tr><td>Medida de resposta</td><td>Não afeta a execução de ambientes previamente instanciados</td></tr>
</table>
</td></tr>
<tr><td colspan="5">Guia de identificação de contexto: Para atender a requisitos de usabilidade, funcionalidades ou elementos arquiteturais podem ser definidos ou adaptados. Portanto, de acordo com os requisitos de usabilidade que se deseja avaliar, caso seja necessário, essas funcionalidades (Funcionalidades Envolvidas) devem ser identificadas. Além disso, devem-se identificar quais elementos arquiteturais serão adaptados ou criados para atender diretamente ao Requisito ou às Funcionalidades Adicionais (Elementos Relacionados). A identificação das funcionalidades e dos elementos arquiteturais pode ser feita através da análise do estímulo, da resposta e da medida de resposta descritos nos requisitos.</td></tr>
<tr><td>41</td><td>Toda Funcionalidade Envolvida, caso tenha sido definida, está associada a algum elemento arquitetural?</td><td></td><td></td><td></td></tr>
<tr><td>42</td><td>Os Elementos Relacionados possuem responsabilidades ou foram organizados visando atender ao requisito avaliado?</td><td></td><td></td><td></td></tr>
</table>

B.2 – Versão do *checklist* utilizado no estudo de observação (versão 2.1)

Instruções para utilização do Checklist para Inspeção de Documento Arquitetural

⇒ Instruções

- Avalie a documentação arquitetural através dos itens de avaliação que compõem o *checklist* apresentado na próxima página;
- Antes de utilizar um item que objetiva avaliar a consistência do documento arquitetural, identifique a que tipo de visão ele se aplica;
- Para avaliar o atendimento aos requisitos de qualidade, utilize as "*Guias de identificação de contexto*". Essas guias permitem identificar que elementos arquiteturais devem ser avaliados para se verificar o atendimento aos requisitos de qualidade;
- Ao avaliar uma visão arquitetural ou as funcionalidades de um elemento arquitetural, leia a descrição textual que compõem o documento e não somente os diagramas gráficos;
- Caso a sua resposta seja diferente da Resposta Esperada – RE (Não – N, Sim – S), um defeito deve ser identificado no Relatório de Discrepâncias;
- Caso haja dúvida em relação aos termos utilizados nos itens de avaliação, utilize o glossário abaixo;

⇒ Glossário com os termos utilizados no *checklist*

- **Documento arquitetural**: Documento que descreve a arquitetura de um software através do uso de visões arquiteturais. Cada visão pode ser composta por diagramas que representam os elementos arquiteturais sob determinada perspectiva. Esse documento é composto também por uma descrição textual, que detalha o papel de cada elemento representado nos diferentes diagramas e identifica a rastreabilidade dos elementos arquiteturais com os requisitos especificados;
- **Elemento arquitetural**: Elemento básico da arquitetura responsável por implementar determinadas funcionalidades;
- **Visão Modular**: Perspectiva utilizada para representar como os elementos que compõem a arquitetura estão organizados;
- **Visão de Contexto Geral**: Essa perspectiva tem como objetivo representar uma visão geral dos principais componentes que formam a arquitetura do software e de como ela se relaciona com os elementos externos ao seu contexto (atores e sistemas externos).
- **Visão Dinâmica**: Esta perspectiva procura descrever o comportamento dos elementos arquiteturais durante a realização das diferentes seqüências de execução presentes no sistema;
- **Visão de Alocação**: Esta perspectiva busca representar o mapeamento das unidades de software para elementos físicos do ambiente (hardware, arquivos do sistema);
- **Seqüência de Execução**: Conjunto de fluxos de mensagens entre os elementos arquiteturais que determina o comportamento esperado do sistema quando se deseja realizar uma determinada tarefa. Em UML, esse comportamento pode ser representado através de diagramas de seqüência;

[Termo úteis somente para representação de "caixa e setas"]

- **Módulo/Módulo Interno**: Elemento pertencente ao mais baixo nível de abstração e cuja representação dos elementos internos não é relevante para o entendimento da solução. Esse elemento arquitetural realiza diversas responsabilidades e se relaciona com outros elementos visando atender a um conjunto de requisitos;
- **Cluster**: Elemento arquitetural utilizado para agrupar módulos que pertencem a um mesmo contexto, ou seja, que possuem responsabilidades similares ou que lidam com um mesmo tipo de dados. Esse elemento é composto por módulos internos cuja representação é essencial para o entendimento da solução;
- **Dependência**: Tipo de relacionamento entre dois elementos arquiteturais (módulo e cluster) que indica algum tipo de dependência entre eles;
- **Porta**: Uma porta é utilizada para representar as propriedades de um elemento que devem ser visíveis aos demais. Uma porta é formada pelas seguintes informações: (1) o nome que a identifica, (2) o tipo de comunicação que ela realiza, classificados em três tipos: fluxo de dados, onde somente dados trafegam, fluxo de controle, por onde são feitas exclusivamente invocações para funcionalidades presentes em outro elemento, e fluxo de comunicação, que permite o fluxo dos outros dois tipos, (3) a forma como a comunicação é realizada: na recuperação dados/recebimento controle, ou persistência dados/invocação de serviços ou ambos e (4) a descrição das funcionalidades que a compõem.
- **Interface**: Especialização de uma Porta. A principal diferença entre uma Interface e uma Porta está no fato da Interface ser desassociada do Módulo. Com isso, caso o módulo que a disponibiliza seja alterado, as modificações dificilmente serão notadas pelos módulos que implementam essa interface

Checklist para Inspeção de Documento Arquitetural

Equipe/Revisor #: _____

Itens de avaliação da consistência do documento				
Nº	**Todas as visões**	**Sim**	**Não**	**RE**
1	Ao analisar todos os diagramas, foi identificado algum elemento arquitetural que não possua relacionamentos, ficando isolado dos demais?			N
2	Ao analisar todos os diagramas, foi identificado algum elemento arquitetural que não tenha sido representado através da mesma abstração em diferentes diagramas?			N
3	A descrição textual que compõem o documento está de acordo com o que foi representado nos diferentes diagramas gráficos?			S
Itens de avaliação da consistência do documento				
Nº	**Visão Modular**	**Sim**	**Não**	**RE**
4	Para todo componente, foi identificado em algum diagrama da visão Modular as classes ou sub-componentes que o compõem?			S
5	Todo relacionamento definido entre dois componentes pode ser mapeado para relacionamentos realizados entre classes/sub-componentes que o compõem?			S
6	Todo relacionamento realizado entre classes/sub-componentes alocados em "componentes pais" distintos foi definido como uma dependência entre esses "componentes pais" em algum outro diagrama?			S
7	Toda interface é disponibilizada por um único componente/classe?			S
Nº	**Visão de Contexto Geral**	**Sim**	**Não**	**RE**
8	Todo relacionamento entre os componentes, descrito na visão de Contexto Geral, pode ser mapeado para os relacionamentos descritos na visão Modular?			S
9	Todo elemento externo que se relaciona diretamente com um componente da arquitetura foi representado na visão Modular?			S
10	Todo relacionamento existente entre um elemento externo e um componente da arquitetura foi mapeado para alguma classe representada na visão Modular?			S
11	Toda interface disponibilizada/implementada pelos componentes descritos na visão de Contexto Geral podem ser mapeadas para uma classe que a disponibiliza/implementa na visão Modular?			S
Nº	**Visão Dinâmica**	**Sim**	**Não**	**RE**
12	Ao analisar as seqüências de execução, descritas na visão Dinâmica, pode-se dizer que todas as classes definidas na visão Modular foram alocadas em ao menos uma dessas seqüências?			S
13	Toda mensagem trocada entre duas classes/componentes, descritas na visão Dinâmica, pode ser mapeada para algum relacionamento definido entre essas classes/componentes da visão Modular?			S
14	Todo relacionamento, descrito na visão Modular, pode ser mapeado para alguma mensagem de comunicação descrita na visão Dinâmica?			S

15	Todo componente externo representado na visão Dinâmica foi representado em algum diagrama da visão Modular?			S
Nº	**Visão de Alocação**	**Sim**	**Não**	**RE**
16	Toda dependência, representada na visão de Alocação, pode ser mapeada para um ou mais relacionamentos da visão Modular?			S
17	Dado as classes/componentes representados na visão de Alocação, todos os relacionamentos definidos entre eles na visão Modular também foram representados na visão de Alocação?			S
Nº	**Itens de avaliação do atendimento aos requisitos**	**Sim**	**Não**	**RE**
18	Todo elemento arquitetural tem a sua presença na arquitetura justificada por um conjunto de requisitos?			S
19	As responsabilidades dos elementos arquiteturais estão condizentes com os requisitos que eles atendem?			S
Itens de avaliação da abordagem de atendimento aos requisitos de qualidade				
Nº	**Itens relacionados a Modificabilidade**	**Sim**	**Não**	**RE**

Requisito 26

The architectural design of the component must be prepared for the use of exchangeable GUI's (Graphical User Interface)

Cenário de Modificabilidade	
Fonte do estímulo	Usuário
Estímulo	Mudança da GUI do Menu
Contexto do sistema	Durante a execução do sistema
Artefato	GUI do menu
Resposta	O componente deve modificar a GUI do Menu
Medida de resposta	---

Guia de identificação de contexto: De acordo com os requisitos de modificabilidade selecionados, identifique os elementos arquiteturais que podem ser modificados (*Elemento Modificável*) e também os elementos que se relacionam com esses *Elementos Modificáveis* de forma direta (*Elemento Relacionado*). A identificação dos *Elementos Modificáveis* é feita através da análise do *estímulo,* do *contexto e do artefato* descritos nos cenários de modificabilidade.

20	As responsabilidades especificadas para cada *Elemento Relacionado* pertencem a um mesmo contexto, ou seja, visam atingir um mesmo propósito, manipulam um mesmo tipo de dado ou são utilizadas em uma mesma seqüência de execução?			S
21	*Elementos Relacionados* possuem responsabilidades comuns ou similares que podem ser alocadas em um único elemento?			N
22	Todo *Elemento Modificável* disponibiliza interfaces para se comunicar com os demais elementos?			S
23	A comunicação entre um *Elemento Relacionado* e um *Elemento Modificável,* e as funcionalidades disponibilizadas pelo conector (interface/porta) que intermedia essa comunicação são realmente necessárias?			S
24	Elementos intermediários (ex: máquina virtual, servidor de nomes, repositórios) são utilizados como mediadores de comunicação entre o *Elemento Modificável* e os demais elementos arquiteturais, diminuindo o impacto das alterações realizadas nos *Elementos Modificáveis*. Existe algum *Elemento Relacionado* que realize esse papel de intermediário?			S

Apêndice C – Relatório de discrepâncias

Grupo de Engenharia de Software Experimental

Relatório de Discrepâncias

Equipe/Revisor #: _____

Tempo de inspeção (em minutos):_____

⇒ **Instruções**

- Os Diagramas envolvidos devem ser identificados através do seu nome e do nome da visão que eles pertencem (Visão::Diagrama)
- Os tipos de defeito podem ser: omissão, ambigüidade, inconsistência, informação estranha ou fato incorreto.

Tipos de defeitos	Descrição
Omissão	1. Quando um elemento arquitetural necessário para o atendimento a um requisito não foi definido;
Ambigüidade	2. Quando a forma como os elementos arquiteturais ou suas responsabilidades foram definidos dificulta ou impossibilita o atendimento a um requisito de qualidade. 3. Quando elementos descritos em visões distintas possuem o mesmo nome, mas responsabilidades diferentes (Homônimo);
Inconsistência	4. Quando elementos descritos em visões distintas possuem mesma responsabilidade, mas nomes distintos (Sinônimo); 5. Quando um elemento arquitetural presente em diagramas das demais visões não foi definido no diagrama avaliado; 6. Quando a representação não condiz com a semântica estabelecida pela abordagem de documentação. 7. Quando um elemento arquitetural é definido com responsabilidades distintas em duas ou mais visões. 8. Quando um elemento é representado de maneira diferente em duas visões.
Fato Incorreto	9. Quando um elemento não foi descrito ou representado de forma correta 10. Quando não é possível mapear um elemento arquitetural para algum elemento descrito em outra visão.
Informação Estranha	11. Quando não é possível determinar o papel de um elemento arquitetural ou de uma de suas responsabilidades no atendimento aos requisitos especificados.

Nº	Item do *checklist*	Diagramas envolvidos	Descrição da Discrepância	Tipo de Defeito
1				
2				

Apêndice D – Plano do Experimento

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

PROGRAMA DE ENGENHARIA DE SISTEMAS E COMPUTAÇÃO

ENGENHARIA DE SOFTWARE EXPERIMENTAL

1. IDENTIFICAÇÃO

Título: Estudo de caso / Abordagem para inspeção de documentos arquiteturais baseada em *checklist* (ArqCheck)

Tema: Avaliação de documentos arquiteturais

Área Técnica: Arquitetura de Software / Qualidade de Software

Autores: Rafael Ferreira Barcelos barcelos@cos.ufrj.br

Guilherme Horta Travassos ght@cos.ufrj.br

Afiliação: ESE - Engenharia de Software Experimental

COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação

Universidade Federal do Rio de Janeiro, Cx. Postal 68.511, CEP 21945-970,

Rio de Janeiro – RJ – Brasil.

Local: COPPE/UFRJ

Versão: 2.1

Data: 19/02/2006

2. CARACTERIZAÇÃO

2.1. Tipo

Estudo de observação: Este tipo de estudo ocorre em um ambiente de estudo, onde indivíduos desempenham alguma tarefa enquanto são observados pelos pesquisadores. O propósito deste estudo é coletar dados sobre como uma tarefa é executada. Desta forma, os pesquisadores podem adquirir uma compreensão mais

refinada sobre como a nova tecnologia é aplicada na prática, presenciando eventuais dificuldades que os indivíduos podem enfrentar (SILVA, 2004).

No contexto dessa pesquisa, tratamos esse estudo de observação como um pré-experimento, ou seja, a sua execução consistiu na aplicação de um tratamento a somente um caso, onde uma comparação é realizada em relação a um determinado baseline (AMARAL, 2003). Com isso, aplicamos a abordagem proposta para inspecionar um documento arquitetural que já foi avaliado através de uma outra abordagem.

2.2. Domínio

Engenharia de Software.

2.3. Língua

O estudo será conduzido tendo o *checklist* que compõem a abordagem de inspeção proposta descrito em português e o documento arquitetural avaliado descrito em inglês.

2.4. Parceiros

O estudo será conduzido no contexto do grupo de Engenharia de Software Experimental (ESE) da COPPE/UFRJ.

Além disso, foi estabelecido um acordo com a empresa SIEMENS-AM, onde eles se comprometem em disponibilizar o documento arquitetural de um de seus projetos e os respectivos defeitos encontrados através da execução de uma abordagem de inspeção própria. Em contrapartida, os experimentalistas se comprometem em fornecer um feedback em relação aos resultados obtidos nesse estudo.

2.5. Links

- Programa de Engenharia de Sistemas e Computação da COPPE/UFRJ
- http://www.cos.ufrj.br
- Equipe de Engenharia de Software Experimental da COPPE/UFRJ
- http://www.cos.ufrj.br/~ese
- Instituto de Pesquisa da SIEMENS-AM
- http://www.siemens.com.br

2.6. Expectativa de Execução

Com base nos resultados do estudo de viabilidade já realizado, identificamos que a abordagem proposta (ARQCHECK) possibilita a identificação de defeitos em um

documento arquitetural. Com isso, no estudo proposto, procuramos identificar agora a eficácia/eficiência do *checklist* que compõem a abordagem ARQCHECK em avaliar um documento arquitetural.

Por eficácia, entendemos a quantidade de defeitos em relação ao número total de discrepâncias encontradas. Por eficiência, entendemos a quantidade de defeitos identificados por unidade de tempo de inspeção.

Para isso, iremos aplicar ARQCHECK a um documento arquitetural que já foi avaliado por uma abordagem ad-hoc e que identificou defeitos que permitiram a melhoria da sua qualidade. Esse documento foi criado e avaliado no contexto de um projeto real de desenvolvimento de software da SIEMENS-AM.

Sendo assim, pretendemos observar a inspeção de um documento arquitetural real através do *checklist* proposto e comparar através da realização desse estudo os resultados obtidos pelas duas abordagens, visando caracterizar a nossa abordagem em relação aos resultados obtidos pela SIEMENS-AM. Para isso, utilizaremos os dados coletados durante esse estudo e os dados fornecidos pela SIEMENS-AM para obter o custo/eficiência e eficácia das duas abordagens.

Entretanto, a comparação entre as duas abordagens não é o nosso objetivo principal, ou seja, não visamos identificar qual a melhor abordagem de avaliação através dessa comparação. Isso ocorre visto que, além dos resultados obtidos com a aplicação dessas abordagens em somente um caso não serem suficientes para tirarmos esse tipo de conclusão, não temos acesso a algumas informações que permitem caracterizar a inspeção realizada pela SIEMENS-AM.

Portanto, o objetivo dessa comparação é de caracterizar ARQCHECK em relação a um controle, ou baseline. Para esse estudo, o baseline escolhido foi a abordagem SIEMENS, pois, mesmo sendo uma abordagem ad-hoc, é a abordagem que de fato está sendo utilizada nesse contexto industrial e seus resultados já foram utilizados para a melhoria da qualidade da arquitetura avaliada.

Com isso, neste estudo de observação buscamos criar um corpo de conhecimento que nos permita avaliar a aplicação da abordagem proposta. Além disso, esperamos que os resultado obtidos, e o corpo de conhecimento construído decorrente de sua condução, nos forneça subsídios que permitam evoluir a abordagem para otimizar a sua aplicação tanto em relação ao esforço necessário para executá-la quanto à quantidade e tipo de defeitos que ela permite identificar.

2.7. Número Estimado de Repetições

A princípio, o estudo de viabilidade será conduzido uma única vez. Caso os resultados do estudo apontem para o sucesso da abordagem ARQCHECK, pretendemos

continuar na condução de estudos, seguindo a metodologia de (SHULL *et al.*, 2001), visando a transferência dessa tecnologia apara a industria. Caso contrário, pretendemos rever as premissas que sustentam a abordagem e, com base nos resultados obtidos por esse estudo, propor melhorias que permitam sua evolução.

2.8. Glossário de Termos

- Defeito: propriedade de um artefato que o impeça de satisfazer seus requisitos de qualidade (LAITENBERGER e ATKINSON, 1999).
- Discrepância: potenciais defeitos detectados por revisores de software durante uma sessão de inspeção de software.
- Utilidade da abordagem de inspeção: capacidade da abordagem em identificar defeitos arquiteturais que permitam a melhoria da qualidade do documento e que justifiquem o custo/benefício dessa avaliação. Uma forma de caracterizar essa utilidade é através da avaliação da sua eficácia e seu custo/eficiência.
- Falso-positivo: discrepância identificada pelo revisor durante uma sessão de inspeção que, quando avaliada posteriormente, não representou de fato um defeito real no documento arquitetural.

3. INTRODUÇÃO

Devido à importância da arquitetura de software no processo de desenvolvimento, a avaliação dos documentos arquiteturais passa a ter extrema importância para os stakeholders.

A avaliação arquitetural consiste em caracterizar e avaliar os documentos arquiteturais através de métodos ou procedimento sistemáticos. Essa avaliação verifica principalmente se a arquitetura descrita através do documento arquitetural atende aos requisitos especificados pelo cliente. Visto que são os requisitos de qualidade os que mais influenciam a construção de uma arquitetura, portanto, é principalmente sob a perspectiva dos atributos de qualidade que a avaliação é realizada.

Devido à importância dessa atividade para a melhoria da qualidade do produto de software e para o sucesso do projeto, os experimentalistas optaram por definir uma abordagem para inspeção baseada em *checklist* (ARQCHECK) visando identificar defeitos arquiteturais em relação aos requisitos que foram especificados pelo cliente.

Após a construção desse *checklist*, os experimentalistas decidiram realizar alguns estudos de avaliação visando caracterizar essa nova abordagem em relação à sua

capacidade em identificar defeitos e visando permitir a transferência dessa tecnologia para o contexto industrial.

Para isso, a metodologia definida por SHULL et al. (2001) está sendo utilizada. Portanto, seguindo essa metodologia, um primeiro estudo de viabilidade já foi realizado com o principal objetivo de caracterizar os itens de avaliação que formam o *checklist* de ARQCHECK em relação à sua clareza e aplicabilidade na identificação de defeitos.

O estudo ao qual esse plano se refere é um estudo de observação com o objetivo de caracterizar a utilidade da abordagem ARQCHECK em identificar defeitos em documentos arquiteturais.

4. DEFINIÇÃO DO ESTUDO EXPERIMENTAL

4.1. Objeto de Estudo

A abordagem para inspeção de documentos arquiteturais baseadas em *checklist* (ARQCHECK) é composta pelos seguintes elementos:

- Atividades de configuração utilizadas para adequar o *checklist* ao contexto da arquitetura a ser avaliada;
- Atividades de execução que devem ser seguidas pelos inspetores para identificar defeitos em um documento arquitetural utilizando como principal ferramenta um *checklist*;
- *Checklist* de avaliação composto por itens que buscam avaliar os documentos sob diferentes perspectivas.
- A taxonomia de defeitos que visa classificar as discrepâncias que podem ser identificadas em um documento arquitetural através da ARQCHECK;

No contexto deste estudo, procuramos avaliar o *checklist* de avaliação, já configurado às características da arquitetura, e a taxonomia de defeitos utilizada.

4.2. Propósito

O propósito desse estudo é realizar uma avaliação da eficiência e eficácia da abordagem ARQCHECK em relação à identificação de defeitos.

4.3. Foco da Qualidade

A avaliação experimental da aplicação da abordagem ARQCHECK em inspeções de documentos arquiteturais tem seu foco de qualidade:

- Na avaliação quantitativa da eficácia da abordagem ARQCHECK em relação à detecção de defeitos;

- Na avaliação quantitativa da eficiência da abordagem ARQCHECK em relação à detecção de defeitos;
- Na avaliação quantitativa do tipo de defeitos encontrados;
- Na avaliação qualitativa da usabilidade da abordagem, ou seja, o quão bem os participantes do estudo acreditam que a aplicação da abordagem ARQCHECK os *auxiliaram na tarefa de detecção de defeitos.*

4.4. Perspectiva

A perspectiva é do ponto de vista dos experimentalistas, que desejam avaliar a aplicação prática da abordagem ARQCHECK.

4.5. Contexto

O estudo experimental de observação será realizado em ambiente acadêmico utilizando estudantes de pós-graduação em Engenharia de Software como participantes. O estudo consistirá na simulação de uma sessão de inspeção de software, onde os participantes aplicarão a abordagem ARQCHECK em um documento arquitetural. Portanto, a condução do estudo caracteriza-se por ser off-line.

O documento arquitetural consiste em um modelo real, criado no contexto de um projeto da SIEMENS-AM. *Esse artefato já foi avaliado através de uma abordagem* própria. Sabe-se somente que esse documento possui defeitos contudo não se tem idéia a princípio de que defeitos são esses ou de que tipos são. Esses dados será fornecidos pela SIEMENS-AM somente após a execução do estudo, quando iniciarmos a análise dos dados.

O estudo experimental trata de um problema real no domínio da Engenharia de Software, que é a falta de apoio adequado aos revisores durante a condução da atividade de detecção de defeitos em inspeções de um documento arquitetural. Mesmo utilizando um documento arquitetural criado em um contexto industrial, não será possível generalizar os resultados para o contexto de projetos reais da indústria devida principalmente às ameaças de validade identificadas na seção 5.8.

4.6. Objetivos Específicos

Analisar	*a aplicação da abordagem ARQCHECK*
com o propósito de	caracterizá-la
em relação	eficácia em identificar defeitos em documentos arquiteturais
do ponto de vista	Experimentalistas
no contexto	*da inspeção de um documento arquitetural que fora criado em um* contexto industrial

Analisar	a aplicação da abordagem ARQCHECK
com o propósito de	caracterizá-la
em relação	Custo/eficiência em identificar defeitos em documentos arquiteturais
do ponto de vista	Experimentalistas
no contexto	da inspeção de um documento arquitetural que fora criado em um contexto industrial

4.7. Questões e Métricas

As questões abaixo se referem à aplicação da abordagem ARQCHECK em sessões de inspeção de documentos arquiteturais. A saber:

- Questões referentes à investigação se a abordagem ARQCHECK atinge o objetivo ao qual foi proposta (apoiar a detecção de defeitos em documentos arquiteturais).

Q1: Qual é a eficácia da abordagem ARQCHECK no que diz respeito à detecção de defeitos?

Métricas:

- Quantidade consolidada de defeitos detectados por ARQCHECK
- Quantidade consolidada de defeitos detectados por ARQCHECK e pela SIEMENS
- Quantidade de defeitos encontrados pela SIEMENS.
- Eficácia da abordagem = Quantidade de defeitos detectados por ArqCheck / Quantidade de discrepâncias identificadas.
- Eficácia 2 da abordagem = Quantidade de defeitos detectados por ArqCheck e comuns a abordagem SIEMENS/ Quantidade de discrepâncias identificadas pela abordagem SIEMENS.

Q2: Qual é o custo/eficiência da aplicação de ARQCHECK no que diz respeito à detecção de defeitos?

Métricas:

- Custo/Eficiência da abordagem (ArqCheck) = Quantidade de defeitos detectados por participante / Tempo de inspeção do participante
- Custo/Eficiência da abordagem (SIEMENS) = Quantidade de defeitos detectados / Tempo total de inspeção

- Questões envolvendo a análise da usabilidade da abordagem ARQCHECK.

Q3: A aplicação de ARQCHECK é útil na tarefa de detecção de defeitos?

Métrica: avaliação qualitativa do participante.

Q4: Qual é o grau de dificuldade na aplicação de ARQCHECK?

Métrica: avaliação qualitativa do participante.

Obs.: as avaliações qualitativas dos participantes sobre a abordagem ARQCHECK serão coletadas através do Questionário de Avaliação Pós-Experimento (Q4 → questões 3, 5, 6, 8; Q5 → questões 1, 2).

4.8. Questões em aberto

- Qual a influência do nível de conhecimento do inspetor no domínio do problema ao utilizar a abordagem ARQCHECK ou SIEMENS para identificar defeitos?
- Qual a influência do nível de conhecimento do inspetor em arquitetura de software ao utilizar a abordagem ARQCHECK ou SIEMENS para identificar defeitos?
- Qual o principal tipo de defeito encontrado por ARQCHECK ?

5. PLANEJAMENTO

5.1. Formulação de Hipóteses

A premissa por trás de uma abordagem de inspeção em geral é a orientação provida ao usuário no que diz respeito à condução da atividade em questão. Para avaliar essa premissa, realizamos uma comparação entre os defeitos detectados pela abordagem proposta e os resultados obtidos por uma abordagem utilizada em um contexto industrial.

Portanto a hipótese nula que desejamos refutar através desse estudo é:

Hipótese nula (H0): Não existe diferença entre os resultados encontrados pela abordagem ARQCHECK e a abordagem SIEMENS em relação à detecção de defeitos arquiteturais

Para se obter resultados que venham a refutar ou a confirmar a hipótese H0, a eficácia e a eficiência das abordagens foram as características avaliadas.

Portanto, para cada uma dessas duas características, as seguintes hipóteses foram definidas:

- Eficácia

DA = Lista de defeitos encontrados pelos inspetores que utilizaram a abordagem ARQCHECK.

DS = Lista de defeitos encontrados pela abordagem SIEMENS.

Hipótese nula A (H0 A): A abordagem ARQCHECK permite identificar os mesmos defeitos identificados através da abordagem SIEMENS.

H0 A: DA = DS

Hipótese Alternativa A (H1 A): ARQCHECK permite encontrar defeitos que não foram encontrados pela abordagem SIEMENS.

H1 A: DA > DS

Hipótese Alternativa A (H2 A): A abordagem da SIEMENS permite encontrar defeitos que não foram encontrados pela abordagem ARQCHECK.

H2 A: DA > DS

- Eficiência

  TA = Tempo médio das inspeções utilizando a abordagem ARQCHECK.

  QA = Quantidade de defeitos encontrados pela abordagem ARQCHECK

  TS = Tempo médio das inspeções utilizando a abordagem SIEMENS.

  QS = Quantidade de defeitos encontrados pela abordagem SIEMENS

Hipótese nula B (H0 B): A relação entre a quantidade de defeitos encontrados e o tempo médio de inspeção utilizando a abordagem ARQCHECK e a abordagem SIEMENS são similares.

H0 B: QA/TA ≈ QS/TS

Hipótese Alternativa B (H1 B): A relação entre a quantidade de defeitos encontrados e o tempo médio de inspeção utilizando a abordagem ARQCHECK é maior que o da abordagem SIEMENS.

H0 B: QA/TA > QS/TS

Hipótese Alternativa B (H2 B): A relação entre a quantidade de defeitos encontrados e o tempo médio de inspeção utilizando a abordagem ARQCHECK é menor que o da abordagem SIEMENS.

H1 B: TI < TS

5.2. Seleção de Variáveis

5.2.1. Independentes

Variável	Valores	Descrição
DOC	Arquitetura_AppMenuX85.pdf	Documento arquitetural a ser inspecionado.
DOC	Requisitos_AppMenuX85.pdf	Documento de requisitos
TEC	ARQCHECK ou SIEMENS	Abordagem de inspeção arquitetural utilizada.
EXP	A: acima da média B: na média C: abaixo da média	Caracterização dos participantes em relação às competências necessárias para a aplicação de TEC e em relação ao conhecimento do problema.
	0: Não especialista 1: Especialista	

5.2.2. Dependentes

Variável	Valores	Descrição
TEMP	Inteiro	O tempo gasto por cada participante durante a detecção de defeitos. O tempo é contabilizado em minutos.
DEF	Inteiro	A quantidade de defeitos encontrados por cada participante é registrada.
FAL	Inteiro	A quantidade de falso-positivos encontrados por cada participante é registrada.
TEC_TIP	Inteiro	Quantidade de defeitos de um determinado tipo encontrados por cada abordagem.

5.3. Seleção dos Participantes

5.3.1. Critério de Seleção de Participantes

Os participantes do estudo serão provenientes da lista de estudantes matriculados nos cursos de pós-graduação do PESC da COPPE/UFRJ. Entretanto, para participar do estudo, os estudantes deverão obrigatoriamente:

- Manifestar interesse em participar do estudo, assinando o Formulário de Consentimento, e
- Já terem participado do primeiro experimento realizado com o ARQCHECK. Pois, durante esse experimento, foi realizado um extenso treinamento sobre os conceitos relacionados à arquitetura de software e que por motivos de tempo não poderá ser repetido.

5.3.2. Critério de Seleção de Grupos

Caso a quantidade de participantes seja superior a três e a avaliação de suas competências individuais aponte diferenças significativas entre os níveis de qualificação, os participantes poderão ser agrupados em dois diferentes grupos, como forma de minimizar o impacto do nível de qualificação dos participantes nos resultados do estudo.

Dessa forma, haveria um grupo com alta experiência e outro grupo com baixa experiência, no que diz respeito às competências necessárias para o projeto de arquitetura de software.

5.3.3. Técnicas de Amostragem

Os participantes do estudo serão escolhidos através de uma amostra não-probabilística baseada por conveniência (WOHLIN *et al.*, 2000).

5.4. Projeto do Experimento

5.4.1. Objetos

A abordagem para inspeção de documentos arquiteturais baseada em *checklist*.

5.4.2. Princípios

- **Aleatoriedade.** Não há aleatoriedade na atribuição do objeto de estudo aos participantes, visto que todos os participantes irão utilizar o mesmo objeto. Os participantes, conforme descrito na seção 5.3, não serão selecionados aleatoriamente da população, pois serão provenientes do grupo ESE da COPPE/UFRJ, que estiverem disponíveis.
- **Blocagem**. Não será possível a realização de agrupamentos devido ao número limitado de participantes.
- **Balanceamento**. Como a amostra de participantes será proveniente do grupo ESE da COPPE/UFRJ, não será possível garantir o balanceamento do conjunto de dados do estudo de acordo com a qualificação individual dos participantes.

5.4.3. Tipo

O estudo experimental é caracterizado como um pré-experimento onde será avaliado um único objeto (ARQCHECK) através da comparação com um controle (SIEMENS).

5.5. Instrumentação

5.5.1. Descrição da Instrumentação

Para obter as competências e experiências dos participantes do estudo serão utilizados o Formulário de Caracterização de Experiência. Esses dados são a entrada para a caracterização da qualificação dos participantes e, portanto, são variáveis independentes do estudo.

Durante a inspeção, um dos documentos necessários para sua realização é o Documento de Requisitos. Esse documento deve identificar os requisitos que serão utilizados como base para a identificação de discrepâncias na arquitetura do software.

Além do Documento de Requisitos, um outro documento essencial pra a realização do experimento é o Documento Arquitetural. Esse documento é responsável por descrever a arquitetura do software a ser avaliada.

As medidas quantitativas do estudo serão coletadas através do preenchimento do Relatório de Discrepâncias.

As medidas qualitativas dos participantes sobre a aplicação da técnica, além da condução do experimento, serão coletadas ao final do estudo através do Questionário de Avaliação Pós-Experimento e do Registro de Entrevistas.

5.5.2. Apoio à Análise Quantitativa

A avaliação quantitativa será realizada em cima dos dados coletados nos formulários de discrepâncias pertencentes à ARQCHECK.

5.5.3. Apoio à Análise Qualitativa

A avaliação qualitativa será em cima dos dados preenchidos pelos participantes do Questionário de Avaliação Pós-Experimento e do Registro de Entrevistas coletados ao final da etapa de detecção de defeitos.

5.6. Mecanismos de Análise

5.6.1. Testes Estatísticos

De forma a avaliar se a aplicação de ARQCHECK apóia em maior grau a detecção de defeitos, em relação ao apoio à detecção de falso-positivos, será utilizado o teste Binomial, que se caracteriza por ser do tipo não-paramétrico (WOHLIN *et al.*, 2000).

5.6.2. Critérios para Eliminação de Outlier

Não se aplica devido ao número limitado de participantes

5.7. Planejamento de execução

5.7.1. Definição de Execução do Estudo Experimental

O estudo experimental de viabilidade será conduzido em apenas uma sessão mas de forma remota, ou seja, os participantes não realizaram a inspeção ao mesmo tempo e nem no mesmo lugar.

Nessa sessão, os candidatos responderão ao Formulário de Caracterização visando identificar a competência e a experiência relacionadas ao propósito do estudo, além de se buscar identificar o conhecimento do participante no domínio do problema ao qual o artefato inspecionado pertence.

Após a caracterização, os participantes serão orientados a preencher um Formulário de Consentimento e receberão um documento que descreve a abordagem ARQCHECK. Esse documento visa explicar os principais conceitos relacionados à aplicação dessa abordagem para a detecção de defeitos.

Além disso, será distribuído um documento arquitetural que deverá ser avaliado e o *checklist* de avaliação que compõem a abordagem ARQCHECK. Além desses artefatos, será distribuído um documento de requisitos que deverá ser utilizado como base para a avaliação e um relatório de discrepâncias que deverá ser preenchido a medida que o participante identificar discrepâncias no artefato avaliado. Não existe um limite de tempo para que essa avaliação seja realizada.

Após a conclusão do experimento, será pedido a cada estudante, o preenchimento do Questionário de Avaliação Pós-Experimento, de forma a obter-se a sua avaliação qualitativa a respeito da aplicação da abordagem ARQCHECK, e dos procedimentos do estudo experimental. Além disso, entrevistas individuais podem ser conduzidas, onde cada participante será incentivado a emitir quaisquer opiniões a respeito da abordagem e da condução do experimento, que serão descritas em um Registro de Entrevistas.

Em posse dos Relatórios de Discrepância gerados por cada participante, o experimentalista, com o apoio dos participantes e de especialistas em arquitetura, irá classificar as discrepâncias identificadas em defeitos ou em falso-positivos. Essas duas listas serão analisadas pelos autores do documento arquitetural (profissionais da SIEMENS) que avaliarão se a classificação foi corretamente realizada.

Após um retorno dessa avaliação, o experimentalista terá acesso aos dados obtidos pela inspeção realizada através da abordagem da SIEMENS e a análise dos dados poderá ser iniciada.

5.7.2. Artefatos

A documentação a ser utilizada no estudo inclui:

- Formulário de Consentimento na Participação do Estudo;
- Formulário de Caracterização de Experiência dos Participantes;
- Documento explicativo sobre a abordagem
- A abordagem ARQCHECK, que inclui: o *Checklist* de avaliação, a Taxonomia de Defeitos e o Relatório de Discrepâncias;
- Documento de Requisitos e Documento Arquitetural usados durante a avaliação
- Questionário de Avaliação Pós-Experimento;
- Registro de Entrevistas.

5.8. Validade dos Resultados

Apesar de havermos identificado possíveis ameaças à validade dos resultados, não extrairemos conclusões além das limitações impostas por estas ameaças, o que não invalidará, portanto, os resultados do estudo.

5.8.1. Validade de Conclusão

Relacionada a questões que ameaçam a habilidade de traçar conclusões corretas sobre o relacionamento entre tratamentos e resultados de um estudo experimental.

- *Confiança das Medidas:* Dois dos objetivos que procuramos alcançar com esse estudo é identificar, em comparação aos resultados obtidos pela SIEMENS, o número de defeitos reais e que tipo de defeito a abordagem ARQCHECK identificou com maior intensidade. Sendo assim, uma potencial ameaça à validade de conclusão do estudo pode ser identificada. Essa ameaça se refere à classificação de cada discrepância em defeitos por parte dos profissionais da SIEMENS, por serem os autores do documento arquitetural. Entretanto, esse processo é deveras subjetivo, o que representa uma ameaça.
- *Heterogeneidade Aleatória dos Participantes*: os participantes serão alunos de pós-graduação em Engenharia de Software, portanto, não esperamos participantes com níveis de competências tão discrepantes, que possa a vir comprometer a validade do estudo. Contudo devido à pequena quantidade de participantes, opções como a blocagem, não poderão ser usadas visando minimizar esse tipo de ameaça, caso ocorra.
- *Confiabilidade na Implementação do Tratamento:* Este risco está relacionado à aplicação do tratamento não ser similar entre os diferentes participantes do estudo. A abordagem ARQCHECK compreende um conjunto de documentos padronizados para sua aplicação e um treinamento será realizado visando também padronizar a aplicação da abordagem

5.8.2. Validade Interna

Relacionada ao risco de outros fatores não identificados a priori, terem influenciado um eventual relacionamento de causalidade entre tratamento e resultado, sem o conhecimento prévio do experimentador.

- *História:* os participantes já aplicaram a abordagem ARQCHECK em um primeiro estudo. Contudo, pelo uso de *checklist* não utilizar um procedimento de execução obrigatório e devido às evoluções significativas que foram feitas sobre esse artefato, acreditamos que o efeito de história não seja pertinente ao estudo.
- *Instrumentação:* os formulários a serem utilizados no estudo experimental podem influenciar de forma significativa a condução do estudo, afetando negativamente seus resultados. Entretanto, por tratar-se de um estudo de viabilidade, essa questão não representa uma ameaça significativa à validade do estudo, haja vista que a avaliação da viabilidade de ARQCHECK também aplica-se à sua documentação associada.
- *Seleção: a* amostra dos participantes não será aleatória, visto que será proveniente da disponibilidade de pesquisadores do grupo de pesquisa em que o experimentalista pertence.

- *Plágio (plagiarism):* um típico risco na condução de estudos experimentais em ambiente acadêmico é a potencial troca de informações entre os participantes sobre as tarefas do estudo. Entretanto, essa ameaça não é pertinente, visto que os participantes se comprometeram a não se comunicarem durante a realização do estudo.
- *Ameaças a Múltiplos Grupos:* essas ameaças não se aplicam ao estudo uma vez que não será possível a realização de blocagem.

5.8.3. Validade de Construção

Problemas relacionados à ameaça de generalizar os resultados do estudo à teoria que o sustenta. As principais ameaças são:

- *Viés mono-operação:* o estudo experimental irá utilizar um único documento arquitetural como objeto de inspeção; portanto, o estudo pode não ser representativo da teoria sob a qual se sustenta, uma vez que pode não ficar claro se a causa dos resultados do estudo é decorrente da aplicação de ARQCHECK ou da "facilidade" de detecção de defeitos no documento utilizado.
- *Teste*: como forma de evitar possíveis influências no desempenho dos participantes, o objetivo do estudo não será apresentado aos participantes. Além disso, os participantes não serão envolvidos com discussão sobre as supostas vantagens e desvantagens da utilização de ARQCHECK, e de abordagens de avaliação arquitetural em geral. Entretanto, devido à presença da ameaça "*Interação entre Tratamentos Diferentes*" descrita abaixo, não temos como garantir a ausência dessa ameaça.
- *Interação entre Teste e Tratamento:* O fato dos participantes terem participados de um estudo piloto de ARQCHECK faz com que eles conheçam o propósito da abordagem ARQCHECK, o que deve ser levado em consideração quando as conclusões sobre esse estudo forem realizadas..
- *Apreensão de Avaliação:* uma possível ameaça é a probabilidade dos participantes "maquiarem" seus dados devido a expectativas pessoais sobre a avaliação de seus resultados. Entretanto, essa ameaça não será levada em consideração visto que o estudo será conduzido por pesquisadores que participarão por livre vontade ao estudo.

5.8.4. Validade Externa

Questões relacionadas à ameaça dos resultados do estudo não serem generalizáveis a projetos reais na indústria. As principais ameaças identificadas foram:

- *Interação entre seleção e tratamento:* a amostra dos participantes selecionados não é representativa dos profissionais da indústria brasileira de software. Além de possuírem relativa experiência industrial, grande parte dos pesquisadores ESE da COPPE/UFRJ possui acesso aos resultados da aplicação de uma pesquisa tecnológica de ponta, o que não ocorre à maioria dos profissionais da indústria de software. Pesquisas têm demonstrado que muitos dos conceitos de Engenharia de Software ainda não estão difundidos adequadamente na indústria de software brasileira (VILLELA, 2004). Portanto, uma ameaça considerável aos resultados do estudo seria atribuir um eventual sucesso do estudo à aplicação da técnica em si, quando na realidade poderia ter sido efeito da alta qualificação do participante, representando um *confounding factor.*
- *Interação entre Ambiente e Tratamento:* por utilizar um documento arquitetural criado em um ambiente industrial, acreditamos que essa ameaça não seja concretizada.

Apêndice E – Questionário de Avaliação Pós-Experimento

Grupo de Engenharia de Software Experimental

Questionário de Avaliação Pós-Experimento

Por favor, responda o questionário abaixo de forma a nos permitir o registro de sua opinião sobre a aplicação da abordagem para inspeção de documentos arquitetural baseada em *checklist* (ArqCheck), além dos procedimentos envolvidos na condução do exercício. A sua opinião é muito importante para nós.

1. Como você classificou o grau de dificuldade da aplicação do *checklist*?

___Muito Fácil ___Fácil ___Mediano ___Difícil ___Muito Difícil

2. Na sua opinião, quais aspectos da técnica tornam sua aplicação fácil / difícil de usar?

3. Você identificou os defeitos utilizando somente os itens de avaliação presentes no *checklist*?

___ Sim. Utilizei somente o conhecimento presente nos itens para avaliar o documento arquitetural fornecido.
___ Não. Eu utilizei conhecimento próprio

4. Caso você tenha identificado defeitos com o auxílio de outro tipo de conhecimento, além do presente no *checklist*, por exemplo, conhecimento do domínio, identifique para que defeitos isso ocorreu e que tipo de conhecimento que você utilizou.

5. Como o *checklist* o auxiliou a identificar defeitos nos requisitos?

___ Negativamente. O *checklist* atuou como um obstáculo. O meu desempenho teria sido melhor se eu não o tivesse utilizado.
___ Neutro. Acho que encontraria os mesmos defeitos caso não tivesse utilizado o *checklist*
___ Positivamente. O *checklist* me auxiliou na detecção de defeitos. Talvez não tivesse detectado alguns defeitos caso não o tivesse utilizado.

6. Como as "Guias de identificação de contexto" auxiliaram na identificação de defeitos relacionados a requisitos de qualidade?

___ Negativamente. As guias atuaram como um obstáculo. O meu desempenho teria sido melhor se eu não a tivesse utilizado.
___ Neutro. Acho que encontraria os mesmos defeitos caso não tivesse utilizado as guias

___ Positivamente. As guias me auxiliaram na detecção de defeitos. Talvez não tivesse detectado alguns defeitos caso não a tivesse utilizado.

7. **Existe** algum **item de avaliação que não foi compreendido? Se sim, identifique-o.**

8. **Você utilizaria a técnica em inspeções futuras?**

___Não. ___ Talvez. ___ Sim.

9. **Na sua opinião, como a técnica poderia ser melhorada?**

   Itens de avaliação para identificar defeitos:

   Taxonomia para categorização de defeitos:

10. **Por favor, registre quaisquer comentários que julgar pertinente.**

Muito obrigado por sua participação!

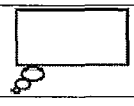
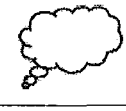
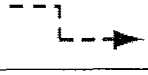
Anexo A – Notação para Modelar Processo de Software

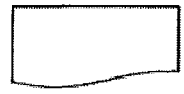
Este anexo apresenta a notação apresentada em (VILLELA, 2004) utilizada na modelagem de processo de software. Esta notação foi utilizada para descrever todos os processos presentes nessa dissertação.

A descrição de um processo apresenta os aspectos que devem ser considerados, os produtos gerados e os responsáveis ou envolvidos em cada atividade.

FALBO (1998) definiu uma ontologia de processo de software visando identificar e descrever os principais elementos que o compõem. VILLELA (2004) definiu uma abordagem de modelagem de processos organizacionais que permite a representação gráfica dos itens de um processo. Essa representação gráfica está parcialmente descrita na Tabela A.1.

Tabela A.1 – Notação de modelagem de processo de software

Entidade	Descrição	Forma de Representação
Processo ou Sub-processos	Sub-processos são processos que fazem parte de um processo mais amplo.	
Ator	Objeto que representa uma pessoa, agente ou unidade organizacional.	
Atividade Elementar	Atividades são tarefas ou trabalhos a serem realizados. Uma atividade requer recursos e pode consumir ou produzir artefatos. Uma atividade atômica consiste em uma atividade que não pode ser decomposta em outras atividades.	
Atividade Composta	Uma atividade composta consiste em uma atividade que pode ser decomposta em outras atividades.	
Conhecimento Explícito	Objeto que representa um conhecimento que pode ser expresso em palavras e números e ser facilmente transmitido e compartilhado.	
Conhecimento Tácito	Objeto que representa um conhecimento que é altamente pessoal e difícil de formalizar, o que o torna também difícil de ser compartilhado.	
Fluxo de Entrada / Saída	Ligação que estabelece um insumo (se o fluxo é de entrada) ou um produto de uma atividade (se o fluxo é de saída).	
Dependência entre Atividades	Atividades, em qualquer nível, podem depender da finalização de outras atividades, denominadas pré-atividades. A dependência entre as atividades do processo é representada por esse símbolo.	

Documento	Documento são produtos de software produzidos ou consumidos por atividades durante a sua realização	
Estado Inicial	Objeto puramente notacional, proveniente dos diagramas de estado e que indica onde é iniciado o fluxo de atividades que definem um processo ou uma atividade composta.	
Estado Final	Objeto puramente notacional, proveniente dos diagramas de estado e que indica onde é encerrado o fluxo de atividades que definem um processo ou uma atividade composta.	

Anexo B – Táticas Arquiteturais

Esse apêndice tem como objetivo descrever as diferentes táticas arquiteturais usadas como base para a criação dos itens do checklist que compõe a abordagem para inspeção de documentos arquiteturais apresentada nessa dissertação. Essas táticas foram extraídas principalmente de (BASS et al., 2003).

B.1 – Táticas relativas ao atributo de qualidade Desempenho

Desempenho para um software consiste principalmente no tempo gasto para realizar um determinado serviço. Essa quantidade de tempo está relacionada a duas coisas: à disponibilidade do recurso para realizar o serviço e à forma como o serviço é executado.

No contexto de arquitetura, o atendimento a um requisito de desempenho influencia principalmente na forma como os fluxos de execução são realizados, ou seja, na forma como os elementos arquiteturais se comunicam para realizar uma determinada responsabilidade.

Entre as principais táticas arquiteturais que podem ser usadas visando atender à requisitos desse tipo, temos:

- **Controle da geração de eventos:** Procurar definir elementos arquiteturais e seus relacionamentos visando reduzir a freqüência de execução de um determinado evento. Como por exemplo, o uso de um *cache* para armazenar informação no lugar de ter que re-executar o processo de obtenção quando necessitar ter acesso à informação, o que permitiria um maior desempenho na realização da tarefa.
- **Controle da freqüência de eventos externos**: Se não existe nenhum controle sobre a chegada de eventos gerados externamente, uma fila de requisições pode perder informações principalmente se a freqüência de processamento dessas requisições for muito baixa. No lugar de tratar cada evento de forma independente, essa tática propõe agrupar esses eventos em blocos para que possam ser tratados. O processamento de requisições em bloco costuma ter um melhor desempenho do que se fossem processadas de forma individual.
- **Redução do overhead computacional**: Elementos intermediários são elementos utilizados normalmente como mediadores de comunicação. A presença de desse tipo de elementos pode aumentar o processamento de um fluxo de eventos, por exemplo. Sendo assim, a eliminação desse tipo de elemento é aconselhável.

- **Aumento da concorrência lógica:** Durante a definição da arquitetura, deve-se identificar que seqüências de execução podem ser realizadas em paralelo, o que pode melhorar o desempenho do software. Dependendo das características do sistema, abordagens como cliente-servidor pode ser utilizada. Nesse caso, os diferentes clientes executariam funcionalidades distintas, por exemplo.

B.2 – Táticas relativas ao atributo de qualidade Disponibilidade

Disponibilidade consiste em permitir que o software execute as funcionalidades sempre que o usuário necessite. No contexto de arquitetura de software, a disponibilidade se traduz na garantia do bom funcionamento dos elementos arquiteturais relacionados a uma responsabilidade.

As possíveis táticas arquiteturais que podem ser utilizadas para garantir essa disponibilidade são:

- **Ping/echo:** Propõe a definição de um elemento arquitetural que seja responsável por enviar um *ping* ao elemento que deve sempre estar disponível. Esse elemento espera portanto receber o *eco* de volta, dentro de um tempo pré-estabelecido, indicando a disponibilidade do elemento principal.
- **Heartbeat:** Propõe que o elemento arquitetural emita um sinal (*heartbeat*) periodicamente e que deve ser recebido por um determinado elemento. Se o *heartbeat* falhar, supõe-se que o elemento emissor falhou e um procedimento de correção de falhas é executado.
- **Redundância Ativa:** Propõe que determinados componentes possuam redundâncias. Esses componentes redundantes respondem aos eventos em paralelo para que ele fique sincronizado com o componente principal. Caso o principal fique indisponível, a redundância deve ocupar o seu lugar.

B.3 – Táticas relativas ao atributo de qualidade Modificabilidade

Atender a um requisito de modificabilidade significa que, para as funcionalidades relacionadas a esse requisito, o custo de alteração deve ser o menor possível.

No contexto de uma arquitetura de software, modificabilidade está relacionada à forma como a arquitetura foi projetada. Para que ela seja facilmente modificável, os elementos que possuem grande chance de sofrerem alterações devem ser projetados visando, caso ocorra alterações em determinados elementos arquiteturais, evitar a propagação dessas modificações em outros elementos que compõem a arquitetura.

Este grupo de táticas tem por objetivo reduzir o número de módulos afetados por uma modificação:

- **Manter a coerência semântica**: Propõem que seja feita atenção aos módulos em relação às funcionalidades que eles implementam e os relacionamentos que eles realizam. O ideal é que tudo isso seja feito em um mesmo contexto semântico, o que facilitaria a identificação dos elementos que devem ser modificados e dificultaria a propagação das mudanças no caso de alterações. Um exemplo seria as separações de responsabilidades proposta pelo MVC.
- **Isolar serviços comuns:** Funcionalidades que são comuns a vários módulos deveriam ser isoladas em um único módulo e quem necessitasse executá-las o fariam a través desse módulo.
- **Isolar candidatos a alterações:** Identificar as funcionalidades que possuem grande possibilidade de serem alteras e isolá-las das que são estáveis. Esse tipo de tática costuma ser aplicada em um contexto de linha de produto, por exemplo.
- **Abstrair informações**: Propõem dividir as informações acessíveis de um módulo em dois tipos: públicas e privadas. Diminuindo assim a propagação de mudanças caso esse módulo deva ser alterado.
- **Separar as interfaces da implementação do módulo:** Definir interfaces para os módulos que apresentem alta probabilidade de modificações e evitar realizar modificações nessas interfaces.
- **Usar intermediários**: Se B possui algum tipo de dependência em relação a A que não seja semântica, é possível inserir um intermediário entre B e A que gerencie as atividades associadas com esta dependência, diminuindo assim a dependência entre A e B.

B.4 – Táticas relativas ao atributo de qualidade Segurança

Segurança está relacionado à habilidade do software em restringir o uso não-autorizado de seus serviços, permitindo o seu uso somente por usuários legítimos.

No contexto arquitetural, esse tipo de requisito influencia o projeto da arquitetura através de novas funcionalidades adicionais que devem ser atendidas.

- **Autenticação de usuários**: Autenticar é assegurar que um usuário ou computador remoto é de fato quem diz ser.
- **Autorização dos usuários**: Autorizar é assegurar que um usuário autenticado possui os direitos para acessar ou modificar tanto dados quanto serviços.
- **Limitar a exposição**: Garantir que o acesso a um componente que contém dados seguros é feito somente por quem realmente necessita ter acesso a esses dados.

B.5 – Táticas relativas ao atributo de qualidade Testabilidade

Visto que testar consome uma grande percentagem do custo de desenvolvimento do sistema, qualquer coisa que o usuário possa fazer para reduzir este custo irá render grandes benefícios. Sendo assim, testabilidade em um software consiste em oferecer facilidades para a realização de teste em relação a determinadas funcionalidades.

No contexto arquitetural, um requisito de testabilidade influencia através da necessidade em criar formas adicionais de acessar e controlar as funcionalidades e os dados dos elementos arquiteturais relacionados ao requisito especificado.

- **Separar a interface da implementação**: Separar a interface da implementação permite a substituição das implementações por vários propósitos relativos aos testes.
- **Interfaces específicas de acesso**: A definição de interfaces específica para a realização de teste permite capturar ou atribuir de valores de variáveis para um componente durante a realização dos testes, independente da sua execução normal.
- **Monitores**: O componente pode registrar a situação, o nível de performance, a capacidade, a segurança ou outra informação acessível através de uma interface. Esta interface pode ser uma interface permanente do componente ou pode ser introduzida temporariamente através de uma técnica de instrumentação como programação orientada a aspectos ou macros pré-processadas. Uma técnica comum é registrar eventos quando situações monitoradas forem alcançadas.

B.6 – Táticas relativas ao atributo de qualidade Usabilidade

Usabilidade em um software consiste em permitir sua fácil utilização por parte do usuário, ou seja, facilidade em entender o uso das funcionalidades, permitir a adaptação do software à suas necessidades e minimiza o impacto de possíveis erros omitidos pelo usuário.

No contexto de arquitetura de software, principalmente as características relacionadas à minimização de erros e adaptação às necessidades do usuário oferecem relevância.

Portanto, o atendimento a um requisito desse tipo consistem atender à funcionalidades adicionais, significando assim o projeto de novos elementos ou a distribuição das novas responsabilidades a elementos já existentes.