MÉTODO DE ARQUITETURA ATAM NO DESENVOLVIMENTO DE SISTEMAS

ATAM ARCHITECTURE METHOD IN SYSTEMS DEVELOPMENT

Paulo Cesar Dias Lima [1]
Andreia Carniello [2]

Data de entrega dos originais à redação em: 29/05/2015
e recebido para diagramação em: 13/11/2015.

Este trabalho propõe a utilização de uma arquitetura de referencia baseada na plataforma Java Enterprise Edition (Java EE), que corresponde a uma infraestrutura de componentes e serviços para o desenvolvimento e execução de aplicações corporativas na tecnologia Java, como proposta de solução para atender aos atributos de qualidade de um projeto de desenvolvimento de software. Utiliza-se uma solução baseada em um estilo de arquitetura padrão para aplicação Web da plataforma Java EE. O método de arquitetura ATAM (Architecture Tradeoff Analysis Method) é aplicado como ferramenta para avaliação da arquitetura de software proposta com relação à adequação aos requisitos de qualidade elicitados e priorizados. As atividades e as etapas deste método de arquitetura abordado foram exemplificadas de forma integrada como parte do processo de desenvolvimento e centradas na arquitetura de software. Um protótipo de desenvolvimento de um sistema com informações de prefeitura, que tem como escopo atender à lei de acesso à informação pública, é utilizado para exemplificar a abordagem proposta.

Palavras-chave: Arquitetura de Software. Avaliação da arquitetura. Desenvolvimento de Sistemas. Qualidade de Software.

This paper proposes the use of a reference architecture based on the Java Platform Enterprise Edition (Java EE), which corresponds to an infrastructure of components and services for the development and implementation of enterprise applications in Java technology, as a solution to meet quality attributes of a software development project. It uses a solution based on a standard architectural style for Web application Java EE platform. The architectural method ATAM (Architecture Tradeoff Analysis Method) is applied as a tool for assessment of software architecture regarding the adequacy of elicited and prioritized quality requirements. The activities and the steps of this architecture method were exemplified in an integrated manner as part of the development process and focused on software architecture. A development prototype of a system with information from the city hall, which is scoped to meet the law on access to public information, is used to illustrate the proposed approach.

Keywords: Software Architecture. Architecture evaluation. Systems Development. Software Quality.

1 INTRODUÇÃO

Preocupações com a deterioração em longo prazo da qualidade de projetos ágeis de larga escala, chamada débito técnico devido à perda de qualidade do *software* com a manutenção evolutiva e corretiva quando não é acompanhada da revisão da arquitetura de *software*, renovaram o interesse da comunidade de desenvolvimento de sistemas que utiliza métodos ágeis em arquitetura de *software*. Este interesse indica um crescente consenso de que uma base sólida de arquitetura de *software* permite que equipes evoluam rapidamente seus sistemas complexos de *software* utilizando desenvolvimento iterativo, incremental e direcionado para o atendimento dos requisitos de qualidade (BELLOMO; KAZMAN; GORTON, 2015).

Um grande desafio que as equipes ágeis enfrentam é a construção de uma base de arquitetura de *software* que equilibre duas preocupações conflitantes:

- entregar em curto prazo requisitos funcionais, com base na filosofia ágil da entrega de valor ao usuário antecipadamente e frequentemente;
- conciliar os objetivos de atender aos atributos de qualidade imediatos e de longo prazo;

A primeira preocupação refere-se à implementação das funcionalidades mais importantes da aplicação do ponto de vista dos usuários de forma incremental através do desenvolvimento em partes de modo a disponibilizar seu uso o mais breve possível. A segunda preocupação refere-se às estruturas da arquitetura de *software* que atenda aos requisitos de qualidade como, por exemplo, desempenho, segurança, robustez à medida que o sistema entre em operação e ao longo da sua evolução e manutenção. Estas preocupações são conflitantes à medida que o compromisso com o prazo da entrega de funcionalidades logo no início do projeto nem sempre permite uma análise completa da

1 - Pós-Graduação em Gestão de Projetos em Desenvolvimento de Sistemas de Software - Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP) - Câmpus Guarulhos - Avenida Salgado Filho, 3501. Bairro: Vila Rio de Janeiro. CEP: 07115-000 - Guarulhos – SP. < pcdl1970@gmail.com >.
2 - Professora Doutora. Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP) - Câmpus Guarulhos. Avenida Salgado Filho, 3501. Bairro: Vila Rio de Janeiro. CEP: 07115-000 - Guarulhos – SP. < andreiacarniello@ifsp.edu.br >.

arquitetura que atenda a todos requisitos de qualidade (BELLOMO; KAZMAN; GORTON, 2015).

Segundo Verner, Sampson e Cerpa (2008), o desenvolvimento de sistemas de software é um processo caro, e muitas vezes difícil, à medida que os projetos de desenvolvimento de software são afetados por uma série de problemas, como má gestão de projetos, custos e atrasos no cronograma, software de má qualidade e desenvolvedores frustrados ao longo do projeto.

Segundo Hidding e Nicholas (2014), os índices das falhas nos projetos de tecnologia da informação permanecem elevados, mesmo depois de décadas de esforços para reduzi-los. A maioria dos esforços para melhorar o sucesso do projeto concentrou-se em variações dentro do tradicional paradigma de gerenciamento de projetos, como pelo aumento na utilização da base de conhecimento em gerência de projetos baseado no PMBOK *(Project Management Body of Knowledge*).

De acordo com Hidding e Nicholas (2014), quando projetos de TI desconsideram uma explícita arquitetura do produto final, a taxa de falhas dos projetos pode aumentar. A importância da arquitetura é apenas reconhecida moderadamente na literatura, por exemplo, por meio de textos sobre arquitetura de sistemas que mencionam que um bom projeto arquitetural pode ser um fator importante para determinar o sucesso de um sistema de *software*.

No trabalho de Pontes (2012), avaliou-se o uso do método de arquitetura para direcionar a evolução do *software* através de um estudo da utilização do método de avaliação ATAM como referência para um roteiro para evolução arquitetura. Porém não abordou a aplicação do método ATAM para validação da proposta inicial da arquitetura de *software* para atender aos requisitos de qualidade do projeto nas fases iniciais do desenvolvimento.

Em Cardoso, Christina Von Flach e Lima (2014), apresentou-se uma abordagem para avaliar a qualidade de arquiteturas de software extraídas por ferramentas de recuperação de estruturas a partir dos códigos fontes dos sistemas e com base em métodos de avaliação preliminar de arquitetura: SAAM (*Software Architecture Analysis Method*) e ATAM que podem ser aplicados para verificar o atendimento dos requisitos de qualidades sem precisar ter os sistemas construídos completamente. Porém é possível que no contexto de um novo sistema ou projeto de desenvolvimento não exista códigos fontes para levantar ou documentar uma nova arquitetura de *software*.

A norma ISO/IEC 12207 define os processos do ciclo de vida de *software* no padrão de sistemas e engenharia de *software*, dos quais o processo do projeto arquitetural de sistemas é definido como sendo responsável por identificar que requisitos serão alocados e atendidos por quais elementos do sistema.

Neste trabalho propõe-se uma abordagem pragmática para utilização do método ATAM para avaliação da solução da arquitetura de *software* apresentada para o atendimento dos requisitos não funcionais, ou seja, dos atributos de qualidades esperados para o sistema. A arquitetura proposta foi escolhida a partir da seleção de uma arquitetura de referência com base na plataforma Java EE. Um projeto de desenvolvimento de um protótipo do sistema de informações de prefeituras é utilizado como exemplo no qual foi aplicado esta abordagem proposta. A partir deste exemplo, no qual o método ATAM foi aplicado, foram avaliados os resultados obtidos para considerações, conclusões e propostas para estudos e trabalhos futuros.

2 ARQUITETURA DE SOFTWARE

Em Hilliard (2000), a arquitetura é definida como "a organização fundamental de um sistema inserido nos seus componentes, as relações entre eles e com o ambiente, e os princípios que orientam sua concepção e evolução".

Conforme Bass, Clements e Kazman (2012), os sistemas de *software* são construídos para atender aos objetivos de negócios das empresas e instituições. A arquitetura pode ser considerada como a ponte que interliga os objetivos de negócio ao sistema final em funcionamento. A arquitetura está direcionada para atender aos requisitos de qualidade.

Segundo a norma ISO/IEC/IEEE 42010, o projeto da arquitetura de um sistema, expresso em uma descrição da arquitetura, auxilia a compreensão da essência e das principais propriedades do sistema referentes ao seu comportamento, composição e evolução, que por sua vez endereçam as questões relacionadas à viabilidade, utilidade e facilidade de manutenção do sistema. As descrições de arquitetura são usadas pelas partes que criam, utilizam e gerenciam os sistemas atuais para melhorar a comunicação e cooperação, permitindo-lhes trabalhar de forma integrada e coerente.

Conforme descrevem Santos et al. (2015), a definição da arquitetura de *software* em estágios iniciais de desenvolvimento é um importante mecanismo de comunicação entre as partes interessadas. Ela envolve não somente as decisões e orientações que devem ser seguidas durante a evolução do *software*, mas também fornece discussão sobre os requisitos e suas partes conflitantes que o *software* possa ter. Decisões de arquitetura inconsistentes podem comprometer o sucesso de um projeto de *software*.

Em Vieira e Reis (2014), propôs-se um modelo baseado em MDA (*Model Driven Architecture*, Arquitetura Dirigida pelo Modelo) para o desenvolvimento de sistemas Web. Realizou-se estudo de procedimentos e técnicas que utilizam o MDA, metodologias ágeis e o DDD (*Domain-Driven Design*, Desenvolvimento Orientado a Domínio) visando aprimorar o desenvolvimento de *software* reduzindo impactos, como atrasos e altos custos, para o projeto. Porém não apresentou uma forma de avaliação e validação com relação ao atendimento dos requisitos de qualidade pela arquitetura de *software* proposta.

Os requisitos de qualidade, de acordo a norma NBR ISO/IEC 9126 para qualidade de produto de *software*, estão distribuídos em seis características principais conforme descritos na figura 1:

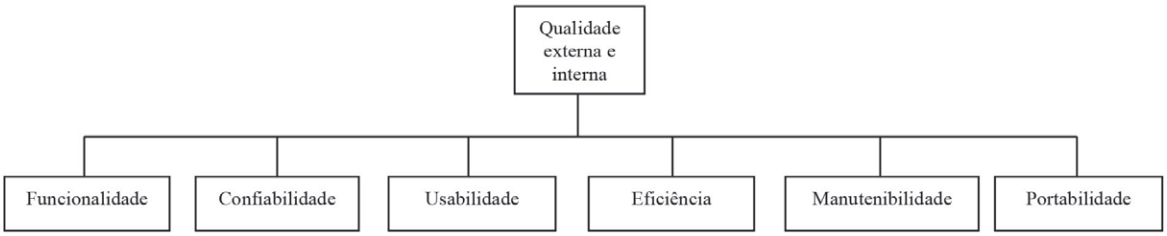

Figura 1 – Modelo de qualidade para qualidade externa e interna

Segundo a norma NBR ISO/IEC 9126, a qualidade interna é a totalidade das características do produto de software do ponto de vista interno. A qualidade interna é medida e avaliada com relação aos requisitos de qualidade interna. Detalhes da qualidade do produto de *software* podem ser melhorados durante a implementação do código, revisão e teste, mas a natureza fundamental da qualidade do produto de *software* representada pela qualidade interna mantém-se inalterada, a menos que seja alterada sua arquitetura de *software*.

A norma NBR ISO/IEC 9126 descreve cada um destes atributos da seguinte forma:

Funcionalidade: refere-se à capacidade do produto de software de prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado sob condições especificadas.

Confiabilidade: refere-se à capacidade do produto de software de manter um nível de desempenho especificado, quando usado em condições especificadas.

Usabilidade: refere-se à capacidade do produto de software de ser compreendido, aprendido, operado e atraente ao usuário, quando usado sob condições especificadas.

Eficiência: refere-se à capacidade do produto de software de apresentar desempenho apropriado, relativo à quantidade de recursos usados, sob condições especificadas.

Manutenibilidade: refere-se à capacidade do produto de software de ser modificado. As modificações podem incluir correções, melhorias ou adaptações do software devido a mudanças no ambiente e nos seus requisitos ou especificações funcionais.

Portabilidade: capacidade do produto de *software* de ser transferido de um ambiente para outro.

Conforme Kazman et al. (2014), arquitetura é mais comumente usada em projetos de médio a grande escala, que têm várias equipes e muita complexidade para qualquer indivíduo compreender plenamente. Tais projetos normalmente envolvem investimentos substanciais e têm duração de vários anos. Para estes projetos, as várias equipes, que podem ser distribuídas, precisam coordenar seus esforços. Nesses projetos, a alta gerência muitas vezes exige tanto curto prazo para lançar no mercado e uma supervisão adequada. Assim, o uso da arquitetura como uma ferramenta para o gerenciamento de projetos é cada vez mais necessário e comum.

Observa-se que a definição da arquitetura de software logo no inicio do processo de desenvolvimento e a sua avaliação com relação ao atendimento dos requisitos de qualidade levantados e analisados para proposta da solução da arquitetura consiste em uma abordagem referenciada em vários trabalhos e que pode aumentar as chances de sucesso nos projetos de desenvolvimento de sistemas. Neste trabalho utilizaremos o método ATAM como prática para avaliação da arquitetura de software proposta como solução para o sistema de Infoprefeituras, o qual deverá atender aos atributos dos requisitos não funcionais levantados e mapeados para o projeto.

3 O MÉTODO DE AVALIAÇÃO DE ARQUITETURA ATAM

Conforme Bass, Clements e Kazman (2012), a maioria dos sistemas de software complexos é requisitada a ser modificável e ter um bom desempenho. Eles também podem precisar ser seguros, interoperável, portátil em várias plataformas e confiável. Mas ao atender a um requisito de qualidade específico pode implicar em penalizar outro requisito também não funcional. Neste contexto faz-se necessário um método que permita a análise dos atributos de requisitos de qualidade, suas priorizações e implicações na arquitetura de *software* proposta para atender ao sistema. O ATAM é o método proposto para avaliar a arquitetura com relação ao atendimento dos requisitos de qualidade, apoiar no levantamento dos riscos e nas decisões que precisam ser balanceadas através de análise e negociação entre a equipe de arquitetura e as partes interessadas do projeto.

Kazman, Klein, Clements (2000) descrevem o ATAM como um método de análise organizado em torno da ideia de que os estilos arquiteturais são os principais determinantes de atributos de qualidade da arquitetura. O método é centrado na identificação dos objetivos de negócio que definem as metas dos atributos de qualidade. Com base nas metas dos atributos de qualidade, utiliza-se o ATAM para analisar como os estilos de arquitetura ajudam na realização destes objetivos.

Conforme Shaw e Garlan (1996), um estilo arquitetural inclui uma descrição dos tipos de componentes e sua topologia, uma descrição do padrão de dados e controle da interação entre os componentes, e uma descrição informal dos benefícios e desvantagens de usar esse estilo.

Segundo Kazman, Klein, Clements (2000), o ATAM é um método de avaliação de arquiteturas de software que realiza a avaliação de uma arquitetura com relação aos objetivos de atributos de qualidade. As avaliações ATAM expõem os riscos arquiteturais que potencialmente inibem a realização dos objetivos de negócio de uma

organização. O ATAM como um método de análise dos prós e contras de arquitetura recebe o seu nome porque ele não só revela o quão bem uma arquitetura atende os objetivos particulares de qualidade, mas também fornece direcionamentos sobre como esses objetivos de qualidade interagem uns com os outros, e como equilibrar os prós e contras do atendimento entre eles.

Segundo Kazman, Klein, Clements (2000), os resultados mais importantes do ATAM são melhores arquiteturas. O produto final gerado no ATAM é uma apresentação e/ou um relatório por escrito, que inclui os principais resultados da avaliação. O ATAM ajuda elucidar conjuntos de requisitos de qualidade em múltiplas dimensões, analisando os efeitos de cada requisito de forma isolada e, em seguida, as interações desses requisitos.

Segundo IONITA, HAMMER, OBBINK (2002), o ATAM é um método de arquitetura baseado em cenários que descrevem as principais interações entre as partes interessadas e o sistema para avaliação de atributos de qualidade como modificabilidade, portabilidade, extensibilidade e integralidade. Além disso, explora a interação entre os atributos de qualidade e suas interdependências destacando o mecanismo de balanceamento e oportunidades entre as diferentes qualidades.

Conforme Kazman, Klein, Clements (2000), no método ATAM utiliza-se três tipos de cenários:

- Cenários de casos usos: que envolvem usos típicos do sistema existente e são usados para levantar informações;
- Cenários de crescimento: que abrangem as alterações previstas para o sistema;
- Cenários exploratórios: abrangendo as alterações extremas de "estresse" que se espera do sistema.

Estes diferentes tipos de cenários são utilizados para mapear um sistema a partir de diferentes ângulos, otimizando as chances de confrontar as decisões de arquitetura com os principais riscos para mitigá-los.

Segundo Kazman, Klein, Clements (2000), para a condução com sucesso de uma sessão de avaliação do ATAM, os avaliadores devem entender a arquitetura do sistema, reconhecer os parâmetros arquiteturais, definir suas implicações com relação aos atributos de qualidade do sistema, e confrontar estas implicações com os requisitos. As áreas problemas na avaliação são chamadas "pontos de sensibilidade", "contrapontos" e "riscos". Um ponto de sensibilidade é uma propriedade de um ou mais componentes e as relações entre componentes que são críticas para a obtenção de resultados esperados dos atributos de qualidade. Os pontos de sensibilidade indicam para o arquiteto ou analista onde concentrar a atenção quando se tenta atender a realização de uma meta de qualidade. Um contraponto (*tradeoff*) é uma propriedade que afeta mais de um atributo e é um ponto de sensibilidade para mais de um atributo. Por exemplo, a alteração do nível de encriptação pode ter um impacto significativo sobre a segurança e desempenho. Aumentando o nível de criptografia melhora a segurança prevista, mas exige mais tempo de processamento. Se o processamento de uma mensagem confidencial tem uma exigência crítica do tempo de latência, em seguida, o nível de criptografia poderia ser um ponto de *tradeoff*. Os pontos de tradeoff são as decisões mais importantes que se pode fazer em uma arquitetura, que é por isso o ATAM direciona para considera-los cuidadosamente. Os valores particulares de pontos de sensibilidade podem tornar-se riscos quando realizados em uma arquitetura. Considere o exemplo, um valor particular na criptografia de nível 128 bits de criptografia pode representar um risco na arquitetura com relação aos possíveis impactos no desempenho e tempo de processamento das mensagens criptografadas.

Os três componentes usados para encontrar os riscos, pontos de sensibilidade e pontos de *tradeoff* são apresentadas na Figura 2: cenários de maior prioridade, as questões específicas de atributos, abordagens ou estilos arquiteturais.

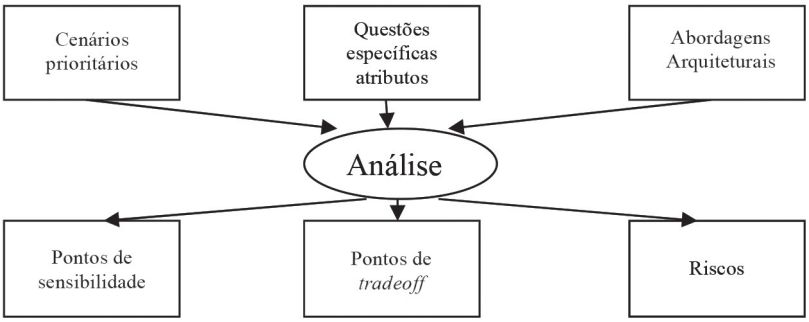

Figura 2 – Interações dos conceitos das áreas problemas

Durante a avaliação o arquiteto endereça os cenários através dos subconjuntos da arquitetura representada pelas abordagens arquiteturais. Para isso, o arquiteto deve identificar os componentes e conectores envolvidos na realização do cenário. Quando o arquiteto identifica os componentes relevantes os avaliadores confrontam com as perguntas e respostas específicas do atributo de qualidade. Algumas das respostas a estas perguntas levam à identificação de pontos de sensibilidade, alguns dos quais irão revelar-se riscos e/ou pontos de *tradeoff*.

Os atributos de qualidade devem ser entendidos através de cenários descritivos para avaliação dos atributos de qualidade. Uma sessão de avaliação do ATAM utiliza como entradas os requisitos iniciais do sistema e a descrição da arquitetura de software do sistema.

Conforme Kazman, Klein, Clements (2000), o método ATAM consiste de quatro fases: apresentação, investigação e análise, teste e relatório. Cada fase consiste em uma coleção de passos. A fase de apresentação envolve a troca de informações entre as apresentações. A fase de investigação e análise se preocupa com a avaliação dos requisitos de atributos chaves de qualidade versus as abordagens arquiteturais. A fase de teste

compara os resultados das fases anteriores para as necessidades das partes interessadas mais relevantes. Finalmente, a fase de relatório sumariza os resultados do ATAM.

Segundo Kazman, Klein, Clements (2000), o método ATAM é composto por nove etapas conforme a sequência a seguir:

1. Apresentar o ATAM: o líder da avaliação descreve o método de avaliação para o grupo de participantes, tenta alinhar suas expectativas e responder às questões que possam surgir.
2. Apresentar as diretrizes de negócio: o porta-voz do projeto, idealmente o gerente de projeto ou cliente do sistema, descreve quais objetivos de negócio são motivadores do esforço de desenvolvimento e, por conseguinte, aquilo que a arquitetura deverá principalmente atender, por exemplo, alta disponibilidade ou agilidade de lançamento no mercado ou forte segurança.
3. Apresentar a arquitetura: o arquiteto descreve a arquitetura, com foco na forma como ela atende as diretrizes de negócios. São abordados os requisitos de atributos de qualidade mapeados de acordo com as prioridades, complexidades e riscos para o projeto.
4. Identificar abordagens arquiteturais: as abordagens arquiteturais são identificadas pelo arquiteto, mas não são analisadas. São levantados os principais tipos de arquiteturas de referência que podem endereçar o atendimento dos requisitos de qualidade.
5. Gerar árvore utilitária de atributo de qualidade: os fatores de qualidade que compõem as qualidades do sistema (confiabilidade, usabilidade, eficiência, manutenibilidade, portabilidade) são elicitados, indicados em nível de cenários, anotados com estímulos e respostas, e priorizados.
6. Analisar as abordagens arquiteturais: com base nos fatores de alta prioridade identificados na etapa 5, as abordagens arquiteturais que endereçam esses fatores são levantadas e analisadas (por exemplo, uma abordagem arquitetural que visa reunir as metas de eficiência será submetida a uma análise de eficiência). Durante esta etapa, os riscos, os pontos sensíveis arquiteturalmente e os pontos de balanceamento são identificados.
7. Realizar um debate e priorizar cenários: descrever as principais interações entre as partes interessadas com o sistema e o comportamento esperado em cada cenário. Um conjunto abrangente de cenários é levantado por parte de todo o grupo de partes interessadas. Neste conjunto os cenários são priorizados através de um processo de votação envolvendo todo o grupo de interessados.
8. Analisar as abordagens arquiteturais: este passo reafirma as atividades do 6º passo, mas usando os cenários prioritários classificados na etapa 7. Os cenários são utilizados como casos de teste para validar a análise realizada até o momento. Esta análise pode revelar outras abordagens arquiteturais, os riscos, os pontos sensíveis e os pontos de balanceamento, que serão, então, documentados.
9. Apresentar os resultados: com base nas informações coletadas durante o ATAM (abordagens, cenários, perguntas específicas dos atributos de qualidade, árvore utilitária, riscos, não riscos, pontos sensíveis, vantagens e desvantagens), a equipe do ATAM apresenta os resultados para o conjunto das partes interessadas.

A partir da aplicação do método ATAM é possível analisar, revisar, priorizar os requisitos de qualidade e mapear as estruturas e elementos da arquitetura de software proposta que propiciarão o atendimento dos atributos de qualidade. O ATAM permite também avaliar os possíveis impactos do atendimento de um requisito não funcional específico em relação a outros atributos de qualidade, de modo a proporcionar uma forma de balanceamento entre os prós e contras das decisões arquiteturais.

4 EXEMPLO DE APLICAÇÃO DO MÉTODO ATAM NO DESENVOLVIMENTO DO SISTEMA INFOPREFEITURA

Neste trabalho o método ATAM foi aplicado no desenvolvimento de um sistema denominado InfoPrefeitura. O escopo deste sistema é criar um site na Internet para uma prefeitura fictícia a fim de atender a demanda da Lei Federal nº 9755/98 que descreve: "Dispõe sobre a criação de homepage na Internet, pelo Tribunal de Contas da União, para divulgação dos dados e informações que especifica, e dá outras providências", adicionando ainda informações sobre o município e outros serviços para a população.

Utilizou-se como arquitetura de referência um estilo de arquitetura de *software* da plataforma Java EE como base para a proposta da solução do sistema Infoprefeitura.

Segundo Johnson (2002), a plataforma Java EE apresenta os seguintes cenários para aplicações: Aplicação monolítica, Aplicação centrada na Web, Aplicação orientada a componentes de negócios, Aplicação orientada a exposição de serviços via Internet e Aplicação orientada a interoperabilidade entre sistemas.

A arquitetura de desenvolvimento utilizado baseia-se no estilo arquitetura Java EE centrada na Web. Neste estilo arquitetural os principais elementos, distribuídos em camadas, são:

- Cliente: composta normalmente pelo navegador da Internet;
- Apresentação: gera as paginas dinâmicas para interação com a camada cliente;
- Serviços: implementa as regras de negócios e interage com os dados persistidos do sistema.

As fases e os passos do método de avaliação de arquitetura ATAM aplicado ao exemplo do protótipo do sistema Infoprefeitura a seguir são apresentados:

Fase Apresentação:
ATAM Passo 1 – apresentando ATAM

Segundo Da Cruz Santos (2013), este passo é realizado pelo gerente de projeto e consiste em apresentar o método para os envolvidos no projeto, explicar os benefícios que podem ser obtidos, e mostrar a relação tempo gasto e qualidade do produto final que poderá ser alcançada.

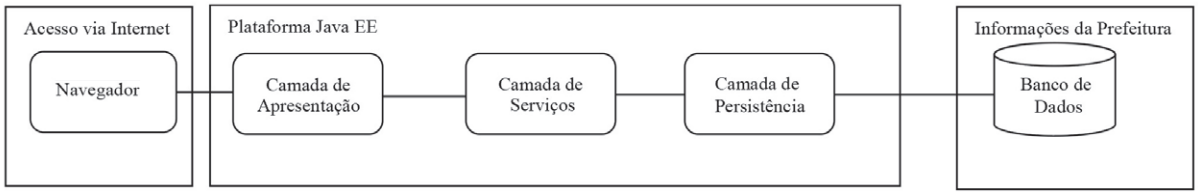

Figura 3 – Arquitetura de software apresentada para o sistema InfoPrefeitura

Neste trabalho, porém será proposta a apresentação pela equipe de Arquitetura de Sistemas, para enfatizar não só o processo ATAM, mas principalmente para explicar a importância da utilização de uma arquitetura de referência para este tipo de sistema como base para a solução proposta e também a reutilização de levantamento dos requisitos deste tipo de sistema voltado para qualidade de informação e lei de acesso a informação para possibilitar maior conformidade no atendimento da lei.

ATAM Passo 2 – Apresentação dos direcionadores de negócio

Para este projeto foram selecionados alguns dos atributos de qualidade tendo como base o levantamento realizado no trabalho de Arouck e Do Amaral (2013) referentes às categorias de atributos de qualidade de informação e lei de acesso à informação: disponibilidade, interoperabilidade, manutenibilidade, desempenho, segurança, testabilidade e usabilidade.

ATAM Passo 3 – Apresentação da arquitetura

No trabalho de Serrano, Serrano e Cavalcante (2015), é proposta uma especificação para Arquitetura de Software de Referência (ASR) baseada no paradigma orientado à convenção sobre configuração para novas contratações de desenvolvimento de Sistemas de Informação Governamentais por terceirizadas.

Neste trabalho será proposta a utilização da arquitetura de referencia para desenvolvimento web da plataforma *Java Enterprise Edition* (Java EE) como base para construção da arquitetura de *software*.

Conforme Jendrock et al. (2014), a plataforma Java EE fornece uma arquitetura baseada em componentes na qual é possível desenvolver aplicações corporativas utilizando uma infraestrutura de serviços existentes facilitando o processo de desenvolvimento e aumentando a produtividade das equipes de projeto.

Além disso, a plataforma Java EE é de código aberto, possui ampla disponibilidade de documentação, exemplos e referencias disponíveis para suportar o processo de capacitação de desenvolvedores.

Segundo Da Silva e Lucrédio (2014), os componentes *Enterprise JavaBeans* (EJB) da plataforma Java EE executados do lado do servidor que encapsulam a regra de negócios da aplicação. Tem como objetivo reduzir a complexidade da codificação de aplicações fornecendo serviços como controle de transação, concorrência, entre outros.

Na arquitetura de software proposta os componentes EJB serão utilizados para implementar a camada de serviços.

O arquiteto apresenta a arquitetura de *software* com a proposta de solução baseada na plataforma Java EE conforme Figura 3.

A seguir as descrições das estruturas que compõem a arquitetura proposta baseada na plataforma Java EE:

Camada cliente (Navegador): Corresponde a parte cliente que irá acessar a aplicação via Internet. Exemplos de navegadores são Microsoft Internet Explorer (IE), Google Chrome, Mozilla Firefox entre outros.

Camada de apresentação: Corresponde a camada de visão da aplicação. Apresenta as interfaces visuais através da qual o cliente interage com o sistema. Na plataforma Java EE a camada de apresentação pode ser implementada através de componentes que encapsulam as propriedades visuais e de interação com o navegador web através da geração de páginas da internet.

Camada de serviços: Corresponde a camada de serviços de negócios. Apresenta os componentes que implementam as funcionalidades e regras de negócio do sistema. Na plataforma Java EE corresponde aos componentes que expõe métodos e funções que são executadas para atender aos comportamentos esperados nos cenários de casos de usos da aplicação.

Camada de persistência: Corresponde a camada de integração com o armazenamento de dados utilizados pela aplicação. Através desta camada é possível recuperar o estado persistido das informações do sistema. Na plataforma Java EE seus componentes representam interfaces de persistência para acesso as diversas fontes de dados para serem utilizadas pelas aplicações.

Banco de dados: Corresponde ao repositório de informações do sistema. Pode ser implementado pela utilização de um Sistema de Gerenciamento de Banco de Dados Relacional (SGBDR) externo.

Fase Investigação e Análise: ATAM Passo 4 – Identificar as abordagens Arquiteturais:

As abordagens arquiteturais consideradas para o InfoPrefeitura são baseadas nos diversos cenários de aplicações da plataforma Java EE.

ATAM Passo 5 – Gerar Árvore Utilitária de Atributo de Qualidade:

As partes interessadas listaram e quantificaram (Quantidade RNF) os fatores de qualidade do sistema

Tabela1- Indicador Final (importância) das categorias de requisitos não funcionais do sistema InfoPrefeitura

Categoria Requisito Não Funcional	Quantidade RNF	Peso (1 a 5)	Indicador Final
Disponibilidade	2	3	6
Interoperabilidade	1	3	3
Manutenibilidade	4	5	20
Desempenho	2	5	10
Segurança	3	3	9
Testabilidade	3	5	15
Usabilidade	4	3	12

de acordo com as seguintes categorias de requisitos: disponibilidade, interoperabilidade, manutenibilidade, desempenho, segurança, testabilidade, usabilidade.

Para cada categoria de requisito de qualidade do sistema (RNF) foi atribuído um grau de importância através de uma escala numérica, na qual a menos relevante foi atribuído o peso um (menos importante) e a mais relevante foi atribuído o peso cinco (mais importante) de acordo com as expectativas e considerações dos envolvidos na definição dos requisitos do sistema.

Foram analisados os cenários de caso de uso e a ocorrência dos requisitos de qualidade e a importância de cada um no contexto do comportamento e resultado esperados no sistema. A partir desta análise obteve-se a quantidade de cada RNF por categoria cuja importância está representada por um Indicador Final que corresponde ao produto da quantidade de RNF multiplicada pelo peso atribuído à categoria do requisito. Deste modo o Indicador Final é obtido pela expressão: Indicador Final = Quantidade RNF x Peso.

ATAM Passo 6 – Analisar Abordagens Arquiteturais

Baseadas nos cenários mais prioritários identificados no passo anterior, as abordagens arquiteturais que endereçam esses cenários foram elicitadas e analisadas. Durante este passo os riscos potenciais, possíveis não-riscos, os pontos de sensibilidade e contrapontos foram identificados. As abordagens arquiteturais avaliadas como possíveis opções são: Aplicação centrada na Web; Aplicação orientada a componentes de negócios; Aplicação orientada a exposição de serviços via Internet.

Fase de Teste:
ATAM Passo 7 – Avaliar e priorizar cenários

Os cenários foram priorizados por votação com relação ao peso de cada um para atendimento aos requisitos de qualidade do sistema InfoPrefeitura. Os resultados foram pontuados na coluna Indicador Final.

ATAM Passo 8 – Reanalisar Abordagens Arquiteturais

Os cenários priorizados foram utilizados para reiterações do Passo 6 com objetivo de selecionar a abordagem arquitetural mais indicada para o sistema InfoPrefeitura. A abordagem selecionada para atender ao projeto da plataforma Java EE foi a Aplicação centrada na Web.

Fase Relatório:
ATAM Passo 9 – Apresentação dos Resultados

A abordagem arquitetural escolhida para atendimento aos requisitos de qualidade do sistema InfoPrefeitura foi o padrão de arquitetura Java EE denominado Aplicação Centrada na Web.

5 DISCUSSÃO DOS RESULTADOS OBTIDOS

A utilização de um estilo de arquitetura de software para a implementação de uma solução de desenvolvimento de sistemas possibilita a avaliação do atendimento dos requisitos de qualidade de forma objetiva e assertiva devido ao conhecimento das características e comportamento documentado dos estilos arquiteturais da plataforma Java EE.

A plataforma Java EE, por apresentar vários modelos de arquitetura padrão para o desenvolvimento de sistemas corporativos na linguagem Java, permite uma maior agilidade e objetividade na escolha do estilo arquitetural da plataforma.

O estilo arquitetura Java EE centrado na Web possibilita o desenvolvimento de sistemas para atender projetos com requisitos comuns a diversas aplicações que necessitam das características de software para serem executados e acessados via interface com navegador de Internet.

O método ATAM aborda os atributos de qualidade e possibilita avaliá-los simultaneamente no mesmo método e inclui a análise de *trade-off* ("perde-e-ganha") e balanceamento entre o atendimento dos requisitos de acordo com a prioridade e importância para o negócio.

O método ATAM foi aplicado como base na definição da arquitetura proposta no sistema de informações de prefeituras InfoPrefeitura. Foi justificada a decisão da arquitetura de software proposta pelo atendimento aos direcionadores do negócio e aos principais atributos de qualidades de acordo com a pontuação final de importância de cada uma das categorias de requisitos não-funcionais, seguindo os cenários indicados pelas partes interessadas.

A realização deste trabalho permitiu propor a utilização do método ATAM baseado em requisitos de qualidade e cenários para análise, projeto e avaliação da arquitetura de software e afirmar que o balanceamento dos requisitos de qualidade conflitantes direciona para uma abordagem assertiva na decisão da arquitetura.

6 CONCLUSÕES

Em Azevedo (2014), propôs-se uma abordagem para gerar recomendações de padrões mais adequados

a cada tipo de sistema baseadas na ocorrência de termos chave na descrição dos padrões para a construção da arquitetura de *software*. Porém a avaliação da arquitetura candidata e uma abordagem de balanceamento do atendimento dos requisitos de qualidades poderiam complementar a assertividade de solução arquiteturais propostas.

Segundo Coelho (2015), na fase de elaboração do processo de desenvolvimento proposto, a arquitetura do sistema é definida, na qual os requisitos anteriormente levantados são revistos e aprimorados e o levantamento de riscos revisado. Esta fase serve de sustentação à construção do *software* por meio da criação de protótipos, e da modelagem do sistema e de seus dados. Porém um método de avaliação de arquitetura como ATAM possibilitaria uma abordagem formal de validação do atendimento dos requisitos de qualidade do sistema.

A arquitetura de um sistema de *software* tem sido um aspecto importante do desenvolvimento de *software*. Uma arquitetura adequada pode aumentar a probabilidade de o sistema atender aos seus requisitos de qualidade.

A partir da pesquisa em arquitetura de *software* e da seleção do método ATAM, foram analisadas neste trabalho as características, propostas e indicações deste método para utilização como apoio à tomada de decisão na definição de arquiteturas de *software*. Uma abordagem pragmática de utilização do método ATAM foi aplicada em um projeto de sistema de informações de prefeituras, revelando o quão bem a arquitetura satisfaz os requisitos de qualidade do sistema.

Para trabalhos futuros seria interessante avaliar uma combinação do método ATAM juntamente com a elaboração de uma avaliação de custo e benefícios para análise das opções de investimentos para as possíveis soluções de arquitetura e desenvolvimento do projeto. Esta análise de custos e benefícios pode ser realizada, por exemplo, aplicando-se o método CBAM (*Cost Benefit Analysis Method*), a fim de avaliar a viabilidade de implementação da arquitetura proposta e a produtividade das equipes envolvidas nessa arquitetura com relação ao atendimento dos prazos, custos e benefícios esperados no projeto.

REFERÊNCIAS

AROUCK, Osmar; DO AMARAL, Sueli Angelica. **Atributos de qualidade da informação e a lei de acesso à informação.** In: Anais do Congresso Brasileiro de Biblioteconomia, Documentação e Ciência da Informação-FEBAB. 2013. p. 4690-4703.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. NBRISO/IEC9126-1 **Engenharia de software** - Qualidade de produto - Parte 1: Modelo de qualidade. 2003.

AZEVEDO, Rafael Pereira Martins et al. **Seleção de padrões para a arquitetura de software:** uma abordagem baseada em procura de termos e sinônimos. 2014.

BASS, L.; CLEMENTS, P.; KAZMAN, R. **Software architecture in practice** (3rd Edition) (SEI Series in Software Engineering). Addison-Wesley Professional, 2012.

BELLOMO, Stephany; KAZMAN, Rick; GORTON, Ian. **Insights from 15 Years of ATAM**. Data: Towards Agile Architecture.

CARDOSO, Mateus Passos Soares; CHRISTINA VON FLACH, G.; LIMA. **Crescencio. Avaliaç ao de Arquiteturas Recuperadas com base em Métodos de Avaliação Precoce**. WTDSoft 2014, p. 27.

COELHO, Alexandre Chaves. **Estudo e proposta de um processo de desenvolvimento de software em uma cooperativa de software livre**. 2015.

DA CRUZ SANTOS, Thiago. **APPLYING THE ATAM ON THE ARCHITECTURAL EVOLUTION OF AN ENTERPRISE SYSTEM**, 2013.

DA SILVA, Wilson Daniel; LUCRÉDIO, Daniel. **Utilização de componentes Enterprise Java Beans no desenvolvimento de Sistemas Web**. Revista TIS, v. 2, n. 3, 2014.

DA UNIÃO, CONTROLADORIA GERAL. **Acesso à Informação Pública:** Uma Introdução à Lei nº 12.527, de 18 de novembro de 2011. Brasília: CGU, 2011.

GUIDE, A. **Project Management Body of Knowledge (PMBOK® GUIDE**). In: Project Management Institute. 2012.

HIDDING, Gezinus J.; NICHOLAS, John M. **Reducing IT Project Management Failures:** Early Empirical Results. In: System Sciences (HICSS), 2014 47th Hawaii International Conference on. IEEE, 2014. p. 4305-4314.

HILLIARD, Rich. **Ieee-std-1471-2000 recommended practice for architectural description of software-intensive systems.** IEEE, http://standards. ieee. org, v. 12, p. 16-20, 2000.

IONITA, M. T.; HAMMER, D. K.; OBBINK, H. **Scenario-based software architecture evaluation methods:** An overview. ICSE/SARA, Eindhoven (HOL), 2002.

ISO/IEC. **Systems and software engineering.** Software life cycle processes. 12207:2008.

ISO/IEC . **Systems and software engineering.** Architecture description. 42010:2011.

JENDROCK, Eric et al. **The Java EE 7 Tutorial.** Addison-Wesley Professional, 2014.

JOHNSON, Mark. **Designing enterprise applications with the J2EE platform**. Addison-Wesley Professional, 2002.

KAZMAN, Rick et al. **Economics-Driven Software Architecture**: Introduction. 2014.

KAZMAN, Rick; KLEIN, Mark; CLEMENTS, Paul. **ATAM:** Method for architecture evaluation. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 2000.

MARY, Shaw; DAVID, Garlan. **Software Architecture:** Perspectives on an Emerging Discipline. Prentice-Hall, 1996.

PONTES, Danielle Pompeu Noronha. **Evolução de software baseada em avaliação de arquiteturas.** Tese de Doutorado. Universidade de São Paulo, 2012.

RODRIGUES, Herik Zednik et al. **ARQUITETURA LÓGICA DO MODELO E-MATURITY-DESENVOLVIMENTO E FUNCIONAMENTO DO SISTEMA**. In: Proceedings of International Conference on Engineering and Technology Education. 2014. p. 442-446.

SANTOS, Gustavo, et al. **Orion Planning:** Improving Modularization and Checking Consistency on Software Architecture." 3rd IEEE Working Conference on Software Visualization (VISSOFT). 2015.

SERRANO, Maurício; SERRANO, Milene; CAVALCANTE, André Cruz. **Arquitetura de Software de Referência para Sistemas de Informação Governamentais**. 2015.

VERNER, June; SAMPSON, Jennifer; CERPA, Narciso. **What factors lead to software project failure?**. In: Research Challenges in Information Science, 2008. RCIS 2008. Second International Conference on. IEEE, 2008. p. 71-8

VIEIRA, Ramon Santos; REIS, Uedson. **Abordagem dirigida ao domínio aplicado na arquitetura de sistemas web.** Revista de Sistemas e Computação-RSC, v. 4, n. 2, 2015.