Tradução de Trechos do Relatório Técnico CMU/SEI-2006-TR-023

WOJCIK, R.; BACHMANN, F.; BASS, L.; CLEMENTS, P.; MERSON, P.; NORD, R.; and WOOD, W., "Attribute-Driven Design (ADD), Version 2.0," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Report CMU/SEI-2006-TR-023, 2006.

Tradução realizada por Gabriela Teixeira Vieira

2 Visão Global do ADD

O método ADD é uma abordagem para definir uma arquitetura de software na qual o processo do projeto é baseado nos requisitos de atributos de qualidade do software. ADD segue um processo de projeto recursivo que decompõe um sistema ou elementos do sistema através da aplicação de táticas arquiteturais [*Bass* 03] e padrões que satisfazem os seus requisitos guias. Como ilustrado na Figura 1, ADD essencialmente segue um ciclo "Planeje, Faça e Verifique":

- **Planeje**: Atributos de qualidade e restrições de projeto são considerados para selecionar quais tipos de elementos serão usados na arquitetura.
- **Faça**: Elementos são instanciados para satisfazer os requisitos de atributos de qualidade bem como os requisitos funcionais.
- **Verifique**: O projeto resultante é analisado para determinar se os requisitos foram satisfeitos.

Este processo é repetido até que todos os requisitos arquitetonicamente significantes sejam satisfeitos.

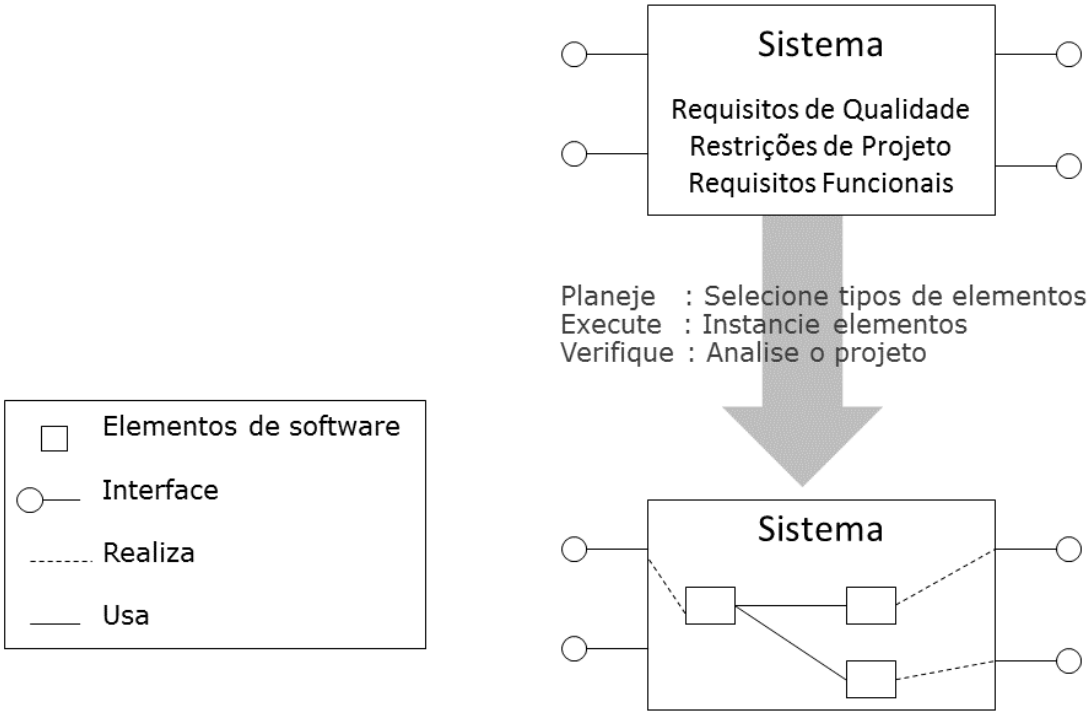

Figura 1: O ciclo de ADD Planeje, Faça e Verifique

A Figura 2 fornece uma visão geral das etapas do ADD. Cada etapa está descrita em detalhes da Seção 4 até a 11, junto com as decisões do projeto feitas a cada etapa.

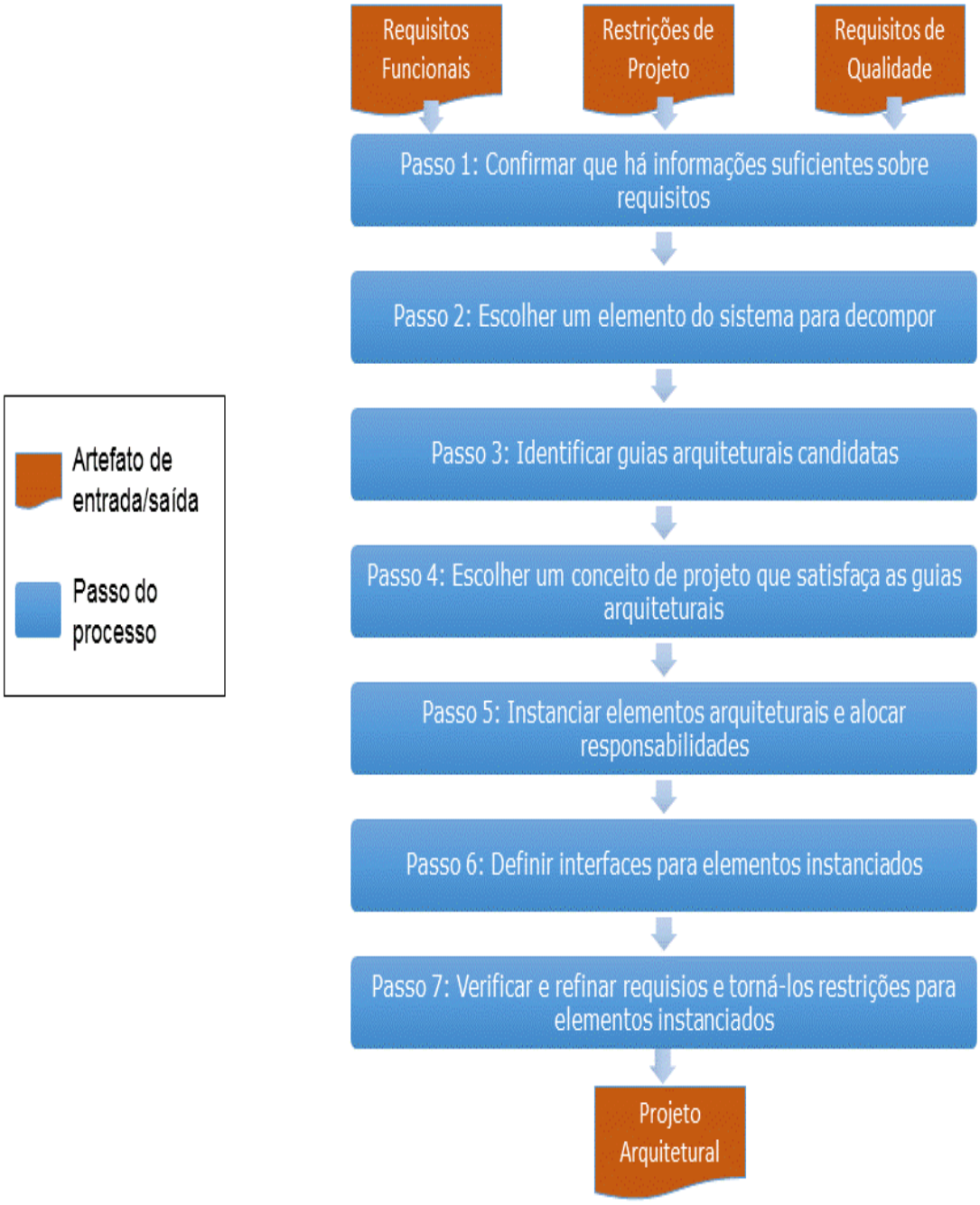

Figura 2: Etapas do ADD

3 Entradas e Saídas do ADD

3.1 ENTRADA DO ADD

Entradas do ADD são requisitos funcionais, restrições de projetos e requisitos de atributos de qualidade que os stakeholders do sistema priorizaram de acordo com o negócio e os objetivos da missão.

Requisitos funcionais especificam quais funções um sistema deve fornecer para conhecer as necessidades dos stakeholders explícitas e implícitas quando o software está sendo usado sob condições específicas [ISO 01]. Por exemplo, aqui está uma lista de exemplos de requisitos funcionais:[1]

• O sistema deve permitir aos usuários comprar e vender Securities.

• O sistema deve permitir aos usuários a avaliar a atividade da conta.

• O sistema deve monitorar e gravar as entradas dos sensores meteorológicos.

• O sistema deve notificar os operadores sobre mudanças de temperatura no núcleo do reator.

• O sistema deve computar e apresentar a órbita e a trajetória para todos os satélites.

Restrições de projeto são decisões sobre um projeto de sistema que devem ser incorporadas em qualquer projeto final do sistema. Representam uma decisão de projeto com um resultado predeterminado. Por exemplo, aqui está uma lista de exemplos de restrições de projeto:

• *Oracle 8.0* deve ser usado para armazenamento persistente.

• Os serviços do sistema devem ser acessíveis através da *World Wide Web.*

• O sistema deve ser implementado usando *Visual Basic.*

• O sistema deve interagir com outros sistemas apenas através do *Publish/Subscribe*.

• O sistema deve ser executado em ambas as plataformas *Windows* e *Unix*.

• O sistema deve integrar com aplicações legadas.

Requisitos de atributos de qualidade são requisitos que indicam os graus aos quais um sistema deve exibir várias propriedades. Por exemplo, aqui está uma lista de exemplos de requisitos de atributos de qualidade:

• Possibilidade de ser construído: o sistema deve ser construído dentro de seis meses.

• Disponibilidade: O sistema deve se recuperar de um acidente de processador dentro de um segundo.

• Portabilidade: O sistema deve permitir que a interface de usuário (UI) seja transferida para uma nova plataforma no prazo de seis meses.

• Performance: O sistema deve processar a entrada de sensor em um minuto.

---

[1] Os exemplos de requisitos funcionais, restrições de projeto e requisitos de atributos de qualidade dados deveriam mostrar o tipo de requisitos que estão sendo discutidos e não são uma recomendação de uma forma específica de

requisitos para usar com o ADD. Na prática requisitos funcionais são frequentemente capturados como casos de uso. Requisitos de atributos de qualidade podem ser capturados como cenários de atributos de qualidade [*Bass* 03].

• Segurança: O sistema deve negar o acesso a usuários não autorizados 100% das vezes.

• Testabilidade: O sistema deve permitir que testes de unidade sejam realizados dentro de três horas com 85% de cobertura de trajeto.

• Usabilidade: O sistema deve permitir aos usuários cancelar uma operação dentro de um minuto.

• Capacidade: O sistema deve ter no máximo 50% da utilização da CPU.

Requisitos funcionais, restrições de projeto e requisitos de atributos de qualidade podem ser implícitos ao invés de explícitos. Por exemplo, aqui estão várias restrições implícitas e requisitos:

• "Dado que todas as transações do sistema estão sujeitas a revisão pela Comissão de Securities e de Troca" implica que o sistema deve manter um log de transações permanentes e geração de relatórios de apoio com base nessas transações.

• "Dado que Joe é o único recurso disponível para lidar com os aspectos de armazenamento persistente do sistema e Joe só tem experiência com o Oracle" implica que Oracle deve ser usado para armazenamento persistente.

• "Dado que a demanda do mercado pelo sistema irá aumentar dramaticamente dentro de oito meses" implica que nosso sistema deve ser construtível dentro de seis meses.

Em alguns casos, pode ser difícil de categorizar uma condição particular tanto como um requisito funcional, como restrição de projeto ou requisito de atributo de qualidade. Por exemplo, aqui estão dois requisitos não categorizados:

- Um requisito de segurança para fornecer autenticação de usuário poderia ser interpretado como um requisito funcional ou um requisito de atributo de qualidade.
- Um requisito para interoperar com um aplicativo de legado particular poderia ser interpretado como um requisito funcional ou uma restrição de projeto.

Felizmente, distinguir entre os três tipos de entradas não é tão importante quanto garantir que estas entradas sejam coletadas antes de começar o ADD.

No restante deste documento, nos referimos as entradas ADD acima coletivamente como "os requisitos". Também assumimos que estamos tratando o arquiteto do sistema através das etapas do método de ADD.

3.2 SAÍDAS ESPERADAS DO ADD

A saída do ADD é um projeto de sistema em termos de papéis, responsabilidades, propriedades e relações entre os elementos do software. Os termos que se seguem são usados ao longo deste documento:

• **elemento de software:** um artefato computacional ou de desenvolvimento que satisfaz vários papéis e responsabilidades, tem propriedades definidas e se relaciona a outros elementos do software para compor a arquitetura de um sistema [*Bass* 03]

• **papel**: um conjunto de responsabilidades relacionadas [*Wirfs-Brock* 03]

• **responsabilidade**: a funcionalidade, dados ou informação que um elemento do software provê

• **propriedade:** informação adicional sobre um elemento do software como nome, tipo, característica de atributo de qualidade, protocolo e assim por diante [*Clements* 03]

• **relacionamento**: uma definição de como dois elementos do software estão associados ou interagem um com o outro

O projeto que resulta do ADD é documentado usando vários tipos de visões arquiteturais, incluindo Módulo, Componente e Conector, e visões de Alocação como apropriados [*Clements* 03]. Em geral, ADD produz uma descrição de arquitetura de software inicial de um conjunto de decisões de projetos para mostrar

1. como fracionar um sistema em grandes elementos computacionais e de desenvolvimento
2. quais elementos serão parte das diferentes estruturas do sistema, o tipo de cada elemento, as propriedades e relações estruturais que eles possuem.
3. quais interações irão ocorrer entre elementos, as propriedades dessas interações e os mecanismos através dos quais elas ocorrem.

   Decisões de projeto específicas em cada uma das categorias acima serão fornecidas na medida em que nós descrevemos as etapas do ADD no restante deste documento.

4 Etapa 1: Confirmar que há Informações Suficientes sobre os Requisitos

4.1 O QUE ESTÁ ENVOLVIDO NA ETAPA 1?

Na primeira etapa você confirma que há informação suficiente sobre os requisitos para prosseguir com o ADD.[2] Na essência, você garante que os *stakeholders* do sistema tenham priorizado os requisitos de acordo com o negócio e os objetivos da missão. Você também deverá confirmar se há informação suficiente sobre os requisitos de qualidade dos atributos para prosseguir.

Como o arquiteto, você usa a lista priorizada de requisitos para determinar em quais elementos do sistema focar durante o projeto.[3] Você considera requisitos e os impactos potencias deles na estrutura da arquitetura em ordem decrescente de importância para os *stakeholders*. Requisitos que não tenham sido priorizados devem ser sinalizados e retornados para os *stakeholders* para classificação.

Além disso, você deve determinar se há informação suficiente sobre os requisitos de atributos de qualidade do sistema. Cada requisito de atributo de qualidade deve ser expressado na forma de "estímulo-resposta", da mesma maneira que os cenários de atributo de qualidade [*Bass* 03]. Cada requisito deve ser descrito como a resposta de um atributo de qualidade mensurável do sistema a um estímulo específico com a explicitação do que se segue:

• fonte de estímulo
• estímulo
• artefato
• ambiente
• resposta
• medida da resposta

Saber esta informação para cada requisito de atributo de qualidade ajuda o arquiteto a selecionar vários padrões de projeto e táticas para alcançar aqueles requisitos. Se a informação acima não estiver disponível para um dado requisito de atributo de qualidade, você deve criar requisitos derivados ou trabalhar com *stakeholders* para clarear os requisitos. Em qualquer evento, cenários de atributos de qualidades podem ser usados para documentar estes requisitos.

Você pode prosseguir com o projeto desde que os requisitos tenham sido priorizados pelos *stakeholders* e que haja informação suficiente em relação a um ou mais requisito de atributos de qualidade. O projeto resultante será suficiente na medida em que os requisitos reunidos até o momento irão influenciar o projeto da arquitetura.

4.2 QUAIS DECISÕES DE PROJETO SÃO FEITAS DURANTE A ETAPA 1?

Nenhuma decisão de projeto será tomada durante esta etapa.

---

[2] Em muitos cenários de desenvolvimento, requisitos plenos e completos não são conhecidos até bem tarde no processo. Requisitos podem estar incompletos quando o ADD começa. ADD irá produzir um projeto arquitetural com base nos requisitos disponíveis no momento. O arquiteto deve indicar decisões no projeto baseadas nos requisitos conhecidos por serem provisórios e estar preparado para voltar e rever o processo com base na melhor entrada.

3 *Stakeholders* devem priorizar os requisitos prévios para a primeira etapa da ADD; este processo de priorização estará fora do âmbito da ADD.

5 Etapa 2: Escolha um Elemento do Sistema para Decompor

5.1 O QUE ENVOLVE A ETAPA 2?

Nesta segunda etapa, você escolhe quais elementos do sistema serão o foco do projeto em etapas subsequentes. Você pode chegar nesta etapa de dois modos:

1. Você alcança a Etapa 2 pela primeira vez como parte de um desenvolvimento "de campo". O único elemento que você pode decompor é o sistema. Por padrão, todos os requisitos são atribuídos a esse sistema.

2. Você está refinando um sistema parcialmente projetado e já visitou a Etapa 2 antes.[4] Neste caso, o sistema foi fracionado em dois ou mais elementos e requisitos foram atribuídos a esses elementos. Você deve escolher um destes elementos como o foco das etapas subsequentes.

No segundo caso, você poderá escolher o elemento sobre o qual focar baseado em uma dessas quatro áreas de interesse:

1. conhecimento atual da arquitetura

− se é o único elemento que você pode escolher (ex. o sistema inteiro ou o último elemento que sobrou)

− o número de dependências que tem com outros elementos do sistema (ex. muitas ou poucas dependências)

2. risco e dificuldade

− o quão difícil seria alcançar os requisitos associados dos elementos

– o quão familiar você está em como atingir os requisitos associados dos elementos

− os riscos envolvidos em alcançar os requisitos associados dos elementos

3. critérios de negócios

− o papel que o elemento desempenha no desenvolvimento incremental do sistema

− o papel que desempenha em versões incrementais de funcionalidade

---

4 O projeto parcial pode ter vindo de repetições prévias do ADD ou de restrições de projeto.

− se será construído, comprado, licenciado ou usado como uma fonte aberta

− o impacto que tem no tempo para o mercado

− se será implementado usando componentes legados

− a disponibilidade de pessoal para tratar um componente

4. Critérios organizacionais

− o impacto que tem sobre a utilização de recursos (ex. recursos humanos e computacionais)

− o nível de habilidade envolvida com o seu desenvolvimento

− o impacto que tem na melhoria de habilidades de desenvolvimento na organização

− alguém com autoridade o selecionou

5.2 QUE DECISÕES DE PROJETO SÃO TOMADAS DURANTE A ETAPA 2?

Nenhuma decisão de projeto é tomada durante esta etapa.

6 Etapa 3: Identificar Candidatos a Guias Arquiteturais

6.1 O QUE ENVOLVE A ETAPA 3?

Neste ponto, você escolheu um elemento do sistema para decompor e os *stakeholders* priorizaram qualquer requisito que afetasse aquele elemento. Durante esta etapa, você irá classificar estes mesmos requisitos pela segunda vez baseado no impacto relativo deles na arquitetura. Esta segunda classificação pode ser tão simples como atribuir "alto impacto", "médio impacto," ou "baixo impacto" a cada requisito.

Dado que os *stakeholders* classificaram os requisitos inicialmente, a segunda classificação baseada no impacto da arquitetura tem o efeito de ordenar parcialmente os requisitos em um número de grupos. Se você usar simples classificações como alto/médio/baixo, os grupos poderiam ser

(A, A) (A, M) (A,B) (M,A) (M,M) (M,B) (B, A) (B,M) (B,B)

A primeira letra em cada grupo indica a importância do requisito para os *stakeholders*. A segunda letra em cada grupo indica o impacto potencial dos requisitos na arquitetura. Requisitos do grupo (A, A) são altamente importantes para os *stakeholders* e espera-se que tenham um alto impacto na arquitetura, e assim por diante. Desses pares, você deve escolher alguns requisitos de alta prioridade (cinco ou seis) como o foco para as etapas subsequentes no processo do projeto.[5]

Os requisitos selecionados são chamados "candidatos a guias arquiteturais" para o elemento a ser decomposto. Análises posteriores podem eliminar alguns candidatos considerados como guias arquiteturais enquanto que outros requisitos podem ser adicionados à lista de candidatos. Por exemplo, embora um requisito esteja classificado como tendo um alto impacto na estrutura da arquitetura, investigações posteriores podem revelar que não tem. Análises também podem revelar que um requisito que não foi considerado como tendo um alto impacto na arquitetura na realidade tem, então é adicionado na lista de candidatos. Em última análise, o nosso objetivo em etapas subsequentes é identificar os verdadeiros guias arquiteturais. A lista dos requisitos que resulta desta etapa pode ou não ter um impacto na estrutura da arquitetura. Aqueles que tiverem serão chamados guias arquiteturais. Antes disso eles são chamados candidatos a guias arquiteturais.

6.2 QUE DECISÕES DE PROJETO SÃO TOMADAS DURANTE A ETAPA 3?

Nenhuma decisão de projeto é tomada durante esta etapa.

---

[5] Se mais de cinco ou seis requisitos são classificados (A, A), esta situação pode significar um risco ao projeto e justifica a renegociação das prioridades. Se uma discussão renovada não é viável, então o arquiteto deverá escolher um punhado de requisitos que acredite que terá o efeito de maior alcance na arquitetura.

7 Etapa 4: Escolha um Conceito de Projeto que Satisfaça os Guias Arquiteturais

7.1 O QUE ENVOLVE A ETAPA 4?

Na Etapa 4, você deverá escolher os principais tipos de elementos que irão aparecer na arquitetura e os tipos de relações entre eles. Restrições de projeto e requisitos de atributo de qualidade (os quais são candidatos a guias arquiteturais) são usados para determinar os tipos de elementos, relações e suas interações.

Como o arquiteto, você deverá seguir estas seis sub etapas:

1. Identificar os interesses/preocupações do projeto que são associadas com os candidatos a guias arquiteturais. Por exemplo, para um requisito de atributo de qualidade, no que se refere a disponibilidade, as principais preocupações/interesses de projeto podem ser a prevenção de falhas, detecção de falhas, e recuperação de falhas [*Bass* 03].

2. Para cada preocupação/interesse de projeto, crie uma lista de padrões alternativos que tratam do mesmo.[6] Padrões na lista são derivados de

− seu conhecimento, habilidades e experiência sobre quais padrões podem ser apropriados

− táticas de arquitetura conhecidas por alcançar atributos de qualidade [*Bass* 03]

Se um candidato a guia de arquitetura diz respeito a mais de um atributo de qualidade, múltiplas táticas podem ser aplicadas.

− outras fontes tais como livros, artigos, materiais de conferência, máquinas de busca, produtos comerciais, e assim por diante

Para cada padrão na sua lista, você deve

a. identificar cada parâmetro discriminador do padrão para ajudá-lo a escolher entre os padrões e táticas na lista. Por exemplo, em qualquer padrão de reinicialização (ex. reinicialização fria ou quente), a quantidade de tempo que leva de uma reinicialização é um parâmetro discriminatório. Para padrões utilizados para alcançar modificabilidade (ex. estratificação), um parâmetro discriminatório é o número de dependências que existem entre elementos no padrão.

b. estimar os valores dos parâmetros discriminatórios

3. Selecionar padrões da lista que você sinta que são mais apropriados para satisfazer os candidatos a guias arquiteturais. Registre a base lógica para as suas seleções. [7] Para decidir quais padrões são apropriados

---

6 Puristas em relação aos padrões podem insistir que um padrão é algo que foi observado "na natureza", pelo menos, três vezes. Nós usamos o termo *padrão* mais livremente e incluímos novas invenções também.

Crie uma matriz (como ilustrada na Tabela 1) com padrões na parte superior e os candidatos a guias de arquitetura listados no lado esquerdo. Use a matriz para analisar as vantagens/desvantagens de aplicar cada padrão para cada candidato a guia arquitetural. Considere o seguinte:

− Quais compensações são esperadas quando estiver usando cada padrão?

− Quão bem os padrões combinam uns com os outros?

− São quaisquer padrões mutuamente exclusivos (ex. você pode usar tanto o padrão A ou o padrão B, mas não os dois)?

Escolha padrões que juntos satisfaçam às guias arquiteturais.

Tabela 1: Estutura da Matriz para Avaliar Padrões de Candidatos

	Padrão 1	**Padrão 1**	**Padrão 2**	**Padrão 2**	**...**	**Padrão n**	**Padrão n**
	Prós	**Contras**	**Prós**	**Contras**		**Prós**	**Contras**
Guia arquitetural 1							
Guia arquitetural 2							
...							
Guia arquitetural n							

4. Considere os padrões identificados até então e decida como eles se relacionam uns com os outros. A combinação dos padrões selecionados resulta em um novo padrão.

a. Decida como os tipos de elementos dos vários padrões estão relacionados.[8]

b. Decida quais tipos de elementos dos vários padrões não estão relacionados.

c. Procure por funcionalidades que se sobreponham e use-as como indicador de como combinar padrões.

d. Identifique novos tipos de elementos que emergem como um resultado de padrões combinados.

e. Revise a lista de decisões de projeto ao final desta seção e confirme que você tomou todas as decisões relevantes.

---

[7] Idealmente, você quer achar um padrão que satisfaz todos os seus candidatos guia; caso contrário, escolha um padrão que chega perto e aumente-o com táticas. Se você não consegue encontrar um padrão que se aproxime, comece com táticas. Estamos usando o termo padrão para descrever tipos de elementos e suas relações·

[8] As relações podem tanto ser relacionadas ao tempo de execução como não. Um exemplo de uma relação de tempo de execução é inserir uma relação "envia os dados para" entre o último filtro num padrão Pipe-e-Filtro e um servidor em um padrão Cliente-Servidor. Um exemplo de uma relação não vinculada ao tempo de execução (neste caso, "é parte de") é atribuir clientes e servidores para camadas em um padrão multi-camada.

5. Descreva os padrões que você selecionou ao começar a capturar diferentes visões arquiteturais, tais como M;odulo, Componente e Conector e Alocação de visões. Você não precisa criar visões de arquitetura totalmente documentadas neste ponto. Documente qualquer informação sobre a qual você esteja confiante ou que você precisa de ponderar sobre a arquitetura (incluindo o que você sabe sobre as propriedades dos vários tipos de elementos). Idealmente, você deve usar modelos de visão para capturar essas informações [*Clements* 03].

6. Avalie e resolva inconsistências no conceito do projeto:

a. Avalie o projeto contra os guias arquiteturais. Se necessário, use modelos, experimentos, simulações, análise formal e métodos de avaliação de arquitetura.

b. Determine se existem quaisquer guias arquiteturais que não foram considerados.

c. Avalie padrões alternativos ou aplique táticas adicionais, se o projeto não satisfaz os guias arquiteturais.

d. Avalie o projeto do elemento atual em oposição ao projeto dos outros elementos da arquitetura e resolva quaisquer inconsistências. Por exemplo, enquanto estiver projetando o elemento atual, você pode descobrir certas propriedades que devem ser difundidas a outros elementos da arquitetura.

7.2 QUE DECISÕES DE PROJETO SÃO TOMADAS DURANTE A ETAPA 4?

O conceito de projeto que você adotou na etapa 4 irá levá-lo a tomar (ou pelo menos considerar) as seguintes decisões de projeto:[9]

• Você decidiu por um conceito de projeto global que inclui os maiores tipos de elementos que irão aparecer na arquitetura e os tipos de relações entre eles.

• Você identificou algumas das funcionalidades associadas com os diferentes tipos de elementos (ex. elementos que *ping* em um padrão *Ping-Echo* irão ter funcionalidade de *ping*).

• Você sabe como e quando tipos específicos de elementos de software mapeiam uns aos outros (ex. estática ou dinamicamente).

• Você pensou sobre comunicação entre os vários tipos de elementos (tanto elementos de software internos como externos) mas talvez adiou decisões sobre eles. Você considerou

− quais tipos de elementos precisam comunicar uns com os outros

− quais classes de mecanismos e protocolos serão usadas para comunicação entre elementos de software e entidades externas (ex. síncronos, assíncronos, acoplamentos híbridos ou chamadas remotas versus chamadas locais)

---

9 Todas estas decisões podem não ser resolvidas pelos conceito de projeto que nós adotamos nesta etapa. Algumas decisões podem ser deferidas a outra etapa, enquanto outras podem ser irrelevantes para o sistema particular sendo desenvolvido.

− as propriedades dos mecanismos requisitadas que irão ser usadas para comunicação entre elementos do software e entidades externas (ex. síncronos, assíncronos, acoplamentos híbridos, rendimento, capacidade da fila e confiabilidade)

- os requisitos de atributos de qualidade associados com os mecanismos de comunicação

− os modelos de dados dos quais a comunicação depende

− os tipos de elementos computacionais suportam os vários usos das categorias do sistema

− como componentes legados e componentes da prateleira (COTS) serão integrados ao projeto

• Você ponderou sobre elementos de software e recursos do sistema, mas talvez adiou decisões sobre eles. Você considerou

- Que recursos são requisitados pelos elementos de software

- Os recursos que precisam ser gerenciados

- Os limites de recursos

- Como os recursos serão gerenciados

- Quais estratégias de agendamento que serão empregadas

- Quais os elementos que **tem estado/não tem estado**

- Os principais modos de operação

Você pensou nas dependências entre os vários tipos de elementos de software internos, mas talvez adiou decisões sobre eles. Você considerou

---

− quais as dependências de execução existem entre elementos

− como e onde as dependências de execução entre elementos são resolvidas

− as dependências de ativação e desativação entre os elementos de software

Você também considerou — e talvez adiou decisões sobre — o seguinte:

− os mecanismos abstratos utilizados

− o que os elementos do sistema conhecem sobre o tempo

− quais modelos de processos/linhas serão empregados

− como requisitos de atributos de qualidade serão abordados

8 Etapa 5: Instanciar Elementos Arquiteturais e Distribuir Responsabilidades

8.1 O QUE ENVOLVE A ETAPA 5?

Na Etapa 5, você instancia os vários tipos de elementos de software que você escolheu na etapa anterior. Aos elementos instanciados são atribuídas responsabilidades de acordo com seus tipos; por exemplo, num padrão de *Ping-Echo,* um elemento do tipo de *ping* tem responsabilidades *Ping* e um elemento do tipo *echo* tem responsabilidades *echo*. Responsabilidades para elementos instanciados também são derivadas dos requisitos funcionais associados com candidatos a guias arquiteturais e os requisitos funcionais associados com o elemento pai. No fim da Etapa 5, cada requisito funcional associado com o elemento pai deve ser representado por uma sequência de responsabilidades dos elementos filho.

Você deve proceder com as seis subetapas que se seguem.

1. Instanciar um exemplo de cada tipo de elemento que você escolheu na etapa 4. Estes casos são referidos como "filhos" ou "elementos filho" do elemento você está decompondo atualmente (isto é, o elemento pai).

2. Atribuir responsabilidades aos elementos filhos de acordo com seu tipo. Por exemplo, aos elementos do tipo *ping* são atribuídas responsabilidades, incluindo a funcionalidade de *ping,* a frequência de *ping*, conteúdo de dados de sinais de *ping* e os elementos para os quais eles enviam sinais de *ping*.

3. Distribuir responsabilidades relacionadas com o elemento pai entre seus filhos de acordo com as propriedades racionais e dos elementos gravados na Etapa 4. Por exemplo, se um elemento pai em um sistema bancário é responsável pela gestão de distribuição de dinheiro, coleta de dinheiro e transações de registros, distribua então essas responsabilidades entre os seus filhos. Note-se que todas as responsabilidades atribuídas ao pai são consideradas, neste momento, independentemente se eles são arquitetonicamente significativos.

Neste ponto, pode ser útil considerar casos de uso que os sistemas geralmente tratam – independentemente se a eles foram dados explicitamente como requisitos. Este exercício pode revelar novas responsabilidades (por exemplo, gestão de recursos). Além disso, você pode descobrir novos tipos de elementos e desejar criar novas instâncias deles. Esses casos de uso incluem

− Um usuário executando duas tarefas simultaneamente

− Dois usuários fazendo tarefas semelhantes simultaneamente

- Inicialização

- Desligamento

- Operação desconectada

- Falha de vários elementos (por exemplo, a rede, o processador, o processo)

− Atualizações de versões

4. Criar instâncias adicionais de tipos de elementos nessas duas circunstâncias:

a. Existe uma diferença nas propriedades de atributos de qualidade das responsabilidades atribuídas a um elemento. Por exemplo, se um elemento filho é responsável pelo recolhimento dos dados dos sensores em tempo real e por transmitir um resumo destes dados em um momento posterior, os requisitos de desempenho associados com a coleta de dados podem levar-nos a instanciar um novo elemento para lidar com a coleta de dados enquanto o elemento original processa a transmissão de um resumo.

b. Você quer alcançar outros requisitos de atributos de qualidade; por exemplo, volta a atribuir a funcionalidade de um elemento a dois elementos para promover modificabilidade. Neste ponto, você deve rever a lista de decisões de projeto listadas no final desta seção e confirmar que você fez todas as decisões pertinentes.

5. Analisar e documentar as decisões de projeto que você fez durante a etapa 5 usando várias visões como estas três:

a. Visões de módulo são úteis para ponderar sobre e documentar as propriedades de um sistema sem tempo de execução (por exemplo, modificabilidade).

b. Visualizações de componentes e conectores são úteis para ponderar sobre e documentar os comportamentos de tempo de execução e propriedades de um sistema (por exemplo, como elementos irão interagir uns com os outros em tempo de execução para cumprir várias exigências e que características de desempenho esses elementos deveriam apresentar).

c. Visualizações de alocação são úteis para ponderar sobre as relações entre software e não software (por exemplo, como elementos de software serão distribuídos aos elementos de hardware).

8.2 QUE DECISÕES DE PROJETO SÃO TOMADAS DURANTE A ETAPA 5?

Algumas das decisões podem não ser relevantes para o seu tipo específico de sistema. No entanto, suas decisões provavelmente envolverão vários dos seguintes procedimentos:

- quantos elementos de cada tipo serão instanciados e quais propriedades individuais e relações estruturais que eles possuirão
- quais elementos computacionais serão usados para apoiar as diversas categorias de uso do sistema
- quais os elementos irão apoiar os principais modos de operação
- como os requisitos de atributos de qualidade foram cumpridos dentro da infra-estrutura e aplicações
- como a funcionalidade é dividida e atribuída a elementos de software, incluindo como funcionalidade é distribuída através da infra-estrutura e aplicações
- como elementos de software mapeiam uns aos outros

− como elementos do sistema em diferentes estruturas arquitetônicas mapeiam uns aos outros (ex. como módulos mapeiam elementos de tempo de execução e como elementos de tempo de execução mapeiam processadores)

− se o mapeamento de um elemento do sistema ao outro é estático ou dinâmico (ex. quando o mapeamento é determinado – no tempo de construção, implantação, tempo de carregamento ou tempo de execução)

• comunicação entre os vários elementos, tanto elementos internos do software como entidades externas

− quais elementos do software precisam comunicar uns com os outros

− que mecanismos e protocolos serão usados para a comunicação entre os elementos do software e entidades externas

− As propriedades requisitadas dos mecanismos que serão usadas para comunicação entre elementos do software e entidades externas (ex. síncronos, assíncronos e acoplamento híbrido)

− os requisitos de atributos de qualidade associados com os mecanismos de comunicação

– os modelos de dados dos quais a comunicação depende

− quais elementos computacionais suportam as várias categorias de uso do sistema

− como componentes legados e de COTS serão integrados ao projeto

• elementos internos de software e recursos do sistema

− quais recursos são requisitados pelos elementos de software

− quais recursos precisam ser gerenciados

− os limites dos recursos

− como os recursos serão gerenciados

- que estratégias de agendamento serão empregados

- quais os elementos que tem estado/não tem estado

- os principais modos de operação

• dependências entre os elementos de software internos

- quais dependências de execução existem entre os elementos

- como e onde as dependências de execução entre os elementos são resolvidas

- as dependências de ativação e desativação entre elementos de software

• quais mecanismos de abstração são usados

• quanto sabem os elementos do sistema sobre o tempo

• quais modelos de processos/linhas serão empregados

• como os requisitos de atributos de qualidade serão abordados

9 Etapa 6: Definir Interfaces para Elementos Instanciados

9.1 O QUE ENVOLVE A ETAPA 6?

Na etapa 6, você define os serviços e as propriedades requisitadas e fornecidas pelos elementos do software no nosso projeto. No ADD, estes serviços e propriedades são referidos como interface do elemento. Note que uma interface não é simplesmente uma lista de assinaturas de operação. Interfaces descrevem as suposições de PROVISÕES e REQUERIMENTOS que elementos de software fazem uns sobre os outros. Uma interface pode incluir qualquer um dos seguintes:

• sintaxe das operações (ex. assinatura)

• semântica das operações (ex. descrição, pré- e pós-condições, restrições)

• troca de informação (ex. eventos assinalados, dados globais)

• requisitos de atributos de qualidades de elementos individuais ou operações

• manipulação de erros

Você deve proceder com estas três etapas:

1. Exercer os requisitos funcionais que envolvem os elementos que você instanciou na Etapa 5.

2. Observe qualquer informação que seja produzida por um elemento e consumida por outro. Considere as interfaces da perspectiva de diferentes visões. Por exemplo, uma visão de Módulo irá permitir que você pondere sobre o fluxo da informação; uma visão Concorrente irá permitir que você pondere sobre desempenho e disponibilidade; e uma visão de Implementação irá permitir que você pondere sobre segurança e disponibilidade.

3. Registe os seus achados na documentação da interface para cada elemento.

9.2 QUAIS DECISÕES DE PROJETO SÃO TOMADAS DURANTE A ETAPA 6?

Algumas das decisões podem não ser relevantes para o seu tipo particular de sistema. De qualquer modo, as suas decisões provavelmente irão envolver vários dos seguintes aspectos:

• as interfaces externas ao sistema

• as interfaces entre as partições do sistema de alto nível

• as interfaces entre aplicações dentro de partições do sistema de alto nível

• as interfaces com a infraestrutura

10 Etapa 7: Verificar e Refinar Requisitos e Torná-los Restrições para Elementos Instanciados

10.1 O QUE ENVOLVE A ETAPA 7?

Na etapa 7, você verifica que a decomposição do elemento até agora atende aos requisitos funcionais, aos requisitos de atributos de qualidade e as restrições de projeto. Você também prepara elementos filhos para posterior decomposição.

Você deve proceder com estas três subetapas:

1.Verifique se todos requisitos funcionais, requisitos de atributo de qualidade e restrições de projeto atribuídos ao elemento pai foram distribuídos a um ou mais elementos filho na decomposição.

2. Traduza quaisquer responsabilidades que foram atribuídas aos elementos filho em requisitos funcionais para os elementos individuais.

3. Refine os requisitos de atributo de qualidade para elementos filho individuais conforme for necessário.

10.2 QUE DECISÕES DE PROJETO SÃO TOMADAS DURANTE A ETAPA 7?

Nenhuma decisão é tomada durante esta etapa.

11 Etapa 8: Repita as Etapas 2 a 7 para o Próximo Elemento do Sistema que Você Quiser Decompor

Uma vez que você completou as etapas de 1–7, você tem uma decomposição do elemento pai em elementos filhos. Cada elemento filho é uma coleção de responsabilidades, cada um tendo uma descrição de interface, requisitos funcionais, requisitos de atributo de qualidade, e restrições de projeto. Agora você pode retornar ao processo de decomposição na Etapa 2 onde você selecionará o próximo elemento para decompor.

12 Resumo

Este relatório revisa o método ADD para tornar o método mais fácil para os profissionais o aprenderem, compreenderem e aplicarem. O método tem sido utilizado com sucesso antes destas revisões. No entanto, esperamos que nosso trabalho ofereça uma compreensão mais profunda para aqueles que já o aplicaram e torne o método mais acessível àqueles que ainda não o aplicaram. ADD é um método poderoso para projetos de arquitetura que se distingue de outros métodos de projeto porque se concentra na decomposição de um sistema de uma perspectiva de qualidade dos atributos. ADD pode ser usado em combinação com outros métodos SEI, tais como a Oficina de Atributos de Qualidade para levantamento de requisitos de entrada em ADD ou o *Architecture Tradeoff Analysis Method* (ATAM®) para avaliar arquiteturas que resultam da aplicação do ADD. ADD também pode ser usado ou adaptado a maioria dos ciclos de vida ou processos (ex. evolucionário, cachoeira, espiral, *Rational Unified Process [RUP],* ou de desenvolvimento ágil). Embora consideremos esse relatório como uma grande revisão ao ADD, nossa intenção é continuar vigiando e melhorando o método na medida em que aprendemos suas forças e fraquezas através de nossas próprias aplicações, bem como através de feedback que recebemos de outros profissionais. Nós somos gratos a aqueles que nos contaram suas experiências usando o método e comentários e sugestões sobre este documento e o uso do método serão bem vindos para ajudar em nossa próxima revisão.