3
Arquitetura para a Coordenação e a Composição de Artefatos de Software

Resumo

Este capítulo apresenta a arquitetura ACCA, que é a parte central deste trabalho. A arquitetura separa os conceitos de coordenação, composição e artefatos de software em três níveis de abstração estruturados em camadas a partir do padrão arquitetural Layer (Buschman, 1996). Após apresentar a arquitetura, são descritas abstrações que podem ser úteis ao entendimento de suas camadas. Finalmente, discute-se uma avaliação conceitual desta proposta.

3.1.
A Abordagem Proposta

Neste trabalho é proposta a separação dos conceitos de coordenação, composição e de artefatos ou componentes de software em uma Arquitetura para a Coordenação e a Composição de Artefatos de Software (ACCA). Esta arquitetura estrutura estes conceitos em camadas, seguindo as diretrizes estabelecidas pelo padrão arquitetural *Layer* (Buschman, 1996), onde cada camada representa um nível de abstração.

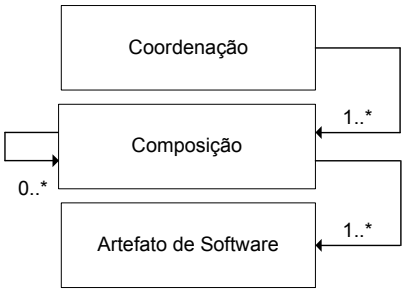

Figura 12 – Arquitetura / Coordenação / Composição / Artefato de Software (ACCA)

Ao estruturar o desenvolvimento de uma solução com ACCA, espera-se que questões específicas dos conceitos escolhidos sejam tratadas em uma única camada, evitando assim que ocorram modificações intrusivas nos artefatos de software já existentes ou mesmo em construções de outras camadas.

Na camada de Artefatos de Software estão presentes as abstrações de mais baixo nível desta arquitetura: os componentes de software. Na camada de Composição são reunidos em uma composição um ou mais artefatos, ou ainda são utilizadas zero ou mais composições em conjunto. As dependências entre uma ou mais composições são elaboradas na camada de Coordenação, representando nesta versão de ACCA, o maior nível de abstração disponível para o trabalho de um engenheiro de software. Todas as três camadas da arquitetura ACCA (Figura 12) são sucintamente descritas abaixo.

Todos os componentes de software disponíveis estão presentes na camada Artefato de Software. Artefatos são disponibilizados de forma independente, sendo passíveis de composição por terceiras partes. A princípio, artefatos podem ser desenvolvidos utilizando-se diferentes tecnologias e paradigmas. Esta camada pode ser entendida como um grande repositório de componentes que oferece serviços para a catalogação, a busca e a alteração de artefatos de software.

A camada Composição é uma abstração para o processo recursivo de reunião e utilização de um conjunto de construções da camada inferior, os artefatos de software. Esta camada é responsável por abstrair detalhes necessários à integração e à convivência de diferentes tecnologias em uma única solução. Esta abstração torna-se concreta com a realização destas atividades de integração em padrões de composição. Atividades de integração correspondem ao processo de implementação e adaptação de protocolos de transporte e protocolos de comunicação necessários à promoção da interoperabilidade entre os componentes. Nesta camada de composição, além do fluxo de conversação, é preciso garantir que ocorra o entendimento semântico das informações que são trocadas entre os artefatos de software em uma solução. É recomendado que o engenheiro de software aplique técnicas para a reunião e a avaliação de composições de artefatos de software.

Finalmente, a camada Coordenação oferece a abstração necessária ao desenvolvimento de cenários de uso de componentes de software. Para elaborar um cenário de uso é preciso analisar: (1) as restrições e contextos impostos por requisitos funcionais e não funcionais, e (2) o conjunto de construções disponibilizadas pela camada Composição. Após confrontar as alternativas disponibilizadas por (2) com as necessidades levantadas por (1), é possível identificar a necessidade de desenvolver novas composições ou novos

componentes, ou ainda prescrever o comportamento da solução em função das unidades já existentes. Nesta camada, o engenheiro de software deve aplicar técnicas para auxiliar a análise, o desenvolvimento, a configuração e a simulação de cenários de uso de composições de artefatos de software.

3.2. Abstração para a Camada de Artefatos de Software de ACCA

Na camada Artefatos de Software estão presentes todos os componentes de software disponíveis para reutilização. Por questões de organização, é necessário que estes componentes estejam armazenados em um ou mais repositórios (Figura 13). Estes repositórios são utilizados para organizar e facilitar o processo de armazenamento, atualização, busca e seleção de componentes de software.

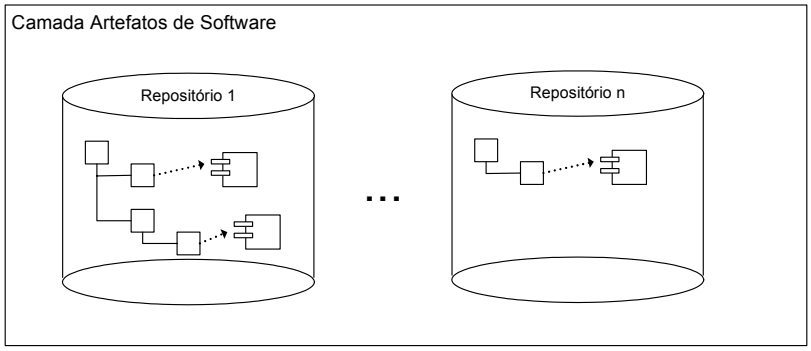

Figura 13 – Camada de artefatos de software de ACCA: uma abstração

Em Sametinger (1997), repositórios de componentes são caracterizados como abstrações capazes de armazenar e recuperar componentes de software. É importante que as informações associadas ao componente estejam documentadas de alguma forma em catálogos associados a estes repositórios. Além disto, esta documentação deve ser suficiente para caracterizar o componente e distinguí-lo dos demais. A partir da análise destas informações, espera-se que a busca por unidades de software seja mais precisa. Nesta abordagem, atributos não funcionais também podem ser associados à semântica do componente, tais como confiabilidade, desempenho, disponibilidade, etc. (Merad, 1999).

Ainda segundo Sametinger (1997), existem diversos tipos de repositórios que podem ser utilizados para auxiliar a realização desta camada de ACCA, dentre eles: repositórios locais, repositórios distribuídos, repositórios específicos a um domínio e repositórios de referência.

Além de estudar a aplicação de diferentes tipos de repositórios de componentes, uma questão importante a considerar é como são feitas a busca e a

seleção de componentes armazenados. A eficiência deste processo está relacionada à qualidade dos mecanismos de organização, de classificação e de estruturação das informações a respeito dos componentes. Vários métodos de classificação de componentes já foram descritos na literatura (Barros, 1995), dentre eles: texto livre, classificação por palavras-chave, classificação enumerada (Sametinger, 1997), facetas e pares de atributos (Prieto-Diaz,1991).

Ainda existe muito trabalho a ser feito com relação à busca e à seleção de componentes. Ao analisar os resultados de um estudo empírico (Poulin, 1995) com alguns métodos de classificação existentes, pode-se concluir que nenhum dos métodos é, em absoluto, vantajoso em relação aos demais. Portanto, para realizar estes métodos em ACCA, deve-se considerar uma abordagem que utilize, conjuntamente, técnicas consagradas de busca por componentes e técnicas que se mostram promissoras como a busca por serviços na *web*, a aplicação de técnicas de aprendizado, dentre outras.

3.3. Abstração para a Camada de Composição de ACCA

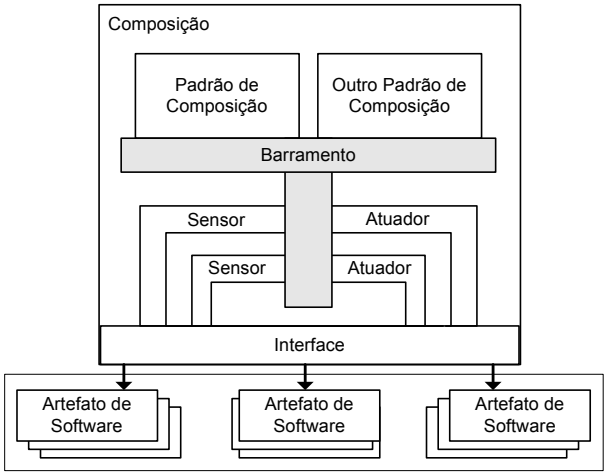

Figura 14 – Camada de composição de ACCA : uma abstração

Sommerville (2001) identificou diversas abstrações utilizadas para classificar um conjunto de componentes funcionais. Ao relacionar algumas destas abstrações, foi possível criar uma entidade abstrata para representar as unidades de software relevantes na camada de Composição de ACCA (Figura 14). Além destas abstrações, foi incluído o conceito de padrão de composição para representar a estrutura lógica que agrega as entidades necessárias às atividades de integração dos artefatos de software em uma solução. Padrões de composição

compreendem a estrutura necessária para a realização dos diferentes modelos de composição. Conforme exemplos apresentados no capítulo anterior, para a realização desta abstração existem diferentes propostas que podem ser estudadas, projetadas e aplicadas, isoladas ou em conjunto, ao tratamento adequado da integração de software.

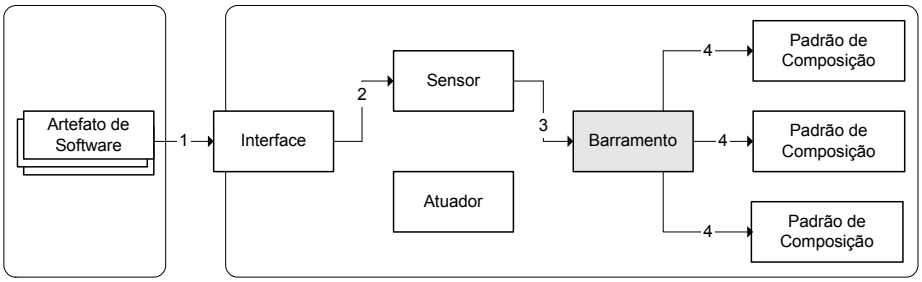

Figura 15 – Estímulo a uma composição : fluxo de mensagens

O mecanismo proposto nesta seção compreende seis entidades (Figura 14): interface, sensores, atuadores, barramento, padrões de composição e artefatos de software externos a camada de composição. A realização destas abstrações corresponde à implementação das atividades de integração necessárias à interconexão entre os artefatos. A interface é a entidade responsável pela troca de dados com componentes externos ao sistema. Sensores são responsáveis por capturar estímulos (Figura 15), respostas de componentes ou requisições de processos, e repassá-las a um padrão de composição. Os atuadores são responsáveis por gerar estímulos a interface, que invocará os serviços disponibilizados pelos diferentes artefatos (Figura 16), a partir de instruções geradas por um padrão de composição. Uma camada única de comunicação, representada pelo barramento, permite que o fluxo e a distribuição de informações entre todos os componentes internos ocorra de forma homogênea.

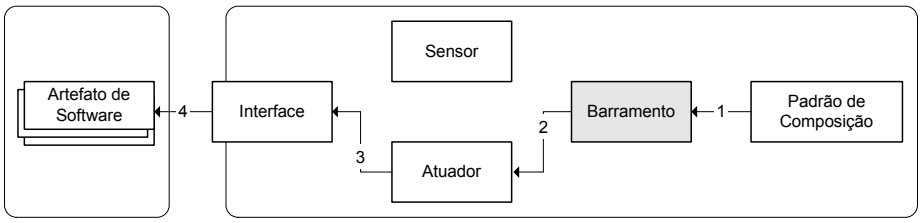

Figura 16 – Invocação de um serviço: fluxo de mensagens

Nesta estrutura (Figura 14), a implementação do protocolo de transporte necessário à conexão entre os componentes fica a cargo da interface de uma composição. A implementação do protocolo de comunicação e das conversões,

necessárias ao entendimento semântico das informações, são responsabilidades dos sensores e atuadores.

Estas abstrações (Figura 14) forem utilizadas para auxiliar o entendimento de questões relevantes a um processo de composição, e foram utilizadas para orientar a geração do framework de composição que realiza esta camada.

3.4. Abstração para a Camada de Coordenação de ACCA

O papel principal desta camada é permitir que seja feito, sistematicamente, um processo de identificação e análise de uma grande variedade de interdependências entre composições de software. O conjunto geral de interdependências passíveis de configuração corresponde justamente ao modelo de coordenação aplicado à resolução do problema, e é projetado por meio de um padrão de coordenação (Figura 17).

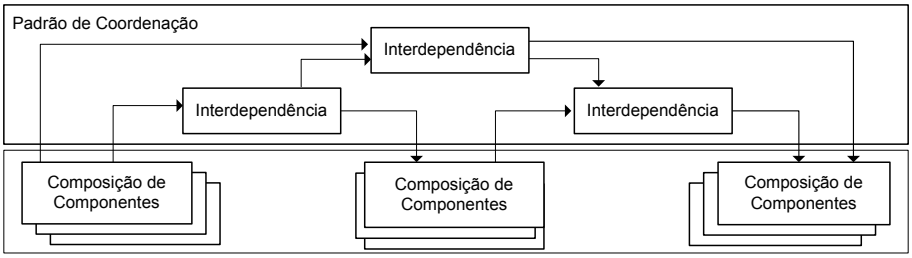

Figura 17 – Padrão de coordenação: interdependências entre composições

A representação explícita de padrões de coordenação possibilita a organização de interdependências entre as composições existentes (Figura 17). Como exemplos de interdependências pode-se citar a ordem em que as requisições a composições são tratadas, a possibilidade de execução em paralelo de um conjunto de composições, ou qualquer restrição que precise ser imposta por uma política de coordenação.

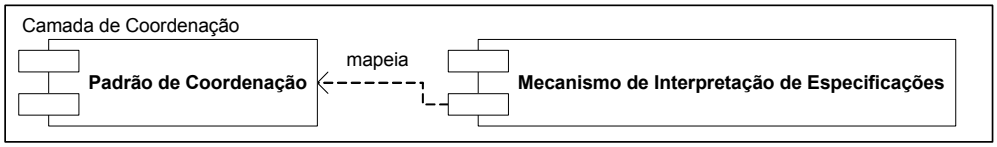

Figura 18 – Camada de coordenação de ACCA: uma abstração

A versão atual da camada Coordenação se baseia no uso de especificações prescritivas e prevê duas subunidades (Figura 18): (1) Padrão de Coordenação e (2) Mecanismo de Interpretação de Especificações. O padrão de coordenação é responsável por executar as instruções prescritas na especificação e interpretadas por (2). Associado ao padrão existe um modelo de coordenação que disponibiliza

todas as características e restrições que podem ser impostas e utilizadas em uma solução.

Especificações prescritivas são instruções ordenadas, formalmente definidas, e suficientes para indicar o comportamento ou configuração que determinado dispositivo terá durante sua execução.

Um exemplo de especificação prescritiva, que pode ser utilizado nesta camada, é a proposta de contratos de coordenação (Lano et al., 2002). As especificações contidas em um contrato podem explicitar, em um nível adequado de abstração, o tipo de componente, sua natureza, suas funcionalidades e restrições de uso, ou principalmente prescrever os seus cenários de reutilização. Desta forma, além de documentar a solução, os contratos podem ser utilizados como entrada para um mapeamento contrato → mecanismo de execução. Este mecanismo, a partir destas prescrições, é responsável por construir parte das regras de negócios adequadas[3] para a execução das unidades.

3.5. Avaliação Conceitual de ACCA

Existem diversos pontos que precisam ser considerados ao utilizar esta proposta. Com relação aos pontos positivos pode-se citar: a possibilidade de reutilização das camadas propostas em outros contextos, a possibilidade de se estabelecer um padrão para o desenvolvimento de software baseado em componentes e a possibilidade de evolução das camadas.

Reutilização de camadas em outros contextos: é possível reutilizar propostas concretas para a realização das camadas de ACCA em outros contextos de desenvolvimento. Um exemplo desta reutilização é a aplicação isolada do *framework* de composição a ser apresentado no capítulo sobre a realização de ACCA.

Suporte à padronização: a partir de uma estrutura como ACCA, é possível propor diferentes padrões para as interfaces e para as tarefas oferecidas por cada camada. A partir destas interfaces padronizadas, é possível manter os relacionamentos entre as construções localizados e confinados a um único conceito. Este tipo de abordagem pode auxiliar a portabilidade de um sistema, o

[3] Prescritas de acordo com a especificação.

desenvolvimento de diferentes propostas comerciais para a realização dos conceitos, e o desenvolvimento de arcabouços de testes para validar camadas desenvolvidas individualmente.

Possibilidade de evolução de camadas: é possível que implementações de camadas individuais sejam substituídas por implementações semanticamente equivalentes (Buschman, 1996).

Existem outros pontos que precisam ser considerados e atentamente observados, pois eles podem ter uma influência negativa na utilização de ACCA. Dentre eles: a possibilidade de encadeamento de conceitos instáveis, uma possível perda da eficiência ou desempenho da execução de aplicações e a dificuldade para o estabelecimento da granularidade correta ao se estruturar as camadas da proposta.

Encadeamento de conceitos instáveis: suponha um domínio onde os artefatos de software são, por demasia, voláteis e vulneráveis a alterações. Alterações em sua estrutura podem ter um impacto severo sobre as demais camadas, por este ser o conceito menos abstrato. Além disto, construções de camadas superiores utilizam intensamente construções desta camada. Por isto, recomenda-se a utilização de ACCA em domínios em que artefatos de software sejam as abstrações estáveis, seguidas das composições de software e por fim, pelo processo de explicitar as interdependências ou coordenar as composições, que são os conceitos menos estáveis de uma solução, portanto mais sujeitos a alterações.

Perda de eficiência: uma arquitetura em camadas é geralmente menos eficiente que uma estrutura monolítica (Buschman, 1996). Se os serviços das camadas de alto nível dependem intensamente de serviços de camadas inferiores, pode haver um excesso de redirecionamentos, transformações ou chamadas de serviços o que implica em um desempenho não otimizado da execução de soluções.

Dificuldade de se estabelecer a granularidade correta das camadas: como a arquitetura ACCA propõe somente três camadas para o desenvolvimento de aplicações, nem todas as abstrações existentes no desenvolvimento de software são detalhadas com esta abordagem. Em contrapartida, a utilização de muitos conceitos estruturados poderia comprometer o entendimento da abordagem e aumentar a complexidade associada à sua utilização.