Mecanismos Arquiteturais

Marco Mendes
masmendes@pucminas.br
Junho de 2021

1 MECANISMOS ARQUITETURAIS

Um mecanismo arquitetural é uma solução comum para um problema frequentemente encontrado. Exemplos incluem a autenticação de sistemas, a persistência de dados ou a interoperabilidade entre aplicações através de protocolo HTTP.

Mecanismos arquiteturais são usados para satisfazer requisitos arquitetonicamente significativos. Normalmente, esses são requisitos não funcionais, como problemas de desempenho e segurança. Quando totalmente descritos, os Mecanismos mostram padrões de estrutura e comportamento no software. Eles formam a base de software comum que será aplicado consistentemente em todo o produto que está sendo desenvolvido. Eles também formam a base para padronizar a maneira como o software funciona; portanto, são um elemento importante da arquitetura geral do software. A definição de mecanismos de arquitetura também permite decisões sobre se os componentes de software existentes podem ser aproveitados para fornecer o comportamento necessário; ou se o novo software deve ser comprado ou construído.

O valor na definição de mecanismos de arquitetura é que eles:

1. Invocam explicitamente aspectos da mecânica da solução que são comuns em todo o sistema. Isso ajuda você a planejar.
2. Colocam marcadores para os desenvolvedores construírem esses aspectos do sistema uma vez e depois os reutilizem. Isso reduz a carga de trabalho.
3. Promovem o desenvolvimento de um conjunto consistente de serviços. Isso torna o sistema mais fácil de manter.

Um Mecanismo Arquitetural pode estar em três estados: Análise, Desenho ou Implementação. Essas categorias refletem a maturidade da descrição do mecanismo.

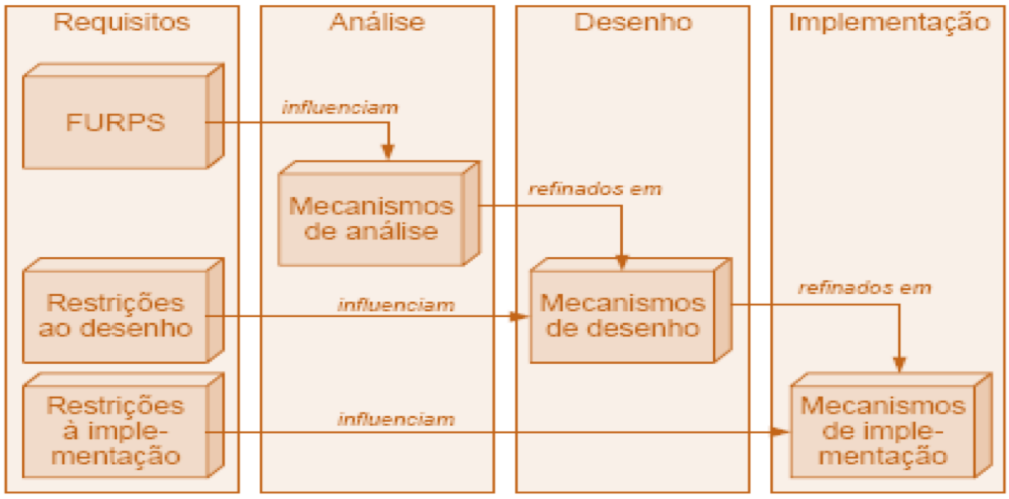
O estado muda à medida que níveis sucessivos de detalhe são descobertos durante o momento em que você refina os seus requisitos.

Esses níveis são descritos abaixo.

- Mecanismos no Estado de Análise: Uma solução conceitual para um problema técnico comum. Por exemplo, a persistência é uma solução abstrata para o requisito comum de armazenar dados. O objetivo desta categoria é simplesmente identificar a necessidade de um Mecanismo Arquitetônico ser projetado e implementado; e capturar atributos básicos para esse mecanismo.
- Mecanismo no Estado de Desenho: Um refinamento de um Mecanismo de Análise em uma tecnologia concreta (por exemplo, RDBMS). O objetivo desta categoria é orientar a seleção precisa de produtos ou tecnologia.
- Mecanismo no Estado de Implementação: Um refinamento adicional de um mecanismo de design para uma especificação para o software. Isso pode ser

apresentado como um padrão de design ou código de exemplo ou mesmo uma solução técnica. Para um RDBMS, por exemplo, dois exemplos de mecanismos de implementação seriam o Oracle DB ou o MySQL Server.

Mecanismos fornecem uma ponte natural entre os requisitos arquiteturais e a solução técnica escolhida dentro de um projeto. A figura abaixo resume isso.



2 MECANISMOS DE ANÁLISE

Um mecanismo de análise, segundo (Eeles, 2005), define uma solução independente de implementação. Um exemplo de um mecanismo de análise é a persistência. Um outro exemplo seria a comunicação entre sistemas (interoperabilidade).

Os mecanismos de análise mostram padrões de estrutura e padrões de comportamento em um sistema. Eles são utilizados durante a análise para reduzir a sua complexidade e aprimorar sua consistência, fornecendo aos arquitetos uma representação abreviada de um comportamento complexo.

Os mecanismos permitem que o esforço de análise enfatize a conversão dos requisitos arquiteturais em conceitos de software, sem se aprofundar na especificação do comportamento relativamente complexo, que é necessário para suportar a funcionalidade, mas não é fundamental para o mesmo. Os mecanismos de análise frequentemente resultam da instanciação de um ou mais padrões de arquitetura ou de análise.

3 MECANISMOS DE DESENHO

Um mecanismo de design é um aperfeiçoamento de um mecanismo de análise correspondente. Um mecanismo de design acrescenta detalhes concretos ao mecanismo de análise conceitual, mas quase não exige tecnologia específica. Como nos mecanismos de análise, um mecanismo de design pode instanciar um ou mais padrões, neste caso, padrões arquiteturais o de design.

Como exemplo, o uso de um banco de dados relacional é um mecanismo de desenho para o mecanismo de análise persistência. O uso de filas de mensagens é um exemplo de um mecanismo de desenho para o mecanismo de análise comunicação.

Se observarmos as propostas Java EE e .NET iremos perceber que estas "soluções" procuram responder a diversos mecanismos de análise. Na arquitetura Java EE, por exemplo, o mecanismo arquitetural transação é endereçado através da especificação JTA enquanto o mecanismo de segurança é endereçado através da especificação JAAS.

4 MECANISMOS DE IMPLEMENTAÇÃO

Da mesma forma, um mecanismo de implementação é um refinamento de um mecanismo de design correspondente, utilizando, por exemplo, uma linguagem de programação específica e outra tecnologia de implementação (como um produto de middleware do fornecedor específico). Um mecanismo de implementação pode instanciar um ou mais *idiomas* (Idioms) ou padrões de implementação.

Por exemplo, o uso do banco de dados Oracle é um refinamento do mecanismo de desenho de persistência. O uso da solução Microsoft Message Queue é um refinamento do mecanismo de desenho filas de mensagens.

5 EXEMPLOS DE MECANISMOS COMUNS

Mecanismo Arquitetural	**Descrição**
Disponibilidade	A porcentagem de tempo que o sistema deve estar disponível para uso, incluindo interrupções planejadas.
Arquivamento	Fornece um meio de mover dados do armazenamento ativo quando atingir um estado específico.
Auditoria	Fornece trilhas de auditoria da execução do sistema.
Comunicações	Um mecanismo para lidar com a comunicação entre processos.
Depuração	Fornece elementos para suportar depuração de aplicativos.
Recuperação de desastres	Fornece recursos para recuperar sistemas, aplicativos, dados e redes.
Gerenciamento de Erros	Permite que erros sejam detectados, propagados e relatados.

Gerenciamento de Eventos	Suporta o uso de eventos assíncronos dentro de um sistema.
Gráficos	Suporta serviços de interface do usuário, como renderização 3D.
Troca de Informações	Suporta intercâmbio de informações através dos limites técnicos e organizacionais com traduções semânticas e de formato apropriadas.
Licença	Fornece serviços para aquisição, instalação, rastreamento e monitoramento do uso da licença. Pode ser necessário como parte dos órgãos corporativos autorizadores.
Localização / Internacionalização	Fornece recursos para suportar vários idiomas humanos e renderizar o idioma preferido pelo usuário.
Correio	Serviços que permitem que os aplicativos enviem e recebam correio eletrônico.
Volumetria	Suporte para lidar com quantidades muito grandes de dados.
Gerenciamento de Memória	Serviços para abstrair como a memória é alocada e liberada.
Metadados	Suporta a introspecção em tempo de execução de componentes e dados.
Ajuda online	Fornece capacidade de ajuda online
Persistência	Serviços para lidar com a leitura e gravação de dados armazenados.
Impressão	Fornece recursos para interface com impressoras.
Gerenciamento de Processos	Fornece suporte para o gerenciamento de processos e threads.
Relatórios	Fornece recursos flexíveis de relatórios
Gerenciamento de Recursos	Fornece suporte para o gerenciamento de recursos caros, como conexões de banco de dados.
Agendamento	Fornece a capacidade de executar tarefas em um horário especificado.
Segurança	Fornece serviços para proteger o acesso a determinados recursos ou informações.
Gerenciamento do Sistema	Serviços que facilitam o gerenciamento de aplicativos em um ambiente operacional.
Tempo	Serviços para sincronizar a hora em uma rede e traduzir horários em fusos horários diferentes.
Gerenciamento de Transações	Um mecanismo para lidar com transações ACID.
Fluxo de trabalho	Suporte para a movimentação de documentos e outros itens de trabalho, geralmente através de uma organização.

6 REFERÊNCIAS ADICIONAIS

Eeles, P. (2005). Capturing architectural requirements. *IBM Rational developer works*.