

Padrões de Design de Serviços

João Marcos Ferreira¹, Kleonte Gomes², Rafael Tavares Rufino³

¹Instituto Federal de Educação, Ciência e Tecnologia da Paraíba - Campus Cajazeiras (IFPB)
Cajazeiras – PB – Brazil

joaomarcos.ads@gmail.com, kleontegomes@gmail.com, rafaeltavares.rofino198@gmail.com

Abstract. *O objetivo desse artigo é apresentar os principais conceitos de uma Arquitetura orientada a serviços (SOA) e seus princípios, bem como padrões de serviços fundamentais apresentados por Thomas Erl em seu livro [Erl 2008], que são aplicados na solução de problemas comuns encontrados no desenvolvimento de sistemas baseados em SOA. A união de princípios e padrões de design, formam um conjunto de boas práticas essencial a qualquer projeto.*

Resumo. *The objective of this paper is to present the main concepts of a Service-Oriented Architecture (SOA) and its principles, well as Foundational Service Patterns presented by Thomas Erl in his book [Erl 2008], which are applied in solving common problems encountered in the development of systems based on SOA. The union of principles and design patterns, form a core set of good practices to any project.*

1. O Que é SOA?

Arquitetura orientada a serviços ou SOA (Service-Oriented Architecture) é descrita de forma diferente por vários autores, alguns o vêem como um estilo técnico de arquitetura que fornece os meios para integrar sistemas distintos e expor as funções de negócios reutilizáveis. Outros autores, no entanto, ter uma visão muito mais ampla:

SOA é uma abordagem arquitetural corporativa que permite a criação de serviços de negócio interoperáveis que podem facilmente ser reutilizados e compartilhados entre aplicações e empresas.

A arquitetura orientada a serviços é um estilo de design que orienta todos os aspectos da criação e utilização de serviços de negócios em todo o seu ciclo de vida (desde a concepção à aposentadoria). [Newcomer and Lomow 2005]

Arquitetura Orientada a Serviços (SOA) é um paradigma para organização e utilização de capacidades distribuídas que podem estar sob controle de diferentes domínios proprietários. [OASIS 2006]

De maneira simples, SOA é uma abordagem de negócios para criar sistemas de TI (Tecnologia de Informação) que permitem alavancar recursos existentes, criar novos recursos e, principalmente, estar preparado para inevitáveis alterações exigidas pelo mercado, obtendo mais produtividade e lucro para a empresa.

2. Princípios de Design

Os Princípios de design são uma representação do paradigma de design orientado a serviços, eles estipulam boas praticas para construção de um sistema baseado em SOA bem implementado. A seguir estão os princípios de design de serviços listados por

- Service Abstraction
- Service Autonomy
- Service Composability
- Service Discoverability
- Service Loose Coupling
- Service Reusability
- Service Statelessness
- Service-Oriented and Interoperability
- Standardized Service Contract

Especificamente, a potencial relação entre princípios e padrões de projeto orientados a serviço pode ser resumido da seguinte forma:

Princípios de design são aplicados em conjunto para uma solução lógica, a fim de moldá-la de tal maneira que ela promova características-chaves de design que suportem os objetivos estratégicos associados à computação orientada a serviços. Os padrões de design fornecem soluções para problemas comuns encontrados ao aplicar os princípios de design e ao estabelecer um ambiente adequado para a execução lógica concebida de acordo com os princípios de orientação a serviços.

3. Padrões de Design

Em [Erl 2008], são apresentados diversos padrões, contudo serão descritos nesse artigo apenas os chamados padrões de serviços fundamentais, pois esses constituem um conjunto de padrões básicos de design que ajudam a estabelecer as características fundamentais do projeto de serviço através de uma sequência de aplicação sugerida. Coletivamente, esses padrões formam a aplicação mais básica de orientação a serviço.

3.1. Agnostic Capability

Funções de serviços derivadas de interesses específicos podem não ser úteis para múltiplos consumidores, assim reduzem a potencial reusabilidade do serviço.

Para resolver este problema, o *Agnostic Capability* recomenda particionar a lógica do serviço em um conjunto de capacidades bem definidas que abordam preocupações comuns não específicos de um problema.

Através da aplicação deste padrão, a lógica de serviço são agrupadas dentro de um contexto determinado serviço, disponibilizado como um conjunto de recursos bem definidos e complementares. Veja a figura 1.

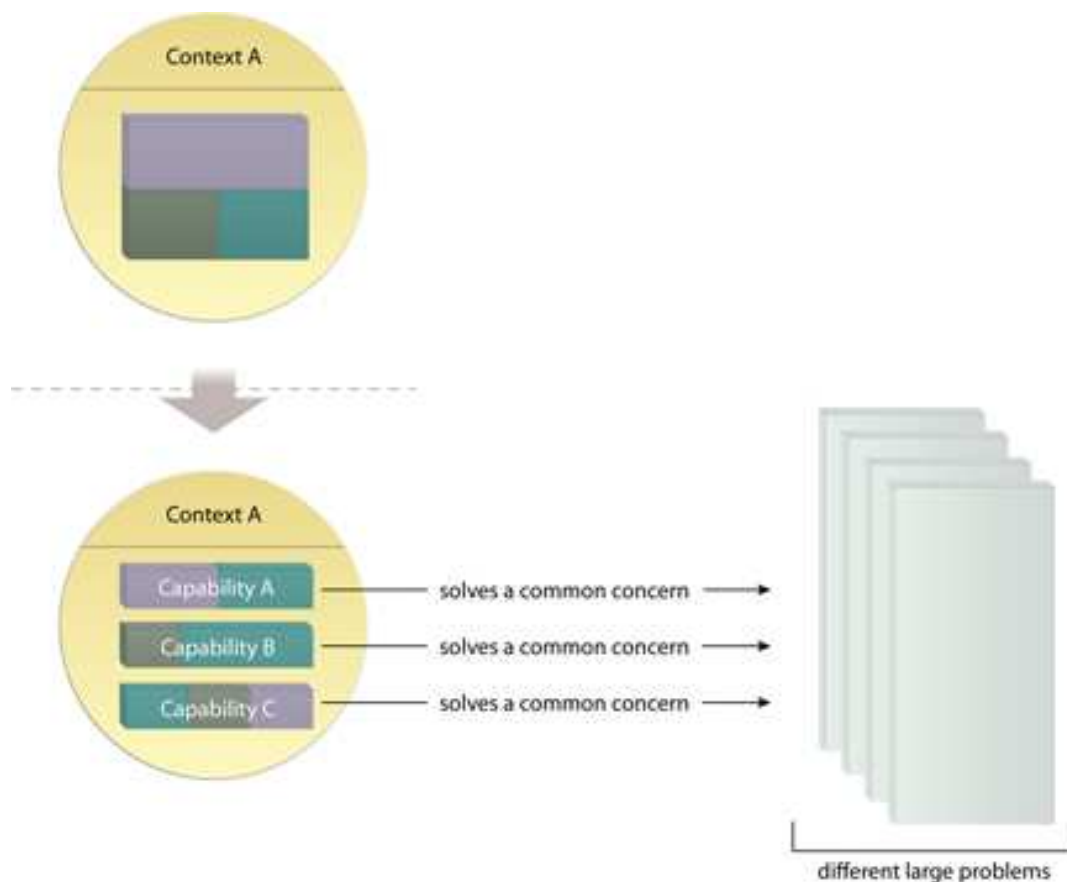


Figura 1. Agnostic Capability

3.2. Agnostic Context

Muita lógica agrupada com um único propósito resultam em serviços com pouco ou nenhum potencial de reutilização, acabando por gerar redundância dentro de uma empresa.

Para este tipo de problema, aqui é sugerido que a lógica que não é específica para uma finalidade seja isolada em serviços separados com contextos diferentes. Isto torna a solução reutilizável em um nível empresarial. Para entendermos melhor, vejamos a figura 2.

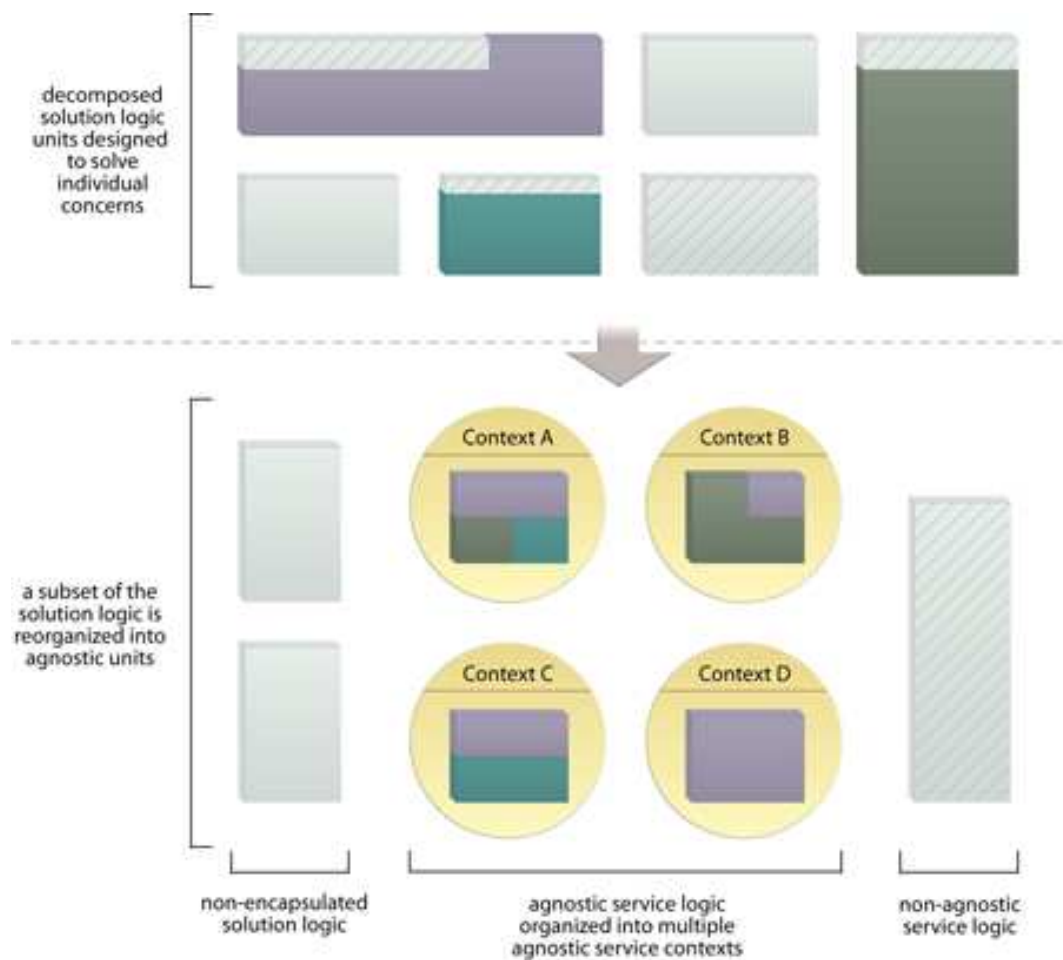


Figura 2. Agnostic Context

A aplicação desse padrão resulta em um subconjunto da solução lógica e em seguida são distribuídos em serviços com contextos específicos.

3.3. Functional Decomposition

Para resolver grandes e complexos problemas uma lógica também complexa deve ser criada, isto resulta em um serviço autônomo com restrições quanto sua reutilização. Para evitar este tipo de problema o padrão *Funcional Decomposition* sugere que quebrems este grande problema em um conjunto de problemas menores. Isto permite que as soluções também seja decompostas em um conjunto de soluções mais simples.

Assim como na computação distribuída que baseia-se em uma abordagem onde um grande problema é decomposto, este padrão resulta na decomposição do problema maior em problemas menores, como mostra figura abaixo.

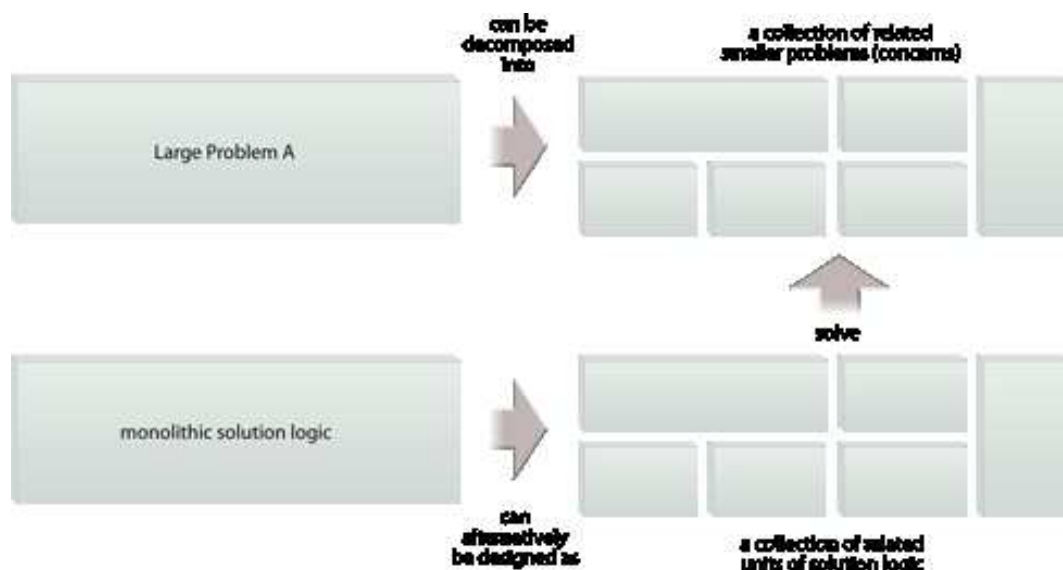


Figura 3. Functional Decomposition

3.4. Non-Agnostic Context

The subsection titles must be in boldface, 12pt, flush left.

4. Considerações Finais

Este artigo apresentou conceitos básicos e fundamentais para o universo de SOA e algumas soluções para problemas comuns encontrados na aplicação de princípios de design, vale ressaltar a importância que uma aplicação adequada com esses princípios e padrões de design tem para construção de sistemas que atendam as características relativas ao paradigma de orientação a serviço, os padrões aqui apresentados são uma pequena parte de um conjunto enorme de padrões que podem ser usados em SOA, quase em sua maioria, estes estão descritos de forma detalhada nos livros SOA Princípios de Design de Serviços e SOA Padrões de Design, ambos de Thomas Erl. Por fim é importante dizer que a visão de SOA traz um melhor entendimento de como pensar em software e como aproximar soluções tecnológicas de problemas reais.

Referências

- [Baccaro 2013] Baccaro, M. (2013). 8 princípios de design para soa. Acessada JAN. 2016.
- [Daigneau 2011] Daigneau, R. (2011). *Service Design Patterns: Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services*. Addison-Wesley.
- [Erl 2008] Erl, T. (2008). *SOA: Design Patterns*. Prentice hall.
- [Erl 2009] Erl, T. (2009). *SOA: Princípios de design de serviços*. Prentice hall Brasil.
- [Group] Group, G. Service-oriented architecture (soa). Acessada JAN. 2016.
- [Newcomer and Lomow 2005] Newcomer, E. and Lomow, G. (2005). *Understanding soa with web services*. Pearson Education.
- [OASIS 2006] OASIS (2006). *The oasis reference model for service oriented architecture 1.0*. Acessada JAN. 2016.

[Oliveira] Oliveira, E. M. D. Vantagens e desvantagens de soa. Acessada JAN. 2016.

[Patterns] Patterns, S. Soa design patterns and design principles. Acessada JAN. 2016.

[Siqueira] Siqueira, B. R. Introdução a arquitetura orientada a serviços (soa). Acessada JAN. 2016.

[Wikipédia] Wikipédia. Service-oriented architecture. Acessada JAN. 2016.