

Introdução à Arquitetura de SW

Curso de Engenharia de SW

Prof: Wilson Wistuba

Links e referências

- Bass, Len and Clements, Paul and Kazman, Rick. (2003, April). *Software Architecture in Practice, Second Edition*. Addison-Wesley Professional.
- Sommerville, Ian. (2004, May). *Software Engineering (7th Edition) (International Computer Science Series)*. Addison Wesley.
- Fowler, Martin - *Patterns of Enterprise Application Architecture* (2002).
- <http://cnx.org/content/m17524/latest/>
- <http://www.sei.cmu.edu/architecture/>
- <http://www.ibm.com/developerworks/rational/library/feb06/eeles/>
- <http://www.bredemeyer.com/>

Breve Histórico

- **1968 – Edsger Dijkstra** – Introduziu conceitos de estrutura de Software e camadas dessas estruturas para SO's, para ele não bastava programação correta.
- **1971-79 – David Parnas** – Criou o conceito de arquitetura de Software, separação da Interface da implementação de componentes . Aumento da extensibilidade para a detecção e tratamento de erros. Divisão de trabalho do Arquiteto e do Construtor.
- **1975-86 – Fred Brooks Jr.** – “Através da arquitetura do sistema quero demonstrar a especificação completa e detalhada da interface do usuário”. Separação de esforço de arquitetura, implementação e realização.
- **1992 – Dewayne E. Perry & Alexander L. Wolf** – Publicam os fundamentos da Arquitetura de Software – São “os pais” da arquitetura de software moderna.
 - a. Arquitetura como um framework para satisfazer os requisitos
 - b. Arquitetura como a base técnica para a concepção
 - c. Arquitetura como uma base eficaz para a reutilização
 - d. Arquitetura de base para a análise consistente

Breve Histórico

- **1994 – Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides** – Publicam o livro “Padrões de Projeto”. Embora fosse um livro voltado para designers e desenvolvedores aplicava claramente os conceitos de arquitetura para viabilizar o uso dos padrões nos projetos de SW.
- **1996 – David Garlan and Mary Shaw** – Ambos da Universidade de Carnegie Mellon publicaram o livro: “Software Architecture: Perspectives on an Emerging Discipline”.
 - Identificaram três níveis de *design* de software (*Arquitetura, Código e Executável*)
 - Introduziram quatro categorias de pesquisa e desenvolvimento de arquitetura
 - a. *Caracterização da arquitetura de software e sistemas - Architectural Description Languages*
 - b. *Catálogo dos princípios e padrões arquitetônicos*
 - c. *Construção de frameworks para domínios-específicos*
 - d. *Compreensão formal da arquitetura*
- Apresentaram uma série de estilos arquiteturais comuns (*Dataflow systems, Call-and-Return Systems, Independent Components, Virtual Machines, Data-Centred Systems*)

Breve Histórico

• **1998 – Bass, Clements and Kazman** – Escreveram o livro “Software Architecture in Practice”.

- Introduziram o "Architectural Business Cycle (ABC)"
- Explicaram o Software Architecture Analysis Method (SAAM)
- Descreveram algumas das Architectural Description Languages (ADLs)

• **1998 – 2000 – Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad e MichaelStal** – Escreveram os livros **PoSa** “Pattern-Oriented Software Architecture: A System Of Patterns”.

- Apresentaram os patterns
 - a. Layers
 - b. Pipes and Filters
 - c. Blackboard Pattern
 - d. Broker
 - e. Model View Controller
 - f. Presentation Abstraction Control
 - g. Micro Kernel
 - h. Reflection

Breve Histórico

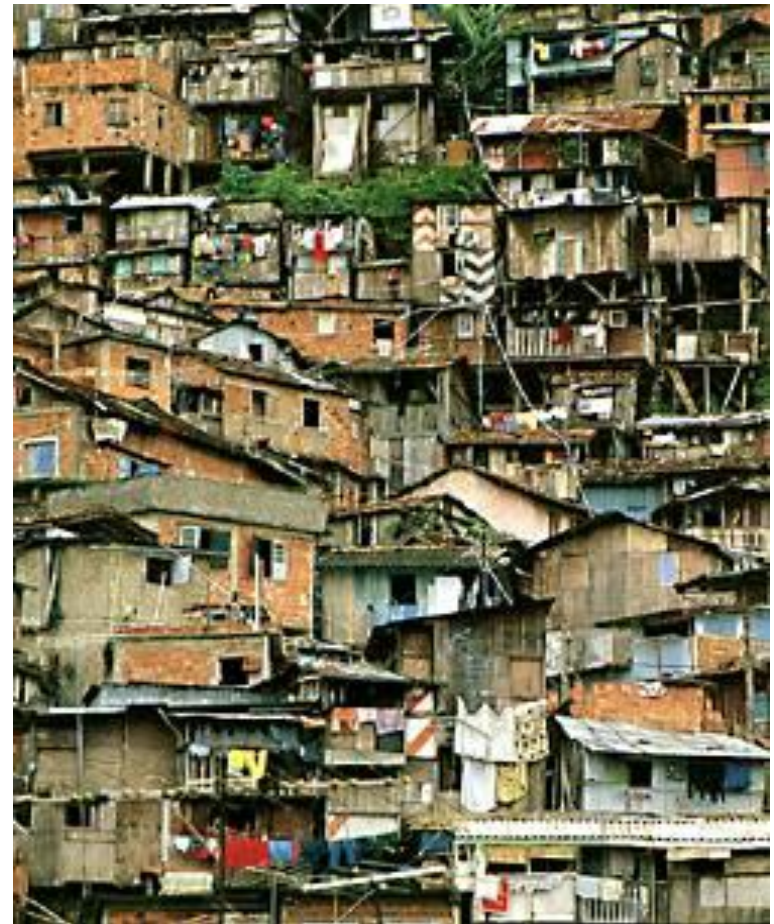
- **2000 – IEEE 1471 - ANSI/IEEE 1471-2000** – . *Recommended Practice for Architectural Description of Software-Intensive Systems*, Primeiro padrão do IEEE que trata da arquitetura de software de forma organizada.
- **2001 a 2007** – Há uma explosão de publicações sobre o temas podemos citar:
 - **Martin Fowler** – *Padrões de Arquitetura de Aplicações Corporativas*.
 - **Luke Hohmann** – *Beyond Software Architecture*.
 - **Rational** – *The Zachman Framework For Enterprise Architecture*
 - **Stephen T. Albin** – *The Art of Software Architecture: Design Methods and Techniques*.

O que é arquitetura de software?

Arquitetura: A arquitetura de software representa a estrutura do sistema, que consiste nos componentes de software, nas propriedades externamente visíveis desses componentes e nos relacionamentos entre eles.

Todo sistema já criado tem sua Arquitetura!

Ela existe independente do seu conhecimento!



O que é arquitetura de software?

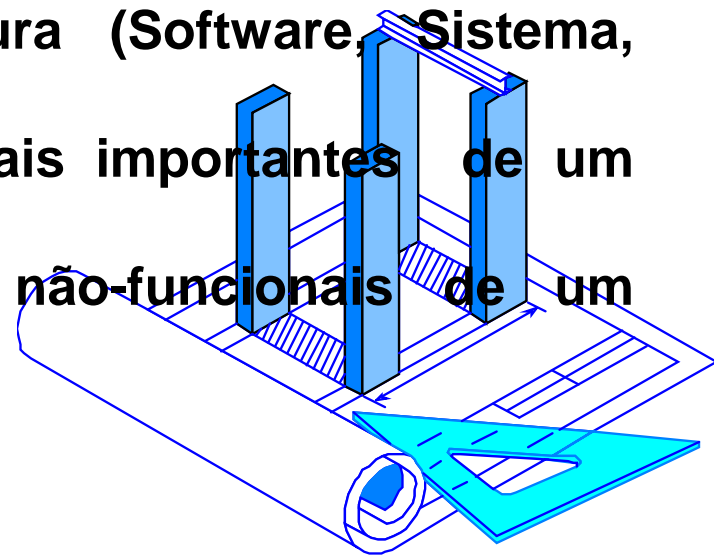
Arquitetura de TI (Tecnologia da Informação): É um conjunto de princípios, guias e regras que direcionam a organização a adquirir, construir, modificar e interagir com os recursos de TI da organização. Envolve:

- **A Composição / Decomposição do Sistema (Subsistemas / Módulos).**
- **A definição de componentes e a Interação entre os mesmos.**
- **A definição de camadas e a Interação entre as mesmas (Ordem / Estrutura).**
- **A organização das partes físicas do software a serem implementadas.**
- **A definição de restrições do sistema (naturais ou auto-impostas).**
- **A descrição geral do sistema.**
- **A Estrutura estática / dinâmica de um Sistema.**
- **O estilo que orienta o desenvolvimento e a evolução de um sistema.**
- **O apoio a funcionalidade do sistema.**

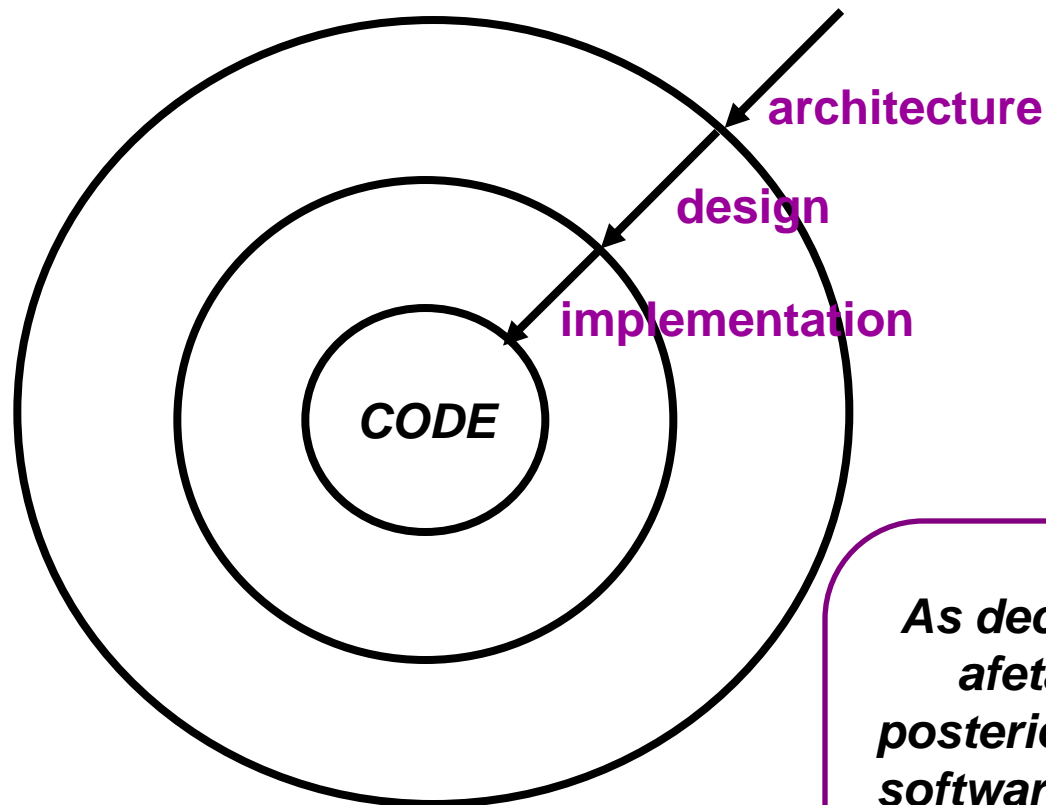


O que é arquitetura de software?

- Arquitetura pode ser vista como um processo
- Arquitetura pode ser vista como um artefato
- Arquitetura define os principais componentes de um sistema escondendo os detalhes de implementação e o que não pertence as iterações do software
- Arquitetura define os relacionamentos (estruturas) e interações entre os componentes de software
- Cada sistema tem uma arquitetura (*até mesmo um sistema composto de um componente apenas*)
- Arquitetura define a lógica por trás dos componentes e da estrutura.
- Existem vários tipos de arquitetura (Software, Sistema, Corporativa, Hardware, Rede etc)
- Arquitetura engloba as decisões mais importantes de um projeto de software!!
- Arquitetura resolve os requisitos não-funcionais de um sistema!!
- Arquitetura é estratégica!!



O que é arquitetura de software?



As decisões de arquitetura afetam todas as fases posteriores de um projeto de software. Desde o design até a codificação. Uma codificação pobre pode ser resultado de uma arquitetura mal definida!!!

Comparando Arquiteturas



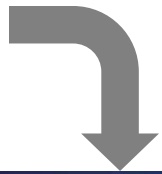
Para construção de uma “casa de cachorro” o processo é simplificado, as ferramentas são simples a modelagem não é tão elaborada. Um paralelo possível poderia ser um sistema de cadastro e consulta de clientes de um pequeno negócio como uma papelaria!

Comparando Arquiteturas



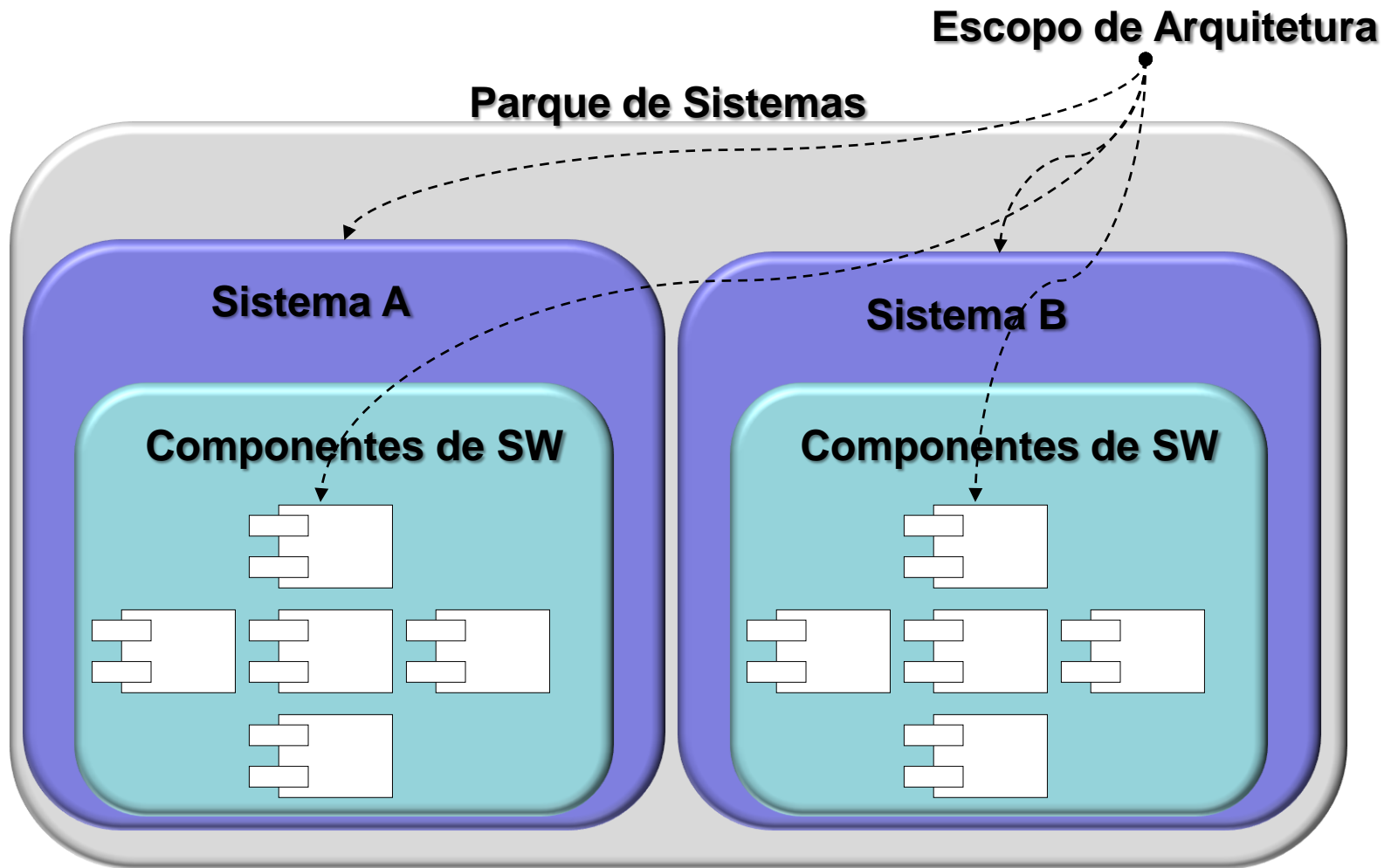
Para construção de uma “residência” as ferramentas devem ser adequadas e mais elaboradas. O processo deve ser bem estruturado e organizado a modelagem é mais complexa e elaborada. Um paralelo seria um sistema de e-commerce de um pequeno negócio ex: suplementos alimentares.

Comparando Arquiteturas



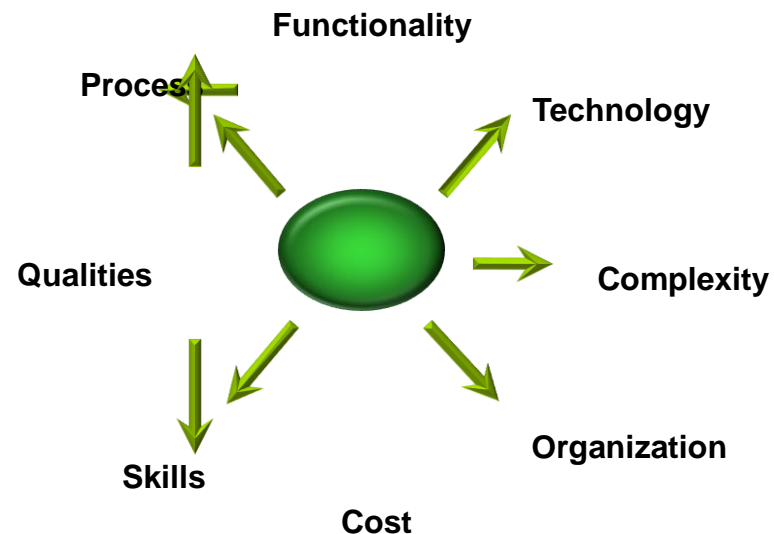
Para construção de um “edifício” são necessárias ferramentas avançadas e capazes de lidar com todos os detalhes do projeto. O processo é mais formal e organizado pois depende de muitas pessoas. A modelagem é fundamental para compartilhar os objetivos entre todos e “simular” a construção antes mesmo desta acontecer. Um paralelo possível seria um sistema de CRM de uma empresa de telecomunicações com milhões de clientes.

Escopo da arquitetura



Com o que se preocupar?

- **Decomposição do software**
- **Divisão de responsabilidades**
- **Integridade entre componentes do Software**
- **Abstração consciente (não inconsequente)**
- **Alinhamento com o negócio**
- **Antecipação dos problemas**
- **Organização e modelagem**
- **Iterações incrementais**
- **Padronização**
- **Simplicidade**
- **Clareza (sem ambiguidades)**



Arquitura vs Design

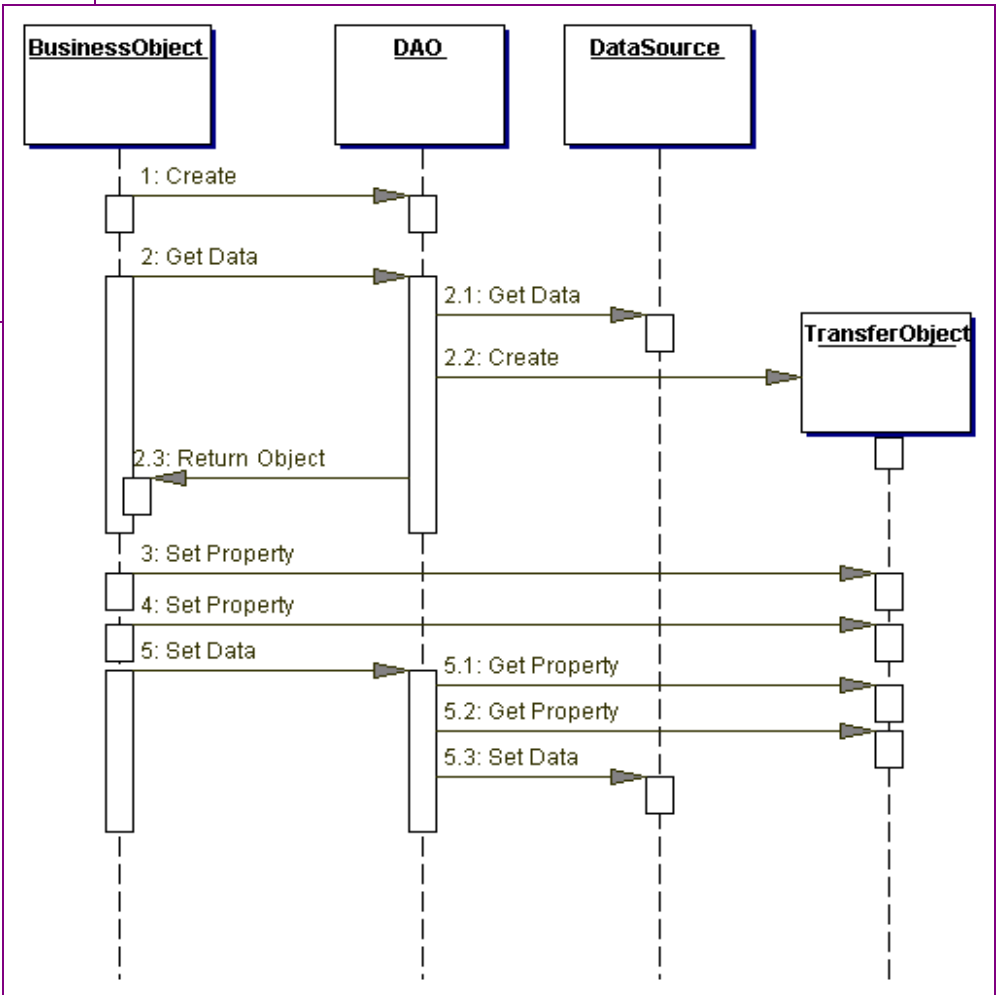
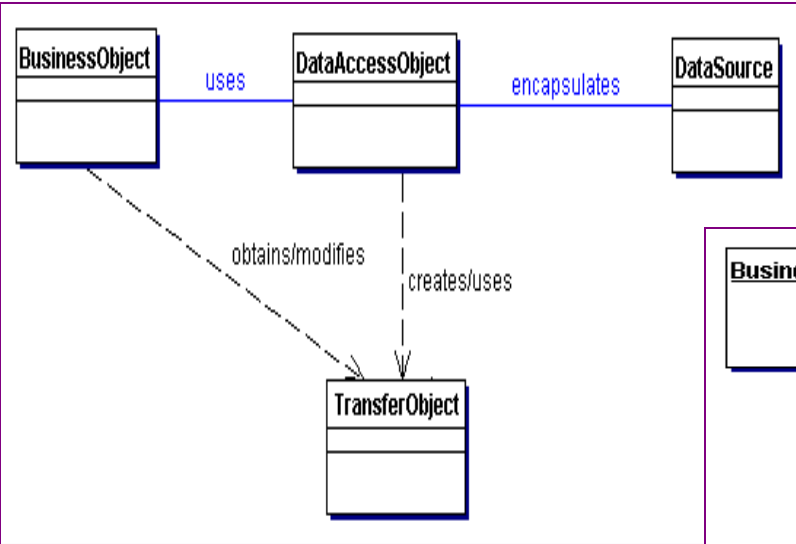
- **Arquitetura \neq Design:** Arquitetura e design são disciplinas distintas, a arquitetura utiliza elementos de design para poder representar as decisões mais importantes mas mantém-se em um grau de abstração muito maior. Assim esta não é um ASPECTO do Design orientado aos elementos que:
 - a. *São estruturalmente importantes. Ex: Representação de uma camada de dados*
 - b. *São diretamente ligados a requisitos como: Capacidade, Performance, Integridade. Exemplo uma fila JMS*

“Toda a arquitetura é design, mas nem todo design é arquitetura”

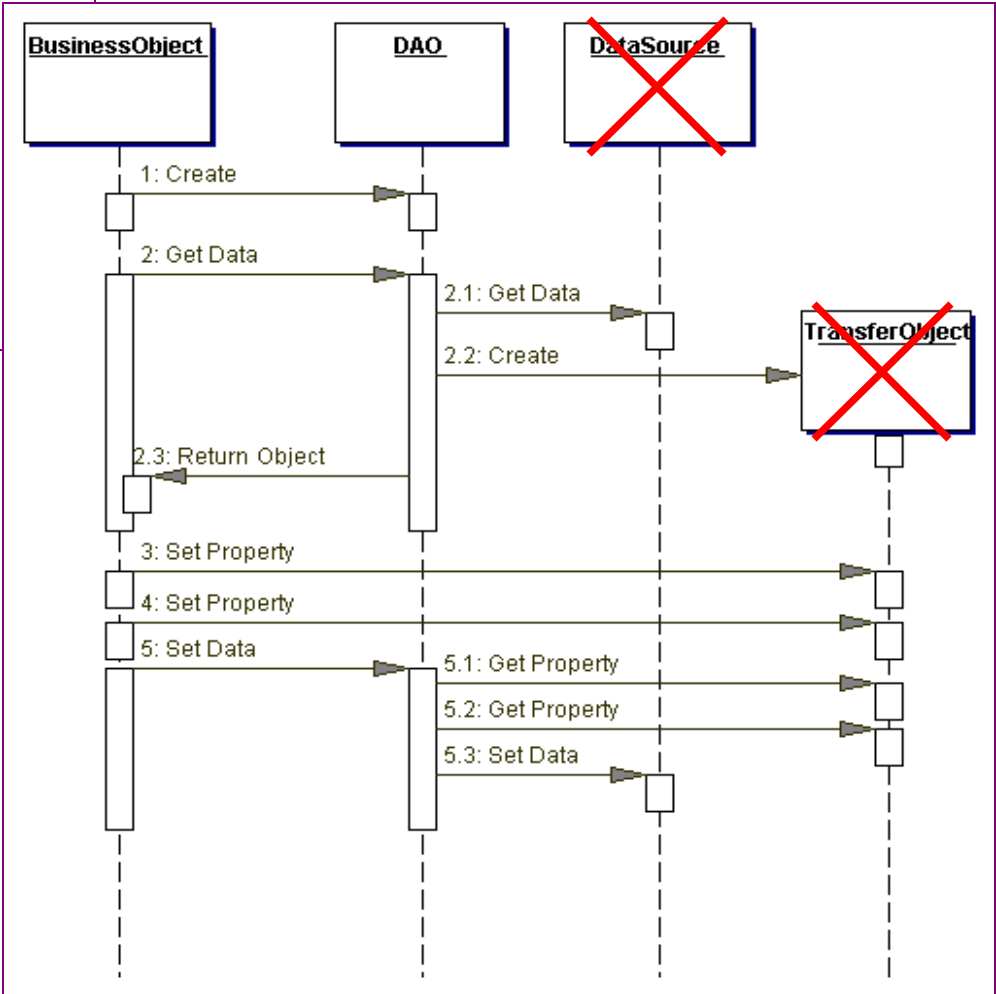
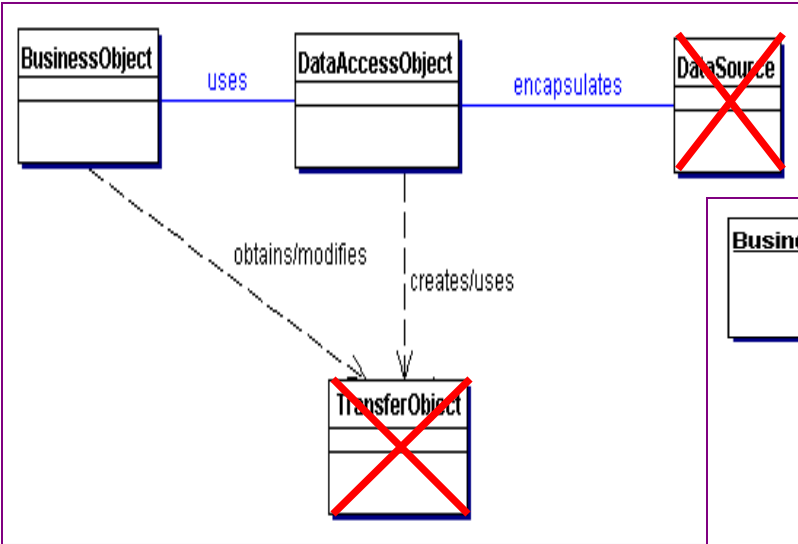
Grady Booch



Arquitetura vs Design – Foco no design



Arquitetura vs Design – Foco na arquitetura



Arquitetura vs Design - Discussão

É difícil “escolher” o que é arquitetonicamente significativo em um sistema, então a decisão deve ser de recursos tecnicamente experientes. Os arquitetos são os mais indicados para isso.!!!

O que é significativo difere em cada caso, dependendo do tipo de sistema e sobre a estratégia, dependendo do valor que este sistema irá entregar e como este irá apoiar o negócio da empresa em seus produtos e serviços. Como vamos diferenciar o que dita os problemas que vamos enfrentar? Onde vamos inovar? Onde temos de estar à frente da concorrência?

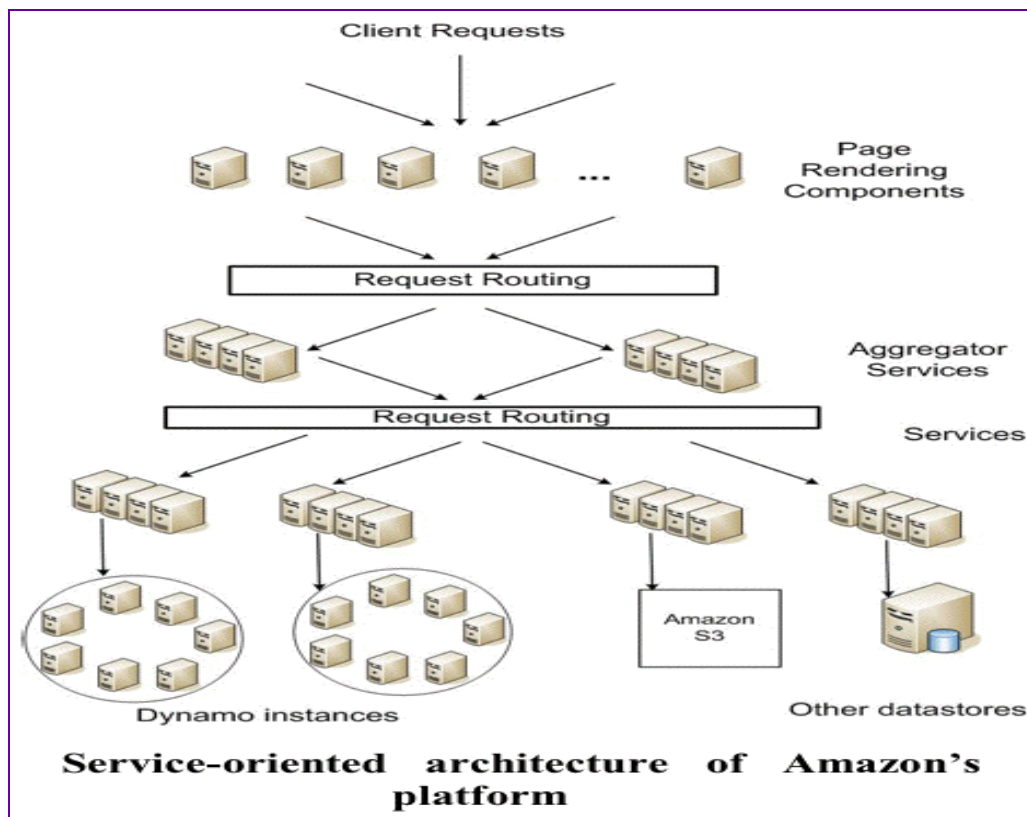
Essas são as perguntas que um arquiteto deve ter em mente ao desenhar a Arquitetura do Software (estratégia, complexidade, custo da mudança).

O design de um arquiteto deve endereçar o que é mais importante.
O design de um projetista deve viabilizar as definições do arquiteto de software.



Arquitetura ≠ Infra estrutura

- A infraestrutura é um aspecto importante da arquitetura mas não é o único
- A arquitetura define (também) o software que roda sobre a infra estrutura
- Arquitetura define a **INTEROPERABILIDADE** entre os componentes de software e a infra estrutura



Arquitetura é somente a estrutura

- A estrutura do software é apenas um dos ASPECTOS tratados pela arquitetura.
- Arquitetura capta também os aspectos dinâmicos do software.
- Arquitetura permeia o processo de negócio e o processo de desenvolvimento.
- Arquitetura requer analisar os software sob múltiplos pontos de vista



Quem é?

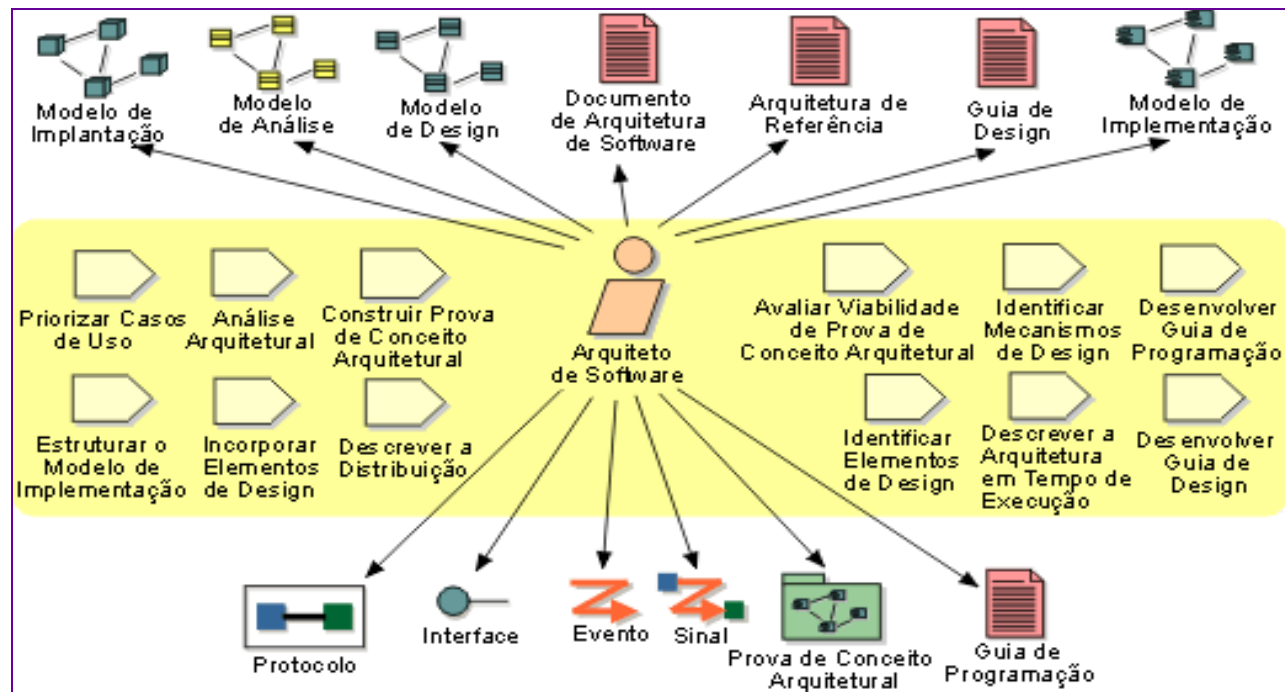
O Arquiteto do Sistema estabelece e refina a estrutura lógica e física do sistema e está preocupado com a otimização dessas estruturas em termos dos principais elementos do sistema e suas interfaces, e em fazer trocas em fatores competitivos e restrições (por exemplo, desempenho, custo, impacto ambiental) ao avaliar soluções potenciais que produzam o comportamento requerido. A visão do Arquiteto do Sistema se estende pelo sistema inteiro e todos os fatores, externos e internos, que podem afetar seu desenvolvimento. Dada a necessidade de manter essa visão ampla, o Arquiteto do Sistema raramente se envolve profundamente na engenharia detalhada de um sistema, preferindo deixar isso para outros profissionais nas diversas especialidades da engenharia.

** Fonte: RUP 7.0.1*

O papel do arquiteto de Software/Sistema?

O arquiteto de software é responsável por tomar as decisões mais críticas de um projeto. A competência técnica aqui é fundamental, o arquiteto precisa ser respeitado pelos desenvolvedores para executar seu trabalho. Portanto a proximidade com sua especialidade técnica é essencial. Ele ainda é um líder um coordenador das atividades técnicas

* Fonte: RUP 7.0.1



Quem são os arquitetos de Software/Sistema nas empresas?

- *Os arquitetos são analistas de sistemas, desenvolvedores, projetistas com experiência e passagem por diversos projetos.*
- *Tem em comum capacidade técnica, a visão abrangente e a perspicácia na tomada de decisões*
- *Nem sempre os profissionais são somente arquitetos, é comum que em empresas menores o arquiteto seja um “papel” desempenhado pelo time de desenvolvedores.*
- *Nas empresas maiores existem times de arquitetura, o profissional executa o trabalho de arquitetura em tempo integral*
- *O cargo de arquiteto de sistemas é considerado o topo da carreira técnica, os salários são maiores que os demais cargos técnicos*

Algumas empresas que empregam arquitetos no Brasil

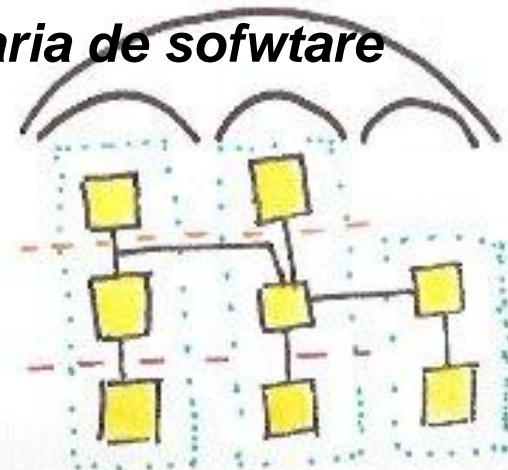


Skills de um arquiteto de Software/Sistemas

- **Experiência**
- **Liderança**
- **Comunicação**
- **Orientação por metas**
- **Proatividade**
- **Negociação**
- **Facilidade de “navegação” entre as camadas de software**
- **Capacidade de abstração**
- **Aceitar as ambiguidades de um software – engenheiros de software são treinados para eliminá-las.**
- **Gerenciar requisitos contraditórios ou o “tradeoff” entre eles**
- **Forte visão e convicção de suas decisões**
- **Entender o trabalho técnico**
- **Profundo conhecedor do processo de engenharia de software**
- **Precisa conhecer o domínio**

"O arquiteto ideal deve ser uma pessoa erudita, um matemático, familiarizado com estudos históricos, um estudioso aplicado de filosofia, conhecedor de música, que não desconheça medicina, detentor de saber jurídico e familiarizado com astronomia e cálculos astronômicos."

Vitruvius, há aproximadamente 25 anos a.C.



Boas práticas do Arquiteto de Software/Sistemas?

- *Realizar a decomposição em camadas*
- *Conquistar a confiança e o respeito do time de desenvolvedores*
- *Alinhamento constante com os stakeholders do projeto*
- *Praticar e aprimorar sempre a comunicação*
- *Testar a arquitetura proposta, isto significa que o arquiteto sempre está próximo do desenvolvimento*
- *Guiar-se por uma arquitetura de referência*
- *Usar alguma forma de especificação da arquitetura (uso de ADL's)*
- *Negociar sempre entre as capacidades do software (ex: manutenibilidade vs complexidade)*
- *Entender qual estilo arquitetural deve ser aplicado para cada caso*
- *Manter-se atualizado tecnicamente conhecer as novas tendências tecnológicas, manter-se aberto a novas idéias.*
- *Sempre traçar vantagens e desvantagens das decisões de arquitetura*
- *Embora arquitetura não trate dos detalhes, os problemas muitas vezes são encontrados nos níveis mais baixos. (Arquitetura minimalista)*

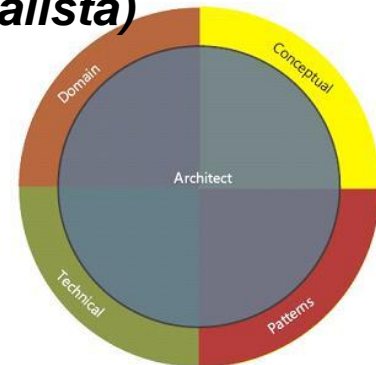
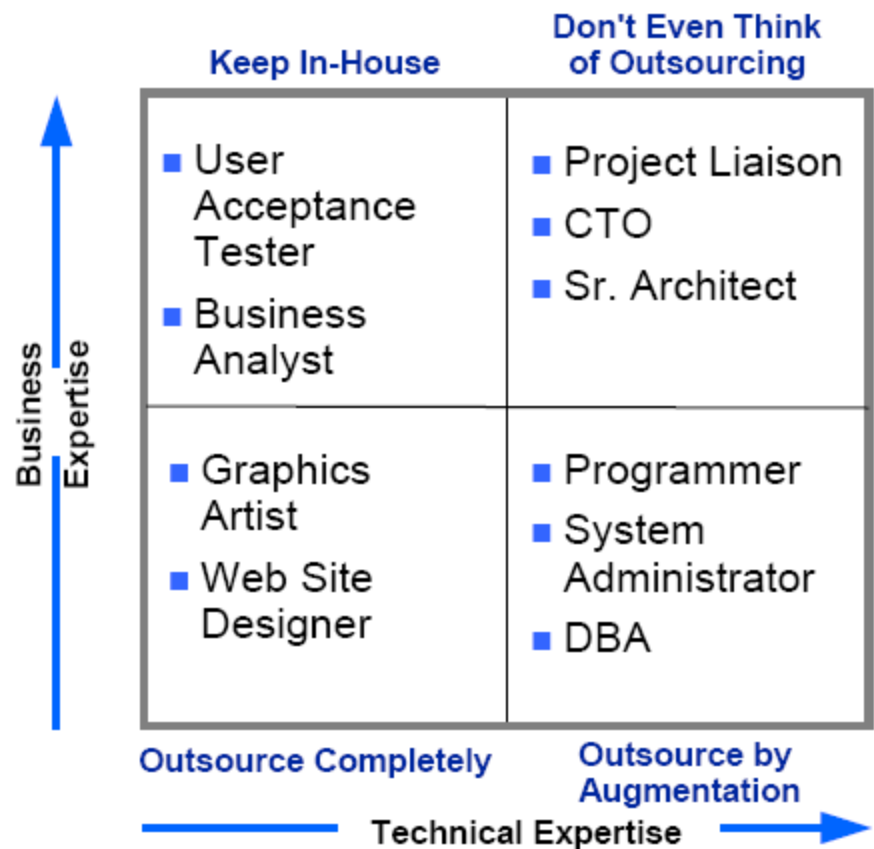


Figure 1- Áreas-chaves de conhecimento para um Arquiteto de Software

Times de arquitetura

- Nas empresas e nos projetos os times de arquitetura são pequenos (*discussão*)
- Os times de arquitetura buscam o equilíbrio entre “issues” técnicas e requisitos de negócio. *Exemplo: A falta de recursos pode obrigar um sistema a rodar sobre única TIER*
- Os times de arquitetura possuem um arquiteto chefe.
- Os times de arquitetura se preocupam com a governança seja ela corporativa ou do projeto.
- O time de arquitetura “mede” seu trabalho de tempos em tempos
- Os times de arquitetura tem uma estrutura flexível, os membros circulam e mudam durante as fases do projeto
- Mantém a integridade de um sistema
- Avaliam os riscos técnicos
- Participam no planejamento do projeto
- Propõem a ordem em que as iterações do software podem ser realizadas
- Estudam as mudanças futuras no produto de software
- Trabalham para mitigar os riscos técnicos e as técnicas utilizadas
- O Time de arquitetura entrega artefatos da arquitetura

Times de arquitetura



Gartner

• Segundo estudos do Gartner em 2007 os arquitetos não devem ser terceirizados!!!

* Esse material foi apresentado em abril de 2007 no WTC

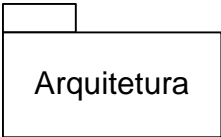
Business Cycle

Architecture Forces

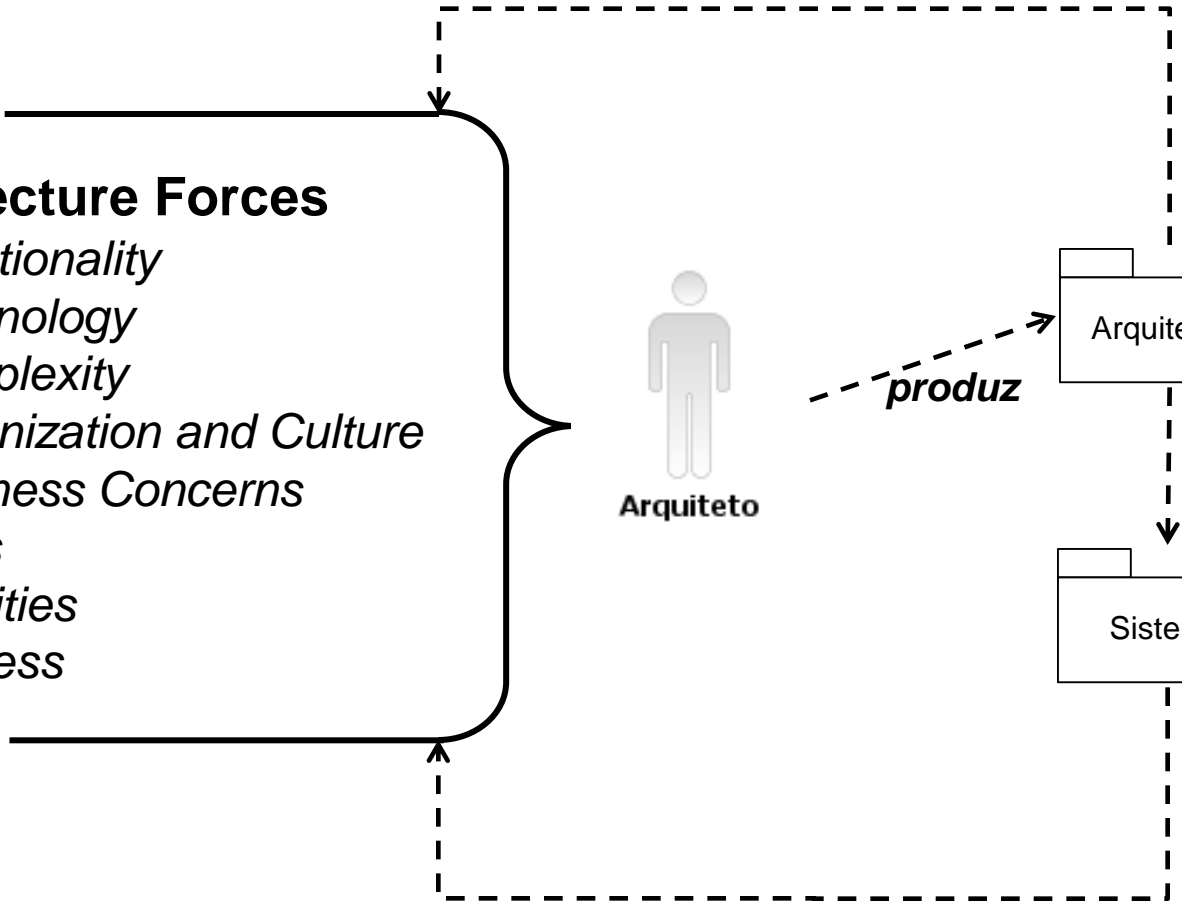
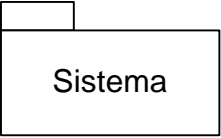
- Functionality
- Technology
- Complexity
- Organization and Culture
- Business Concerns
- Skills
- Qualities
- Process



produz



direciona

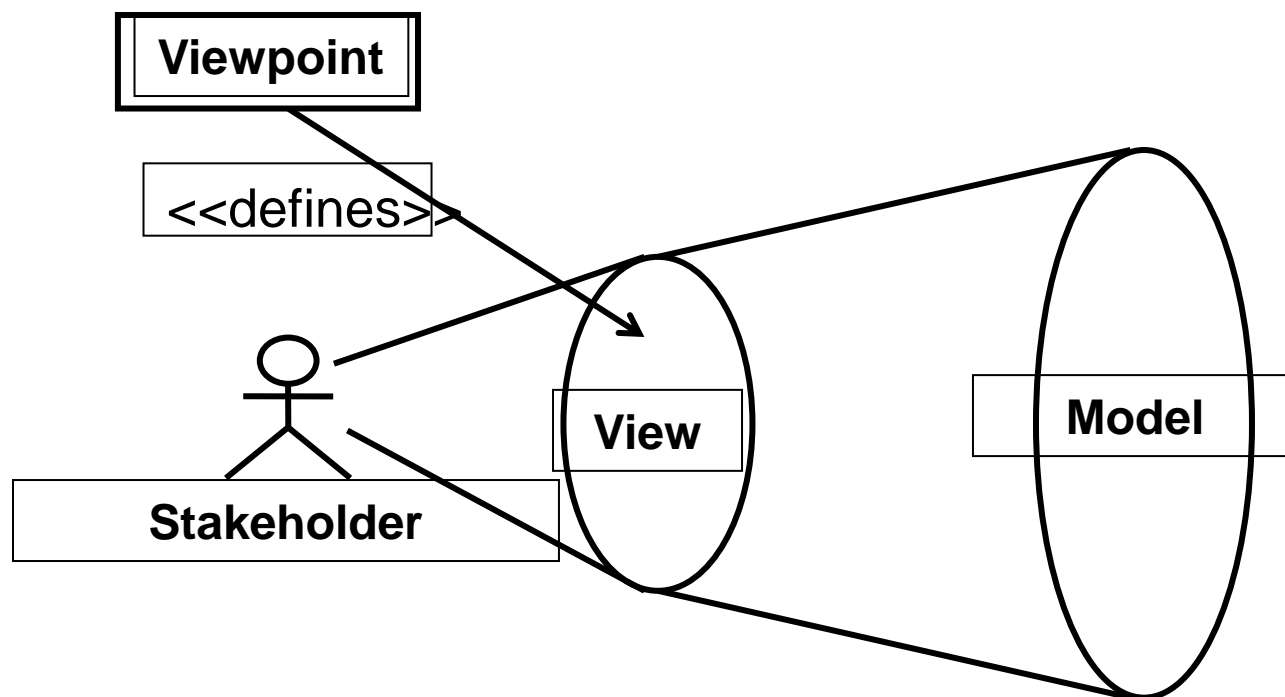


Descrição Arquitetural

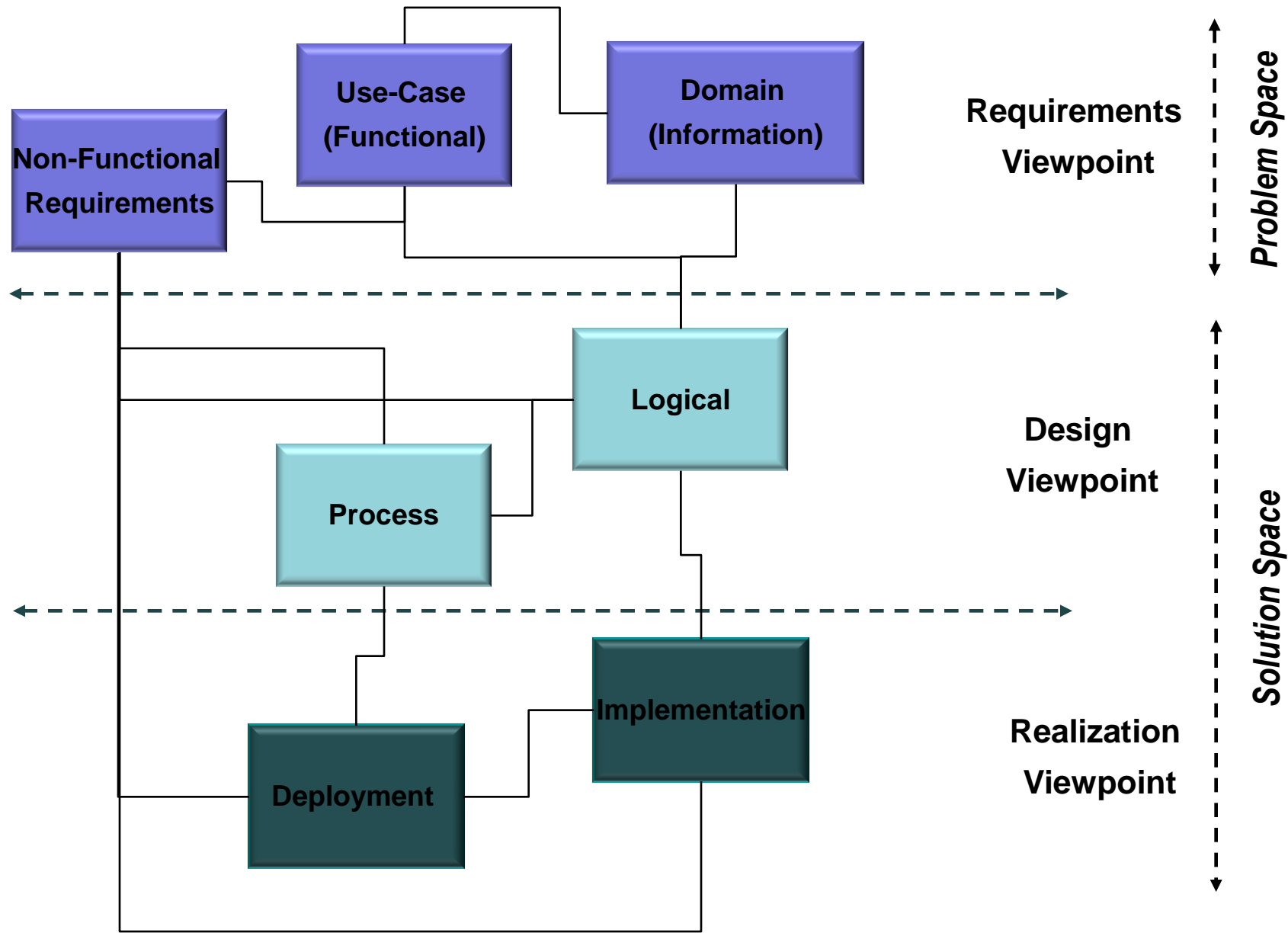
- A descrição arquitetural de um software ou sistema envolve capturar os aspectos arquiteturais, requisitos não funcionais e alguns funcionais. Reunir de forma lógica os aspectos do sistema. Envolve o uso de estilos arquiteturais e visualizações típicas (Ex: 4+1 do RUP) - Visualização de casos de Uso, Visão lógica, Visualização de Implementação, Visão de Processos, Visão de implementação.
- Em uma descrição de arquitetura um lugar comum é referenciar-se a uma “arquitetura de referência”, padrões de arquitetura e boas práticas de desenvolvimento.
- Em um sistema a descrição arquitetural se preocupa com o software e na integração dos componentes entre TIERS e outros ativos da solução
- Um exemplo de documento que reúne a descrição de arquitetura. DAS (Documento de Arquitetura de Software)

Descrição Arquitetural

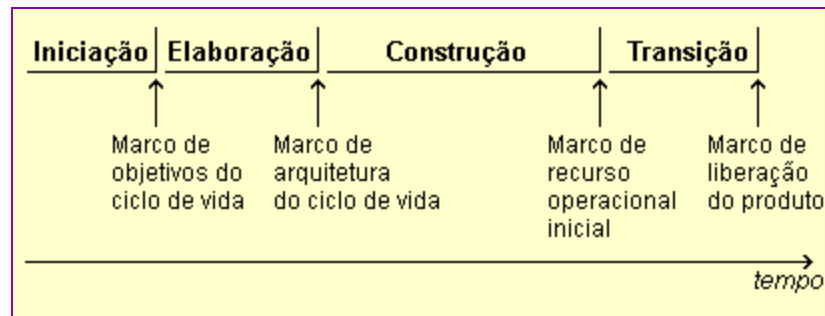
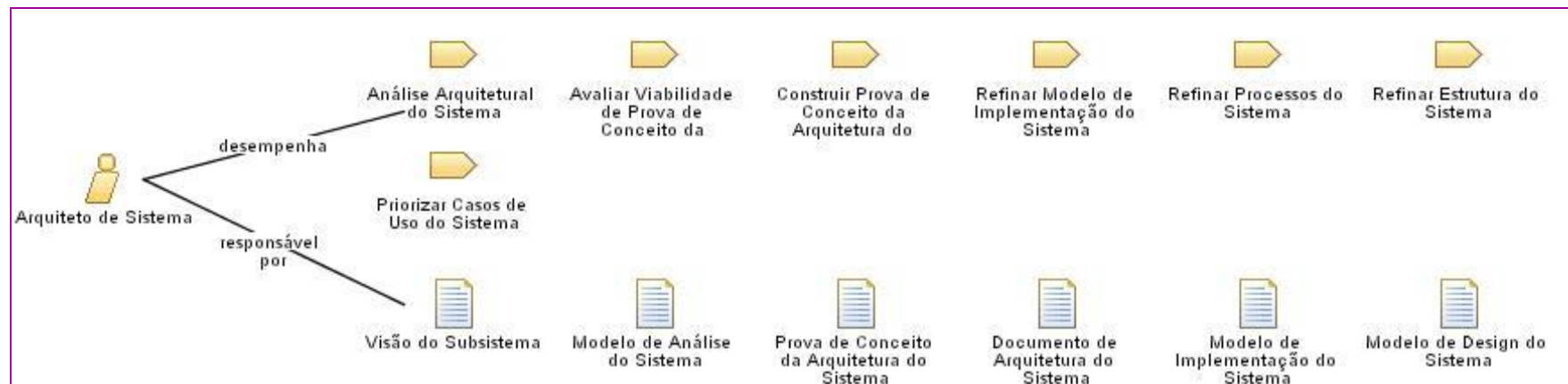
- **Modelo:**
Descrição completa, estruturada e organizada de um sistema a partir de uma perspectiva. Todo modelo assume um certo grau de abstração
- **Visão:**
Projeção do modelo que vê o sistema de uma determinada perspectiva geralmente exaltando uma vantagem e omitindo outros detalhes relevantes
- **Ponto de vista:**
Definição ou descrição da visão, conteúdo significando uma representação em um determinada notação e usando técnicas de modelagem.



Visões e Pontos de Vista de Arquitetura



Processo de Arquitetura (RUP)



ADL's – Exemplo: ACME

```
System simple_cs = {  
  Component client = {Port send-request}  
  Component server = {Port receive-request}  
  Connector rpc = {Roles {caller, callee}}  
  Attachments : {client.send-request to rpc.caller;  
                  server.receive-request to rpc.callee}  
}
```

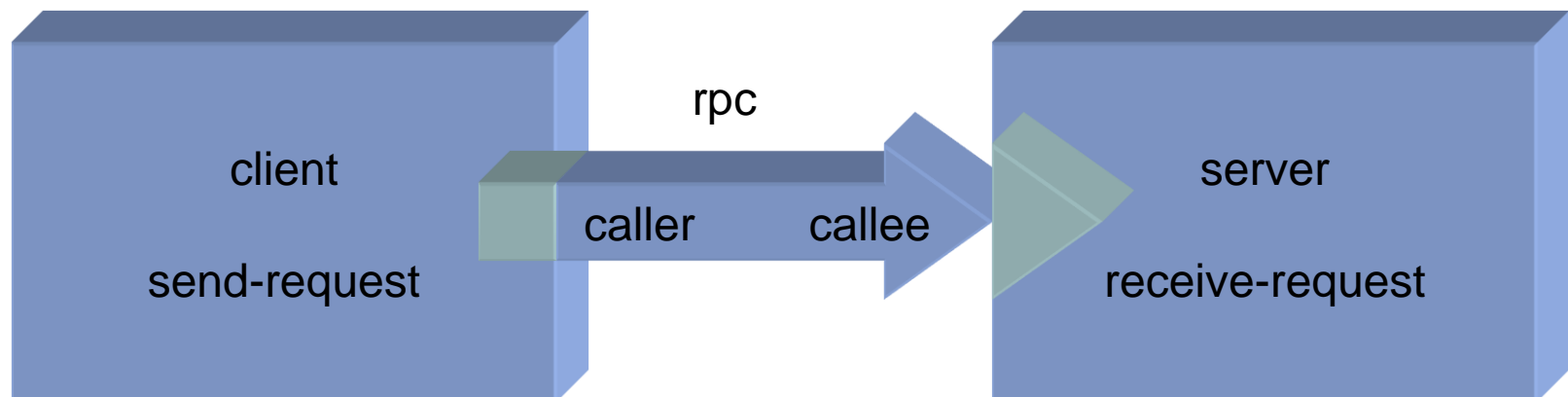
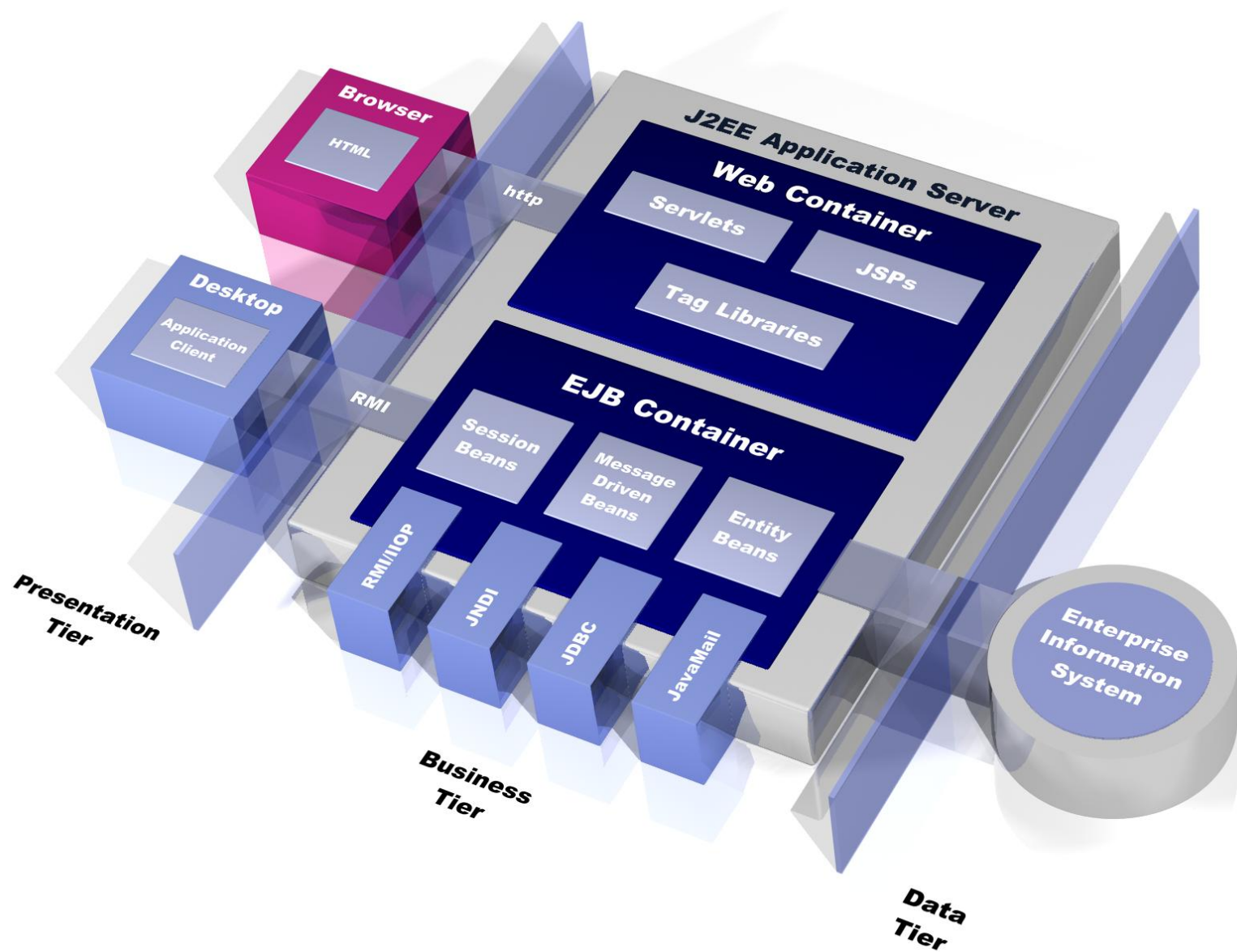
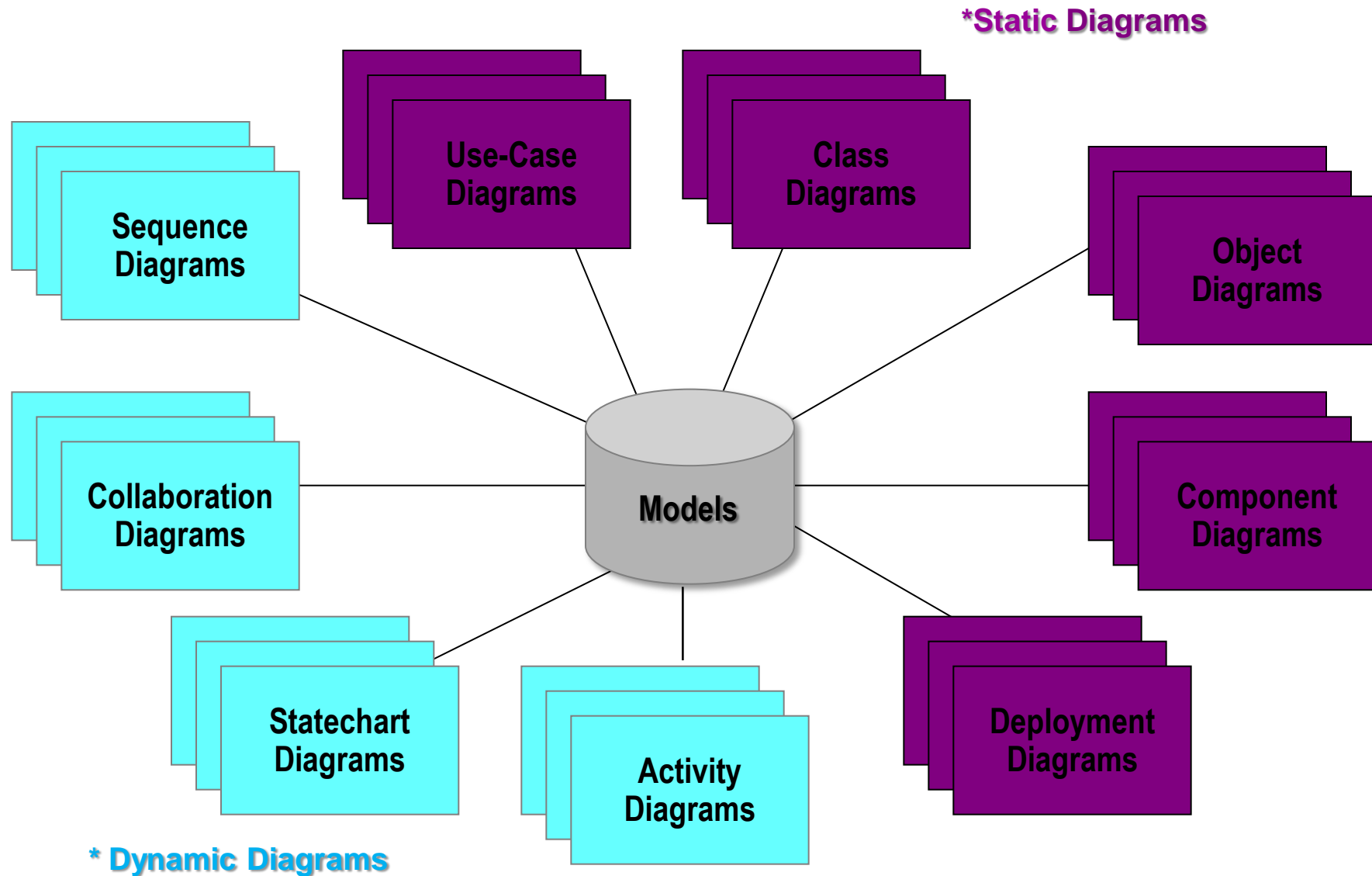


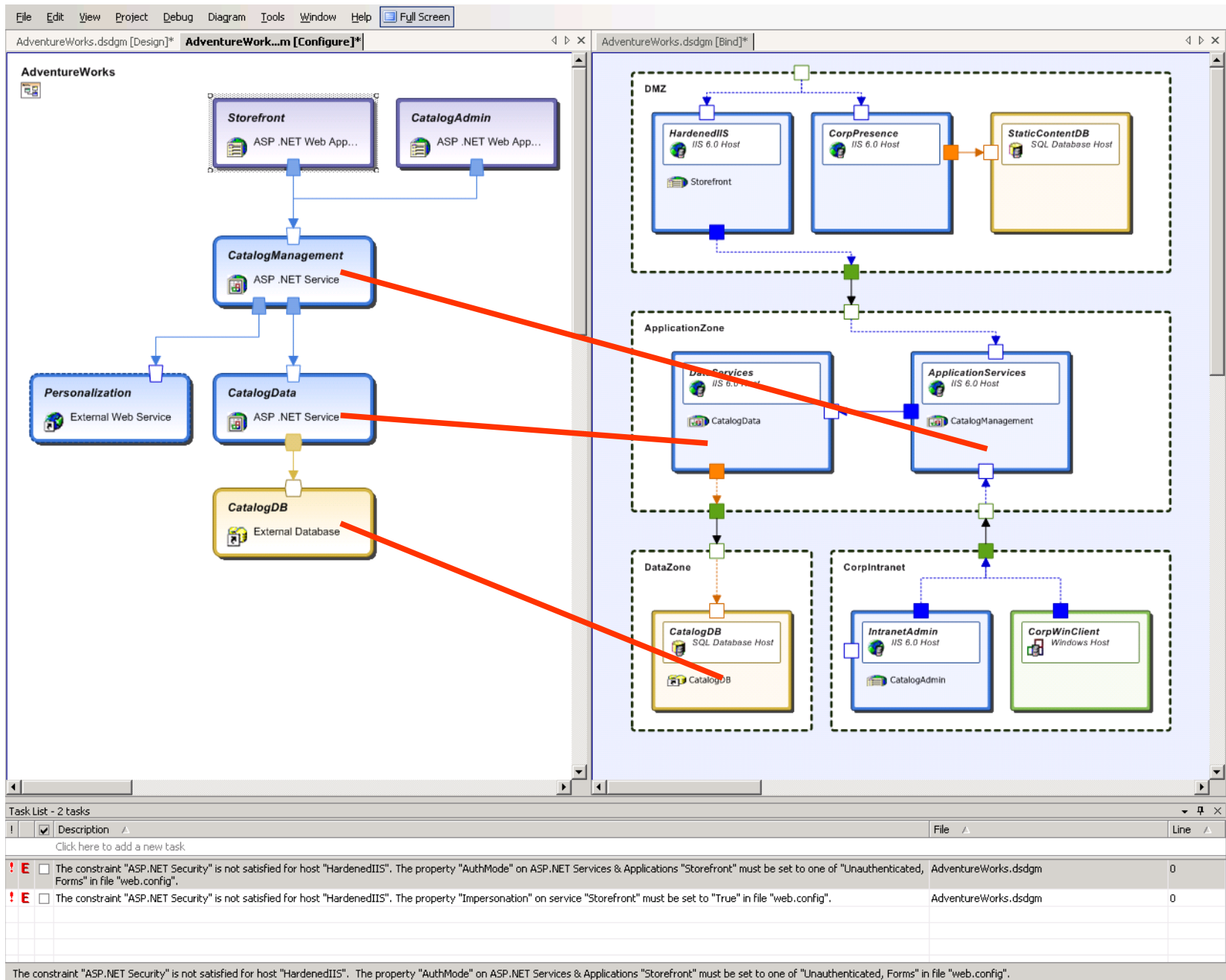
Diagrama de blocos (sem padrão)



UML 2.0



ADL's – DSL (Domain Specific Language)



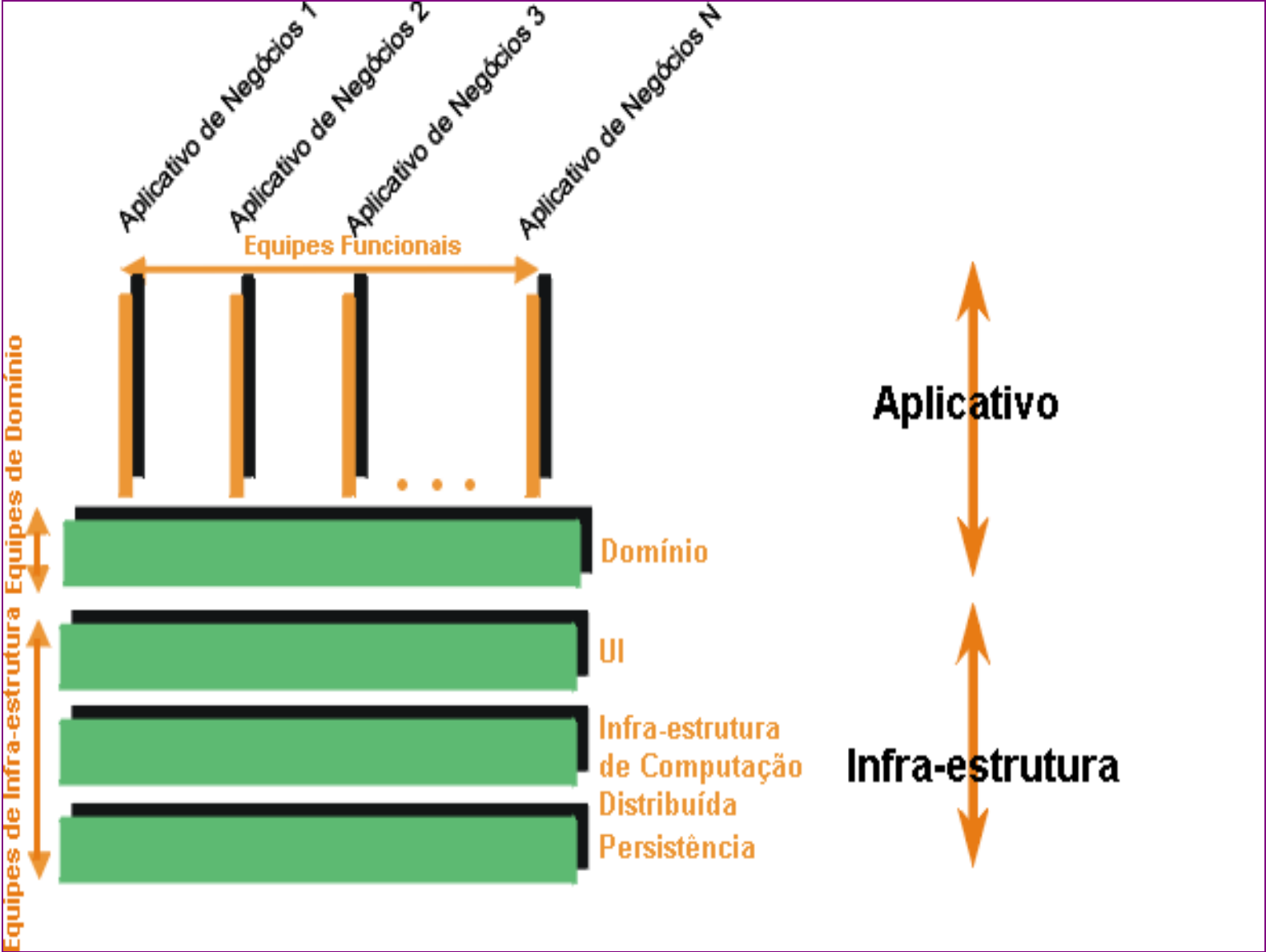
Benefícios da Arquitetura de Software

- **Arquitetura de software proporciona:**
 - Integridade sistêmica entre diversos componentes
 - Absorver novos requisitos
 - Mudança de requisitos durante o projeto
 - Cumprimentos de requisitos críticos
 - Evoluções no sistema de forma gradual e menos impactante
 - Maior garantia de “ROI” ou “Payback”
 - O valor da arquitetura pode ser observado durante a manutenção do software e o processo de desenvolvimento
 - Garante que o sistema irá responder sob condições adversas.
 - Separação de responsabilidades:
 - *Um web designer desenvolve a camada web*
 - *Um AD desenvolve a camada de dados*
 - *Um programador java desenvolve os EJB's*

Benefícios da arquitetura

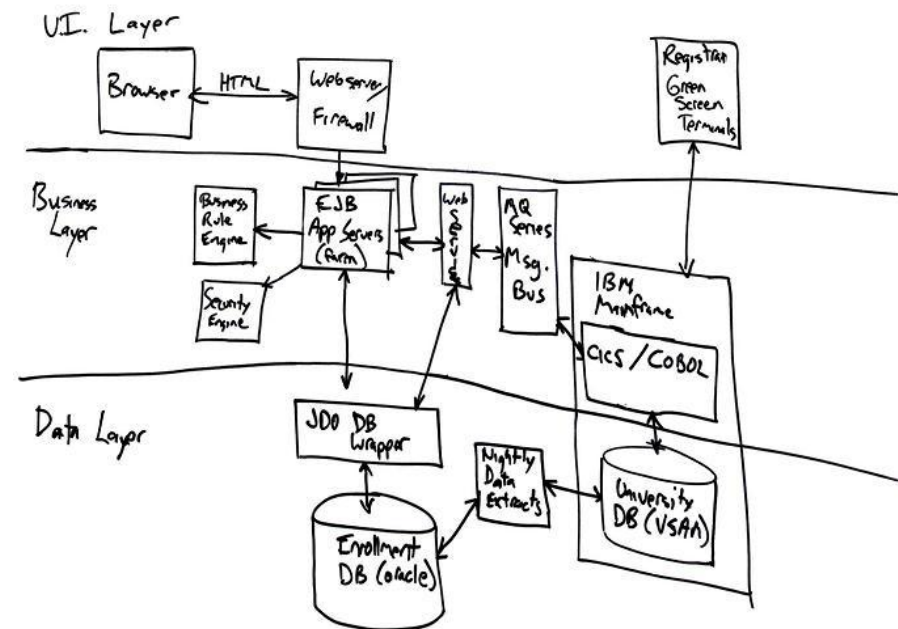
- **Components de sistemas baseados em uma arquitetura de software suportam melhor:**
 - **Diagnósticos**
 - **Rastreabilidade**
 - **Detecção de erros**
- **Arquitetura habilita diferentes graus de re-uso**
 - **Componentes**
 - **Camadas**
 - **Produtos**
 - **Frameworks**
 - **Configurações**
- **Arquitetura define os limites de um sistema**

Exemplo

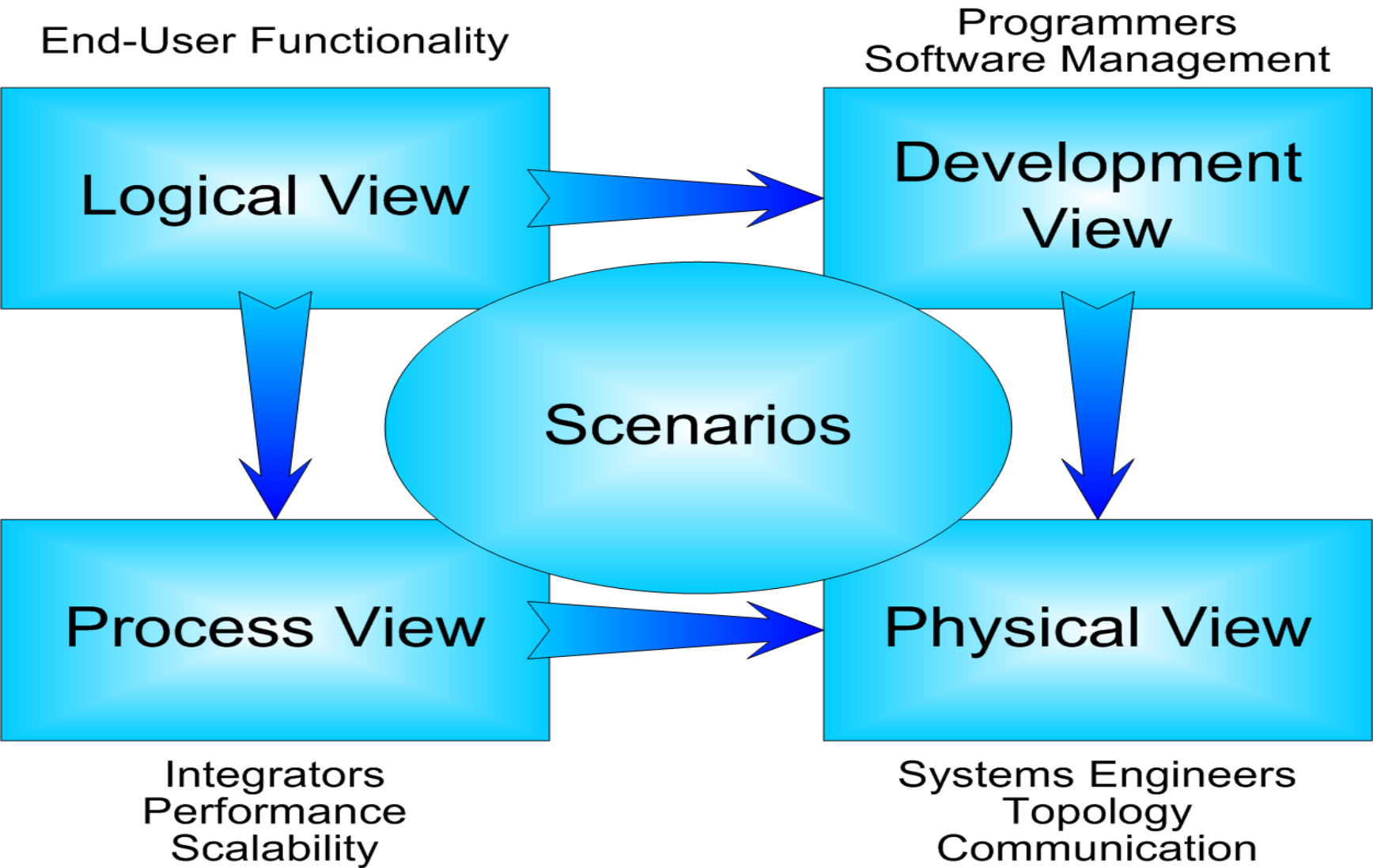


Frameworks de Arquitetura







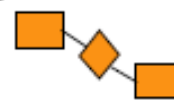
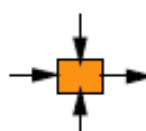



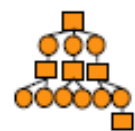
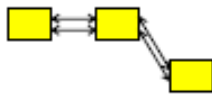
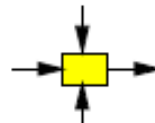
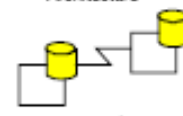


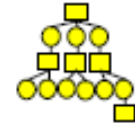
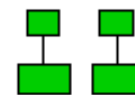
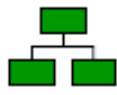
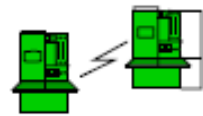

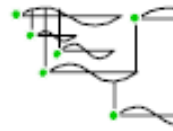
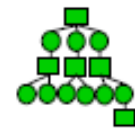






- **Definição:** Em geral são metodologias e normas organizadas quase sempre com foco na Arquitetura Corporativa. A arquitetura de sistemas em empresas pode ser complexa, de maneira que esses frameworks ajudam a organizar, classificar e resolver os problemas envolvendo cenários diversos como: Governo, Financeiras, Telcos e outras indústrias.
- **Alguns dos frameworks mais citados:**
 - *RUP (4+1)*
 - *Zachman Framework*
 - *TOGAF*
 - *DoDAF*



Ex: Frameworks de Arquitetura – RUP

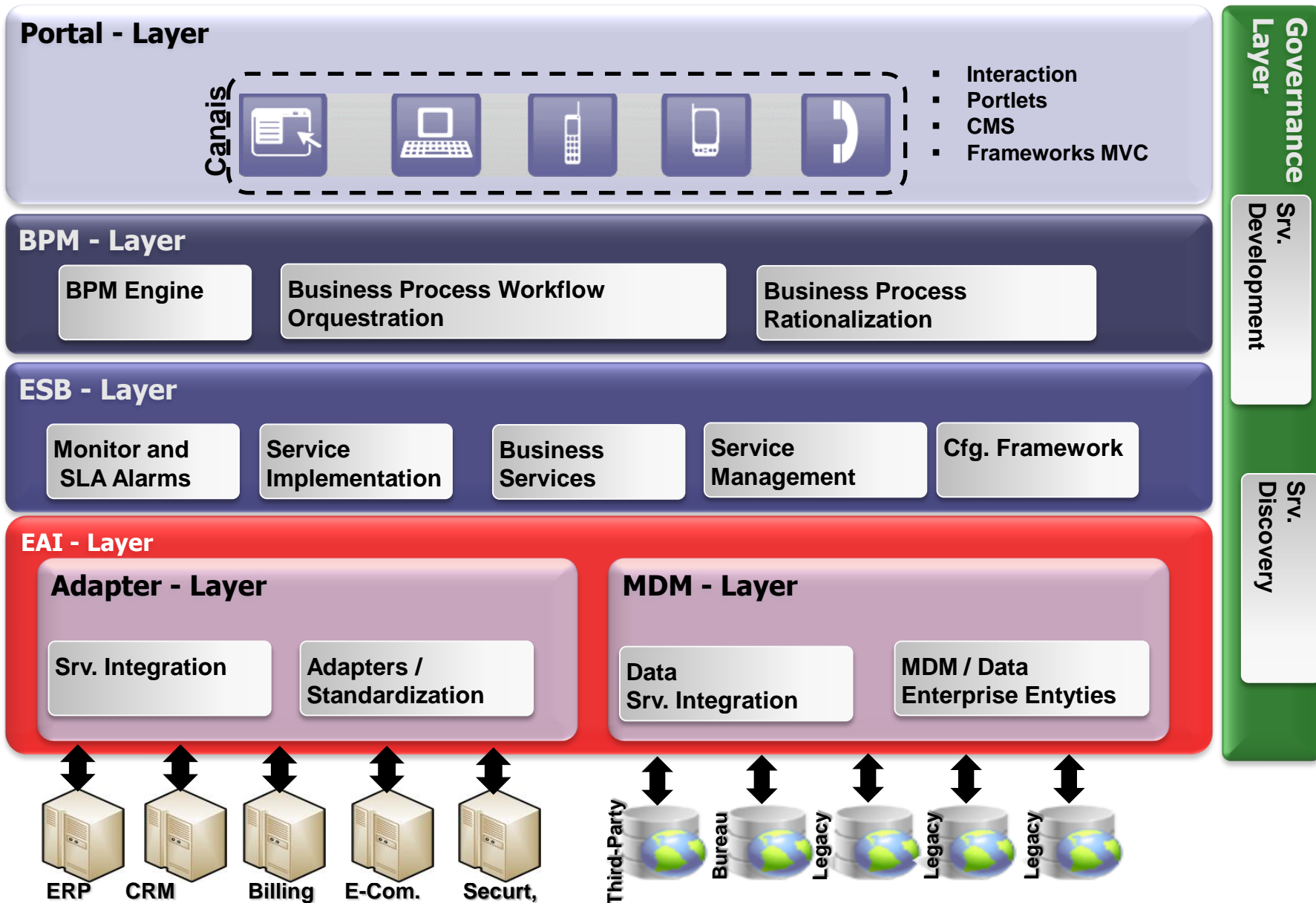


Ex: Frameworks de Arquitetura - Zachman

	DATA <i>What</i>	FUNCTION <i>How</i>	NETWORK <i>Where</i>	PEOPLE <i>Who</i>	TIME <i>When</i>	MOTIVATION <i>Why</i>
SCOPE (CONTEXTUAL)	List of Things Important to the Business 	List of Processes the Business Performs 	List of Locations in which the Business Operates 	List of Organizations Important to the Business 	List of Events Significant to the Business 	List of Business Goals/Strat 
Planner	ENTITY = Class of Business Thing	Function = Class of Business Process	Node = Major Business Location	People = Major Organizations	Time = Major Business Event	Ends/Mean=Major Bus. Goal/Critical Success Factor
ENTERPRISE MODEL (CONCEPTUAL)	e.g. Semantic Model 	e.g. Business Process Model 	e.g. Business Logistics System 	e.g. Work Flow Model 	e.g. Master Schedule 	e.g. Business Plan 
Owner	Ent = Business Entity Rein = Business Relationship	Proc. = Business Process I/O = Business Resources	Node = Business Location Link = Business Linkage	People = Organization Unit Work = Work Product	Time = Business Event Cycle = Business Cycle	End = Business Objective Means = Business Strategy
SYSTEM MODEL (LOGICAL)	e.g. Logical Data Model 	e.g. Application Architecture 	e.g. Distributed System Architecture 	e.g. Human Interface Architecture 	e.g. Processing Structure 	e.g. Business Rule Model 
Designer	Ent = Data Entity Rein = Data Relationship	Proc. = Application Function I/O = User Views	Node = I/S Function (Processor, Storage, etc.) Link = Line Characteristics	People = Role Work = Deliverable	Time = System Event Cycle = Processing Cycle	End = Structural Assertion Means = Action Assertion
TECHNOLOGY MODEL (PHYSICAL)	e.g. Physical Data Model 	e.g. System Design 	e.g. Technology Architecture 	e.g. Presentation Architecture 	e.g. Control Structure 	e.g. Rule Design 
Builder	Ent = Segment/Table/etc. Rein = Pointer/Key/etc.	Proc. = Computer Function I/O = Data Elements/Sets	Node = Hardware/System Software Link = Line Specifications	People = User Work = Screen Format	Time = Execute Cycle Cycle = Component Cycle	End = Condition Means = Action
DETAILED REPRESENTATIONS (OUT-OF-CONTEXT)	e.g. Data Definition 	e.g. Program 	e.g. Network Architecture 	e.g. Security Architecture 	e.g. Timing Definition 	e.g. Rule Specification 
Sub-Contractor	Ent = Field Rein = Address	Proc. = Language Stmt I/O = Control Block	Node = Addresses Link = Protocols	People = Identity Work = Job	Time = Interrupt Cycle Cycle = Machine Cycle	End = Sub-condition Means = Step
FUNCTIONING ENTERPRISE	e.g. DATA	e.g. FUNCTION	e.g. NETWORK	e.g. ORGANIZATION	e.g. SCHEDULE	e.g. STRATEGY

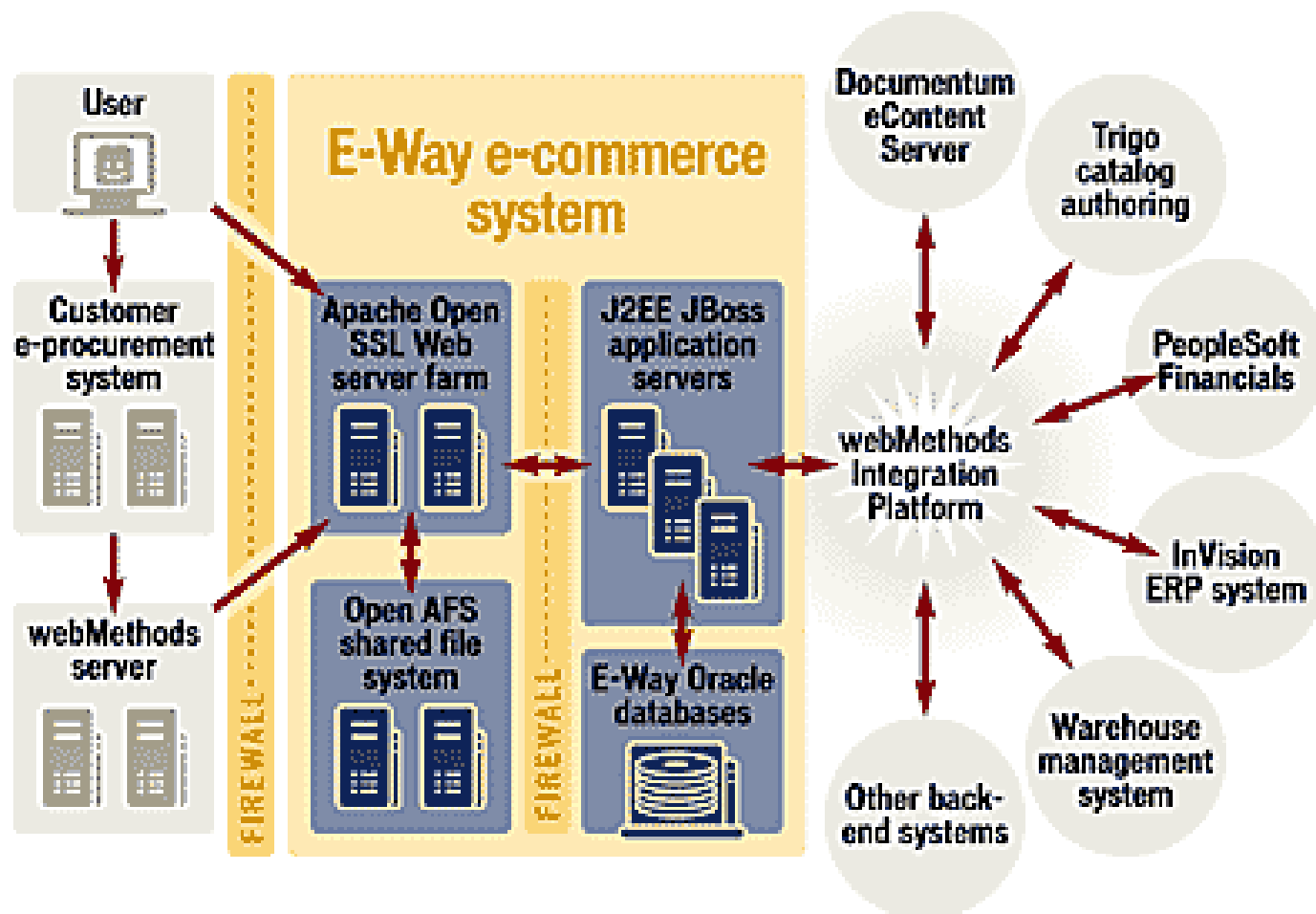
Exemplos reais de arquitetura de Sistemas

- Cases em Telcos: SOA enviroment



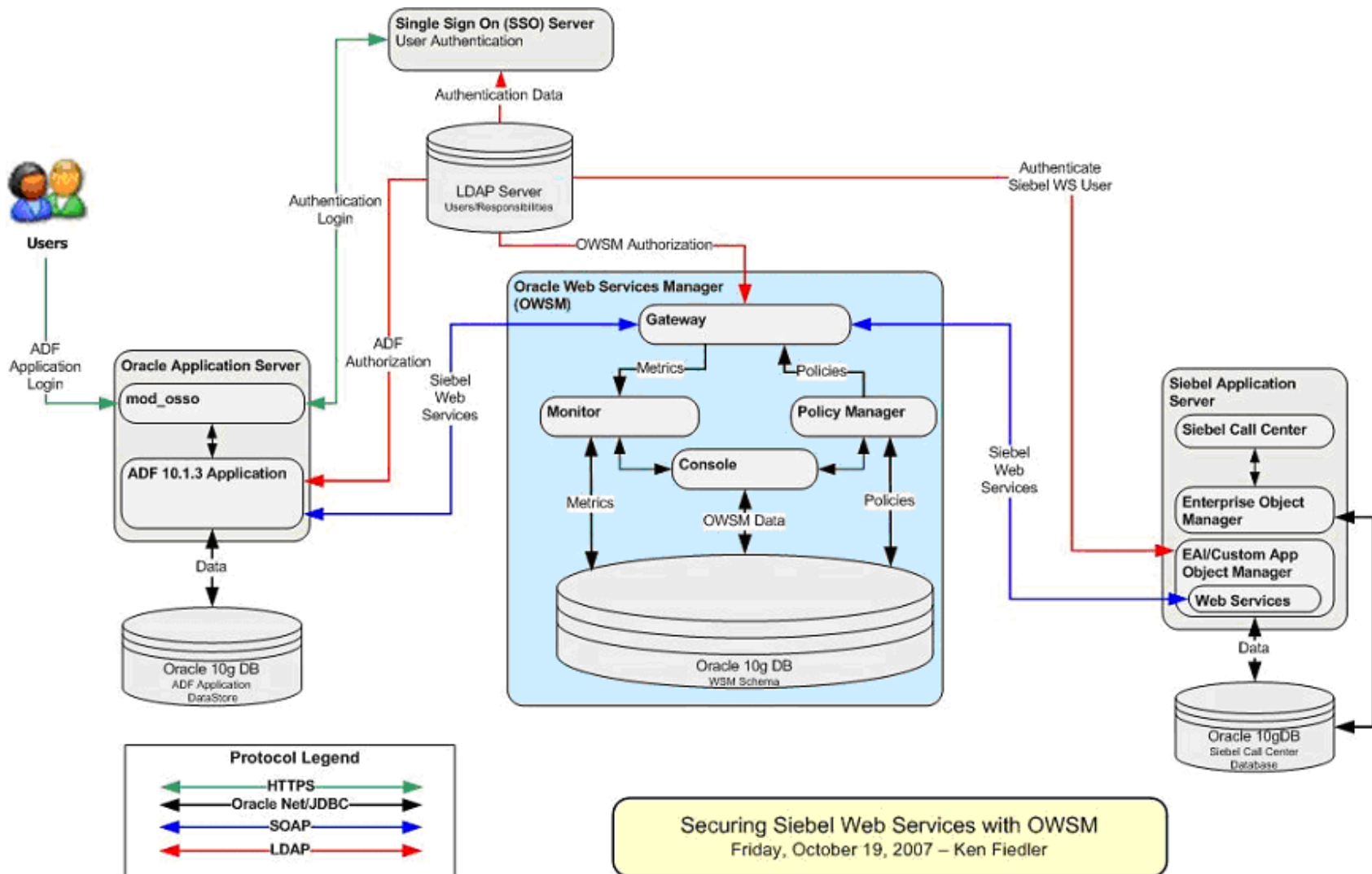
Exemplos reais de arquitetura de Sistemas

- Cases de E-commerce:



Exemplos reais de arquitetura de Sistemas

- Cases de CRM



Exemplos reais de arquitetura de Sistemas

- ERP

