

Modelagem de Serviços de Comércio Eletrônico

Modelo Café

- Abordagem sistemática para a construção de serviços de comércio eletrônico.
- Provê uma especificação incremental de serviços de comércio eletrônico.
- É orientado a bens e serviços comercializados
 - produtos são entidades críticas no processo de comercialização
- O ponto de partida é uma descrição do modelo de negócio a ser provido.

Café: Níveis

- Conceitual
 - define entidades características
- Aplicação
 - instância valores das entidades
- Funcional
 - estratégia de implementação dos serviços
- Execução:
 - requisitos tecnológicos e de implementação

Métodos Formais

- Objetivo
 - Minimizar falhas em servidores de comércio eletrônico
- Estratégias
 - Métodos formais constroem modelos matemáticos do sistema a ser verificado e consideram todas as execuções possíveis desse modelo.
- Modalidades
 - provas manuais
 - provadores semi-automáticos de teoremas
 - sistemas automáticos de verificação de modelos

Sistemas Automáticos de Verificação de Modelos

- Funcionamento
 - Ferramentas verificam se um modelo satisfaz às propriedades especificadas de forma eficiente
- Fases
 - Especificação de modelos
 - Verificação de modelos

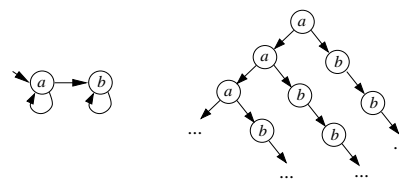
Sistemas Automáticos de Verificação de Modelos

- Desafio
 - Correção
 - Robustez
- Por que é difícil?
 - Variedade de serviços
 - Complexidade dos sistemas
 - Complexidade dos componentes dos sistemas
 - Complexidade das interações entre sistemas e componentes

Sistemas Automáticos de Verificação de Modelos

- Sistemas são modelados como grafos e sequências de execução são caminhos possíveis no grafo.
- Estados no grafo representam configurações do sistema
- Transições correspondem à evolução temporal do servidor modelado.
- Variáveis são modeladas como proposições atômicas, havendo uma variável lógica para cada proposição atômica.

Sistemas Automáticos de Verificação de Modelos



- Três proposições atômicas representadas por a , b e c
- Estados possíveis:
($a=1, b=1, c=1$), ($a=0, b=0, c=1$) e ($a=1, b=0, c=0$)
- Representação simbólica: (a, b, c), (\bar{a}, \bar{b}, c), e (a, \bar{b}, \bar{c})
- $a \vee c$ é verdadeira em todos os três estados
- Transições são representadas por fórmulas lógicas usando dois conjuntos distintos de variáveis.

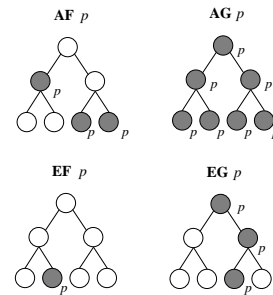
$$\langle a, \neg b, \neg c \rangle \rightarrow \langle a, b, \neg c \rangle \Rightarrow \neg a \wedge \neg b \wedge \neg c \wedge a' \wedge b' \wedge \neg c'$$

- Especificação é feita através de fórmulas em lógica temporal
- Escopo
 - A - Todos os caminhos
 - E - Algum caminho
- Quantificadores
 - F - Algum momento no futuro
 - G - Em todos os momentos
 - X - No próximo estado

Especificação de Propriedades

- **AG** ($req \rightarrow AF\ ack$)
Se req for verdadeiro, então ack será verdadeiro no futuro.
- **EF** ($iniciado \wedge \neg pronto$)
É possível alcançar um estado onde a variável *iniciado* é verdadeira, mas *pronto* não.
- **AG EF restart**
A partir de qualquer estado é possível alcançar o estado de *restart*.
- **AG** ($send \rightarrow A[send\ U\ recv]$)
Sempre que *send* acontecer, *recv* acontecerá no futuro, e até este instante *send* deve permanecer verdadeiro.

Especificação de Propriedades



Especificação de Propriedades Quantitativas

Determinam os tamanhos máximos e mínimos dos caminhos que levam de um conjunto inicial a um conjunto final de estados.

- **MIN**[pedido, compra] e **MAX**[pedido, compra]: respectivamente os tempos mínimo e máximo entre fazer o pedido e completar a compra.
- **MINCOUNT**[pedido, cancelar, compra] e **MAXCOUNT**[pedido, cancelar, compra]: respectivamente o menor e o maior número de vezes que um pedido pode ser cancelado antes de se completar a compra.

Verificação de Modelos

- Tem por objetivo demonstrar que o modelo satisfaz as suas propriedades.
- Estratégias
 - Manual
 - pode ser complicado e lento
 - Simulação
 - Técnica não exaustiva
 - Lógica temporal
 - Busca exaustiva em largura no grafo de estados para determinar quais estados satisfazem ou não as propriedades

SMV Symbolic Model Verifier

- Enfoque simbólico
- Implementação baseada em BDDs (*Binary Decision Diagrams*)
- Verificação é automática
- Quando propriedades são violadas apresenta contra-exemplos
- Complexidade da verificação continua exponencial, mas oferece recursos para evitar explosões combinatórias

Verificação de Modelos Semáforo

- Controla o acesso a uma seção crítica entre dois processos.
- Modelado como uma variável lógica que assume os valores verdadeiro (há um processo na seção crítica) e falso.
- Processo pode estar em um de quatro estados: *naocritico*, *entrando*, *critico* e *saindo*.

Declaração de variáveis

VAR

```
estado : {naocritico, entrando, critico, saindo};
semaforo: boolean;
```

Semáforo Expressão Lógica

Fórmula que define o valor da variável semáforo:

$$((estado = entrando) \wedge semáforo') \vee ((estado = saindo) \wedge \neg semáforo') \vee ((estado = naocritico) \vee (estado = critico) \wedge semáforo \leftrightarrow semáforo')$$

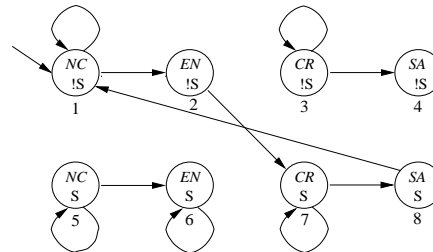
Semáforo Atribuição de Valores de semáforo

```
init(semáforo) := 0;
next(semáforo) :=
case
  estado = entrando : 1;
  estado = saindo   : 0;
  1                  : semáforo;
esac;
```

Semáforo Atribuição de Valores de estado

```
init(estado) := naocritico;
next(estado) :=
case
  estado = naocritico      : {naocritico, entrando};
  estado = entrando & !semáforo: critico;
  estado = critico        : {critico, saindo};
  estado = saindo        : naocritico;
  1                        : estado;
esac;
```

Semáforo Grafo de Estados



Semáforo Instância de Processos

```
MODULE main
VAR
  semáforo : boolean;
  proc1    : process usuario(semáforo);
  proc2    : process usuario(semáforo);
ASSIGN
  init(semáforo) := 0;
```

Semáforo Propriedades

```
SPEC AG !(proc1.estado = critico &
           proc2.estado=critico)
SPEC AG (proc1.estado = entrando ->
         AF proc1.estado = critico)
SPEC AG (proc2.estado = entrando ->
         AF proc2.estado = critico)
```

Semáforo Exemplo de Verificação

State	proc.selector	proc1.estado	proc2.estado	semáforo
1	main	naocritico	naocritico	0
2	proc1	naocritico	naocritico	0
3	main	entrando	naocritico	0
4	proc2	entrando	naocritico	0
5	proc2	entrando	entrando	0
6	proc1	entrando	critico	1
7	proc2	entrando	critico	1
8	proc2	entrando	saindo	1
9	main	entrando	naocritico	0
3	main	entrando	naocritico	0
...

Café: Elementos

- Conceitual: entidades
- Aplicação: itens de produtos, ciclo de vida do item, ações, agentes
- Funcional: serviços, produtos, requisitos funcionais, estratégia de armazenamento
- Execução: arquitetura do servidor, ambiente de execução, modelo de execução, endereçamento, ferramentas

Nível Conceitual

- Define as classes de entidades que vão compor o modelo
- Estima-se que conterà uma única instância para grande parte dos modelos

Café: Nível Conceitual

- Itens - instâncias dos produtos
 - Produtos são agrupados em classes
- Estados - caracterizam os itens
- Ações - modificam o estado dos itens
- Agentes - executam ações durante o processo de aquisição de produtos

Café: Nível Conceitual

- Aplicação de comércio eletrônico pode ser caracterizada por uma tupla $\langle P, I, D, Ag, Ac, S \rangle$ onde:
 - P é o conjunto de produtos da aplicação,
 - I é o conjunto de itens comercializados,
 - D são os domínios possíveis para os itens (seus ciclos de vida),
 - Ag é o conjunto de agentes,
 - Ac é o conjunto de ações e
 - S é o conjunto de serviços

Café: Nível Conceitual Item

- Produtos são conjuntos de itens, ou seja, $i \in I$ significa que $i \in p, p \in P$.
- Os produtos particionam os itens, ou seja, todo item pertence a um e somente um produto.
- Formalmente, $I = \bigcup_{p \in P} p$ e $r \cap s = \emptyset$ para $r, s \in P$.

Café: Nível Conceitual Ação

- Cada ação gera uma transição no grafo de ciclo de vida do item.
- Ação é definida por uma tupla $\langle a, i, tr \rangle \in Ac$, onde
 - $a \in Ag$ é o agente que efetuou a ação,
 - $i \in I$ é o item ao qual a ação foi aplicada e
 - $tr \in D_i X D_i$ é a transição associada à ação.
- Ações são totalmente ordenadas com relação aos momentos de execução.

Café: Nível Conceitual Serviço

- Definidos pela tupla $\langle p, A \rangle$ onde
- $p \in P$ e
- $A = a_1, a_2, \dots$ é uma sequência de ações tal que se
 - $a_i = (d_1, d_2)$, $a_{i+1} = (d_3, d_4)$ então $d_2 = d_3$ e
 - $\forall i, d_i \in D_j$ onde D_j é um domínio de um item de p .

Café: Nível de Aplicação

- Define:
 - os atributos dos produtos
 - os agentes que executam ações
 - os estados de cada item
 - as ações e seus efeitos
 - os tipos de produtos comercializados

Todas as entidades conceituais são instanciadas no nível de aplicação

Café: Nível de Aplicação Definição de Atributos

- Durabilidade
 - intervalo de tempo durante o qual uma instância de um atributo permanece válida
- Tipos de atributos
 - estáticos
 - não mudam como consequência de ações
 - dinâmicos
 - ações podem alterar o seu valor
 - durabilidade variável: sub-grupos

Café: Nível de Aplicação Propriedades

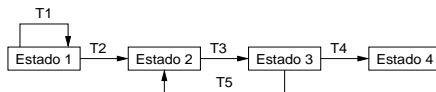
- Estáticas
 - invariantes que sempre devem ser válidas em qualquer estado
- Exemplo: se um produto está disponível, então a quantidade de itens existentes tem que ser maior que zero:
- $AG (produto_disponivel \rightarrow produto_quant > 0)$
- Dinâmicas
 - condições de operação a serem satisfeitas pelo servidor
- Exemplo: uma vez que um item foi comprado ele fatalmente será entregue ao cliente

$AG (item_comprado \rightarrow AF item_entregue)$

Café: Nível de Aplicação Taxonomia de Ações

- Inócuas
 - não afetam o estado de um item
- Temporárias
 - estado do item pode ser restaurado à sua instância original
- Perene
 - irreversíveis

Café: Nível de Aplicação Tipos de Ações



- Inócuas: T1
- Temporárias: T3 e T5
- Perenes: T2 e T4

Café: Nível de Aplicação Especificação

```

VAR
  estado: {1, 2, 3, 4};
  acao: {T1, T2, T3, T4, T5, nula};
ASSIGN
  init(estado) := 1;
  next(estado) := case
    estado = 1 & acao = T1: 1;
    estado = 1 & acao = T2: 2;
    estado = 2 & acao = T3: 3;
    estado = 3 & acao = T4: 4;
    estado = 3 & acao = T5: 2;
  1: estado; -- acoes invalidas sao ignoradas.
esac;
  
```

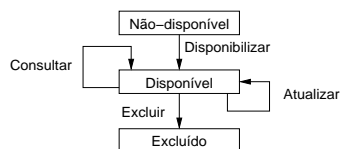
Café: Nível de Aplicação Propriedades

- $AG (estado3 \rightarrow AF estado4)$:
 - Sempre que o estado 3 for alcançado, o estado 4 o será no futuro. Essa propriedade pode significar, por exemplo, que todo item vendido será entregue. Note que a propriedade é falsa no exemplo.
- $\neg E[\neg estado3 \cup estado4]$:
 - Se o estado 4 for alcançado, então passou-se antes pelo estado 3. Isso é expresso através da sua negação, ou seja, não existe um caminho onde *estado3* é falso até que alcance o estado 4.
- $AF estado2$:
 - Sempre se alcançará o estado 2, ou seja, a partir do estado inicial o cliente fará alguma compra (assumindo-se que o estado 1 seja apenas de consulta e os outros estados de compra). Esta propriedade também é falsa.

Café: Biblioteca Digital Nível de Aplicação

- Agentes:
 - provedores e clientes
- Produtos:
 - documentos
- Estados:
 - não-disponível, disponível, excluído
- Ações:
 - disponibilizar, consultar, atualizar, excluir

Café: Biblioteca Digital Nível de Aplicação



Café: Biblioteca Digital Nível de Aplicação

- Atributos:
- tamanho do documento
 - dinâmico
 - data da última modificação
 - dinâmico
 - tipo do documento (p.ex., HTML, texto)
 - estático
 - (chave de acesso)
 - estático

```
VAR
estado: {nao_disponivel, disponivel, excluido};
acao: {disponibilizar, atualizar, consultar, excluir, nula};
ASSIGN
init(estado) := nao_disponivel;
next(estado) := case
estado = nao_disponivel & acao = disponibilizar: disponivel;
estado = disponivel & acao = excluir: excluido;
estado = disponivel & acao = consultar: disponivel;
estado = disponivel & acao = atualizar: disponivel;
1: estado; -- acoes invalidas sao ignoradas.
esac;
```

- A execução de serviços de comércio eletrônico comumente envolvem mais de um produto.
- Há relações de dependência entre os produtos
- Exemplo: serviços de TV a cabo
 - Assinatura e *Pay-per-view* têm ciclos de vida diferentes
 - *Pay-per-view* somente é disponível para assinantes ativos, criando uma relação entre eles

- Modelagem baseada no conceito de aplicação composta.
- Aplicações compostas são caracterizadas pelo número de produtos agregados, que é denominado ordem da aplicação.
 - uma aplicação composta de terceira ordem congrega as aplicações de primeira ordem de três produtos
- Uma aplicação composta de ordem n é definida pela tupla $T = \langle P, I, D, Ag, Ac, S \rangle$ a partir das tuplas de T_1, \dots, T_n de primeira ordem, onde:
 - P é o conjunto de produtos da aplicação composta, onde $\langle p_1, \dots, p_n \rangle \in P$ se e somente se $p_i \in P_i, 1 \leq i \leq n$.
 - I é uma tupla de n itens, um de cada produto que faz parte do modelo composto.

- D é o conjunto dos domínios possíveis para os itens, onde cada instância d do domínio D é uma tupla $\langle d_1, \dots, d_n \rangle$, onde d_i é uma instância do domínio da aplicação unitária T_i .
- Ag é o conjunto de agentes, que resulta da união dos conjuntos de agentes das aplicações unitárias.
- Ac é o conjunto de ações compostas, e que podem ser caracterizadas por uma tupla $\langle a, i, tr \rangle$, onde $a \in Ag, i \in I$ e tr é uma tupla $\langle tr_1, \dots, tr_n \rangle$, onde $tr_i, 1 \leq i \leq n$ é uma transição da tupla T_i ou a transição nula (0).
- S é o conjunto de serviços, definido como união dos serviços dos produtos unitários e pode ser estendido, criando novos serviços que correspondam a execução simultânea de vários desses serviços.

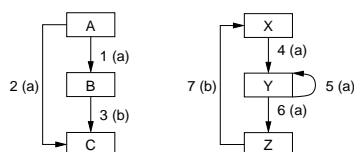
Algoritmo para geração de ciclos de vida compostos

Entrada: D : domínio da aplicação composta
 Ag : agentes da aplicação composta
 $AcUn$: ações das aplicações unitárias
 n : ordem da aplicação composta
Saída: Ac : ações da aplicação composta
Variáveis: $acao$: ação composta
 $acao1$: ação unitária

Algoritmo para geração de ciclos de vida compostos

```
for  $w \in D, z \in D, agente \in Ag$ 
  hatransicao = true
   $acao_{ag} = agente$ 
  for  $j, 1 \leq j \leq n$ 
    if  $\exists acao1 \in AcUn_j$  onde  $acao1_{tr} == w_j \rightarrow z_j$  e  $acao1_{ag} == agente$ 
      then  $acao_{tr_j} = acao1_{tr}$ 
    else if ( $w_j == z_j$ )
      then  $acao_{tr_j} = 0$ 
    else hatransicao = false
  if (hatransicao)
    then  $Ac = Ac \cup acao$ 
```

Exemplo: Aplicação composta dos produtos α e β .

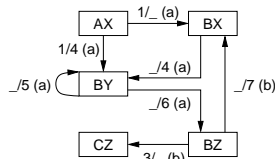


Exemplo: Matriz de compatibilidade do produto composto δ .

	X	Y	Z
A	•		
B	•	•	•
C			•

Café: Nível de Aplicação Múltiplos Ciclos de Vida

Exemplo: Ciclo de vida composto do produto δ .

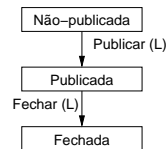


Café: Nível de Aplicação Licitação Eletrônica

- Produtos
 - licitações
 - propostas
- Agentes
 - licitantes
 - proponentes
- Estados da licitação
 - não publicada, publicada e fechada
- Estados da proposta
 - não registrada, registrada, submetida, verificada, classificada e desclassificada

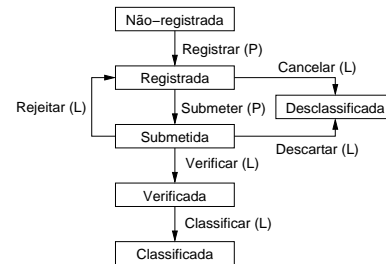
Café: Nível de Aplicação Licitação Eletrônica

Ciclo de vida de uma licitação.



Café: Nível de Aplicação Licitação Eletrônica

Ciclo de vida de uma proposta.



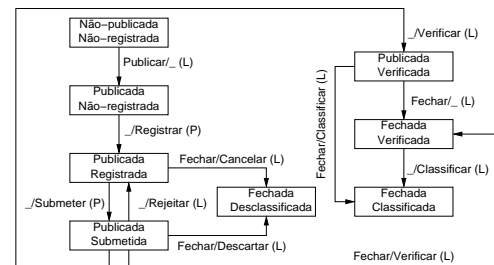
Café: Nível de Aplicação Licitação Eletrônica

Matriz de compatibilidade entre licitação e proposta.

Proposta	Licitação		
	Não publicada	Publicada	Fechada
Não registrada	•	•	
Registrada		•	
Submetida		•	
Verificada		•	•
Classificada			•
Desclassificada			•

Café: Nível de Aplicação Licitação Eletrônica

Ciclo de vida composto de licitação e proposta.



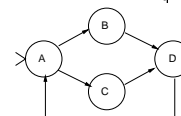
Café: Nível Funcional

Define:

- Serviços
 - Modelo de caminhamento
 - Descrição detalhada
- Requisitos funcionais
 - demandas por componentes
- Estratégia de armazenamento
 - localização dos atributos

Café: Nível Funcional Modelo de Caminhamento

- Ações modificam o estado de itens
- Agentes não executam ações diretamente
- Agentes requisitam serviços, que são seqüências de ações
- Serviços não são requisitados de maneira aleatória
- Modelos de caminhamento definem as seqüências válidas de serviços



Café: Nível Funcional Descrição dos Serviços

- Denominação
- Descrição
- Parâmetros
- Ações
- Atributos
- Classificação
- Requisitos de dados

Café: Nível Funcional Requisitos Funcionais

- Caracterização de demandas em termos de tipos de processamento
 - suporte ao processamento de transações
 - persistência de dados
 - processamento de cadeias de caracteres
- Mapeamento dos serviços nos componentes de acordo com as suas capacidades
 - deve considerar vários compromissos na seleção dos componentes
 - exemplo: SGBDs provêm persistência e atomicidade, mas a um alto custo.

Café: Nível Funcional Estratégia de Armazenamento

- Onde e como os atributos dos produtos serão armazenados
 - classificação dos atributos
 - funcionalidades dos componentes
- Granulação
 - horizontal - adição à cesta de compras
 - vertical - busca por um produto

Café: Biblioteca Digital Nível Funcional

- Modelo de caminhamento
 - contém apenas um serviço para clientes: consulta
- Serviços
 - publicação: ação disponibilizar (perene), altera tamanho e data, atributos pouco voláteis e consistidos periodicamente
 - alteração: ação atualizar (inócua), altera tamanho e data, atributos pouco voláteis e consistidos periodicamente

Café: Biblioteca Digital Nível Funcional

- Serviços
 - consulta: ação consultar (inócua), não altera atributos, que são estáticos e replicáveis
 - exclusão: ação excluir (perene), altera apenas a data, que é definitiva e replicável

Café: Nível Funcional Especificação Funcional

Representação do modelo de caminhamento no SMV.

ASSIGN

```
acao := case
    estado = nao_disponivel: disponibilizar;
    estado = disponivel: {excluir, consultar, atualizar};
    1: nula;
esac;
```

Café: Nível Funcional Verificação Funcional

- SPEC AF estado = excluído.
Todo documento é eventualmente excluído. **Falsa**
- SPEC AG (acao = consultar -> estado = disponivel)
Todo documento consultado está disponível. **Verdadeira**
- SPEC AG (acao = disponibilizar -> AX estado = disponivel)
Se a ação disponibilizar acontecer, o próximo estado é disponível. **Verdadeira**
- SPEC AG (acao = excluir -> AX !EF estado = disponivel)
Uma vez que o documento seja excluído, ele nunca mais estará disponível. **Verdadeira**

Café: Nível Funcional Especificação para Múltiplos Agentes

MODULE loja(estado, acao1, acao2)

VAR

estado: {nao_disponivel, disponivel, excluido};

DEFINE

```
disponibilizando := (acao1 = disponibilizar) |
                    (acao2 = disponibilizar);
excluindo := (acao1 = excluir) | (acao2 = excluir);
consultando := (acao1 = consultar) | (acao2 = consultar);
atualizando := (acao1 = atualizar) | (acao2 = atualizar);
```


Café: Nível Funcional Especificação do Estado da Aplicação

```
ASSIGN
init(estado) := nao_disponivel;
next(estado) := case
    estado = nao_disponivel & disponibilizando: disponivel;
    estado = disponivel & excluindo: excluido;
    estado = disponivel & consultando: disponivel;
    estado = disponivel & atualizando: disponivel;
1: estado; -- acoes invalidas sao ignoradas.
esac;
```

Café: Nível Funcional Especificação de Múltiplos Agentes

```
MODULE cliente(estado, acao)
VAR
    acao: {disponibilizar, atualizar, consultar, excluir,
           nula};

ASSIGN
    acao := case
        estado = nao_disponivel: {disponibilizar, nula};
        estado = disponivel:
            {excluir, consultar, atualizar, nula};
1: nula;
esac;
```

Café: Nível Funcional Especificação de Múltiplos Agentes

```
MODULE main
VAR
    loja1: loja(estado, acao1, acao2);
    cliente1: cliente(estado, acao1);
    cliente2: cliente(estado, acao2);
```

Café: Nível Funcional Propriedades

```
DEFINE
    acao1_inocua := (acao1 = consultar) | (acao1 = nula);
    acao2_inocua := (acao2 = consultar) | (acao2 = nula);
    acoes_ok := acao1_inocua & acao2_inocua;

SPEC AG (acao1 = disponibilizar -> acao2_inocua)
SPEC AG (acao2 = disponibilizar -> acao1_inocua)
SPEC AG (acao1 = excluir -> acao2_inocua)
SPEC AG (acao2 = excluir -> acao1_inocua)
```

Café: Biblioteca Digital Nível Funcional

Requisitos funcionais

- Acesso da Internet:
 - meio básico para acesso aos documentos
- Leitura de arquivos:
 - operação básica de uma requisição
- Escrita de arquivos:
 - necessária para manutenção de documentos

Café: Biblioteca Digital Nível Funcional

- Componente
 - servidor WWW
- Estratégia armazenamento
 - disco local do servidor
 - um arquivo por documento
 - granulação de acesso do serviço

Café: Nível de Execução

- Define:
 - Arquitetura do servidor
 - Ambiente de execução
 - Modelo de execução
 - Endereçamento
 - Ferramentas

Café: Nível de Execução Arquitetura do Servidor

- define a natureza dos componentes de software a ser utilizados
- apresenta a justificativa para a utilização dos componentes
- exemplo
 - requisito: armazenamento persistente
 - resultado: deve-se utilizar um servidor de banco de dados
- definição deve considerar não apenas aspectos funcionais, mas custo e plataforma existente
- necessário definir os protocolos de comunicação entre componentes

Café: Nível de Execução Ambiente

- Descreve:
 - interligação entre os componentes
 - interface com o cliente
- Conteúdo:
 - paradigmas de implementação
RPC, cliente-servidor, mensagens
 - primitivas de sistema
suporte TCP/IP, fork, rsh, RMI

Café: Nível de Execução Modelo de Execução

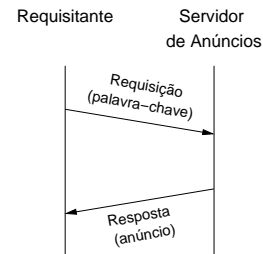
Descreve a interação entre componentes

- diagrama temporal
- abstração de mensagem
 - finalidade
 - protocolo utilizado
 - emissor e receptor
 - conteúdo
 - formato

Café: Nível de Execução Servidor de Anúncios

- Mensagens
 - requisição de anúncio
 - anúncio
- Comunicação através de TCP/IP
- Requisição
 - parâmetros de geração de anúncios
- Anúncio
 - arquivo binário da imagem

Café: Nível de Execução Servidor de Anúncios



Café: Nível de Execução Endereçamento

- Serviços devem ser acessados de forma determinística
- URLs:
 - identificação do servidor
 - especificação do serviço
 - parâmetros de execução do serviço
 - exemplo:
- <http://anuncio.com.br/banner/?e-speed>

Café: Nível de Execução Ferramentas

Critérios:

- Uso na implementação de serviços
- Custo
- Componentes ou geradores de programas:
 - geradores de interface
 - SGBD
 - servidor de transações
 - servidor WWW

Café: Biblioteca Digital Nível de Execução

- Arquitetura do servidor
 - um único componente de software
- Ambiente de execução
 - conexão TCP/IP
 - IP válido e domínio
- Modelo de execução
 - Requisições
 - especifica o documento
 - Respostas
 - código de resultado
 - documento desejado

Café: Biblioteca Digital Nível de Execução

- Endereçamento
 - nome do documento (incluindo diretórios) antecedido pelo nome do servidor
- Ferramentas
 - editores documentos
 - gerenciadores de sistemas de arquivo

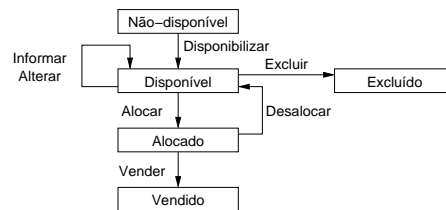
- Livraria virtual que permite aos clientes procurar livros e comprá-los eletronicamente através da Internet.
 - Objetivo: facilitar e agilizar as compras
 - Cadastro de clientes: apenas no momento da transação propriamente dita
- * Mata próxima a Belo Horizonte.

Jambreiro é uma loja virtual cujo único produto comercializado são livros, que são adquiridos utilizando cartão de crédito. Esses livros podem ser encontrados através de um mecanismo de busca por título ou autor, ou através de categorias de navegação estabelecidas previamente. O cliente pode colocar os livros escolhidos em uma "cesta" para posterior compra. A entrega dos livros é feita por correios e o custo deve ser calculado de acordo com o peso do(s) livro(s) adquiridos.

Jambreiro: Aplicação

- Agentes: vendedores e compradores
- Produtos: livros
 - atributos: info, preço, cliente
- Estados: indisponível, disponível, alocado, vendido, excluído
- Ações: disponibilizar, alocar, vender, desalocar, excluir, informar, alterar

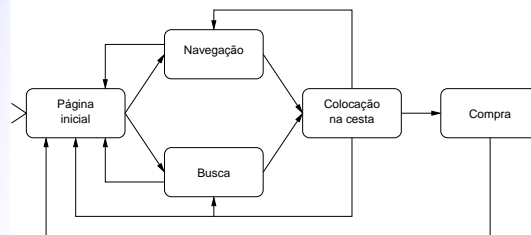
Jambreiro: Aplicação



Jambreiro: Funcional

- Serviços
 - página inicial
 - navegação
 - busca
 - colocação na cesta
 - compra

Jambreiro: Funcional Modelo de Caminhamento



Jambreiro: Funcional Notação

- Modo de acesso:
 - (L): leitura, (E): escrita
- Volatilidade:
 - EST: estático, PVL: pouco volátil, MVL: muito volátil, DEF: definitivo
- Forma de compartilhamento:
 - REP: replicável, CMP: compartilhado, MEX: mutuamente exclusivo

Jambreiro: Funcional Navegação

- Descrição: disponibilizar informações sobre produtos pré-classificados (p.ex., editora, gênero)
- Ação: informar
- Atributos: info(L), preço(L)
- Classificação: inócu
- Volatilidade: info-EST, preço-PVL
- Compartilhamento: info-REP, preço-CMP
- Processamento: consultas atributo-valor, leitura de arquivos, processamento de cadeias de caracteres

Jambreiro: Funcional Busca

- Descrição: disponibilizar informações sobre produtos a partir de consultas dos usuários sobre info (e.g., busca de padrões)
- Ação: informar
- Atributos: info(L), preço(L)
- Classificação: incóua
- Volatilidade: info-EST, preço-PVL
- Compartilhamento: info-REP, preço-CMP
- Processamento: consultas atributo-valor, leitura de arquivos, processamento de cadeias de caracteres

Jambreiro: Funcional Colocar na Cesta

- Descrição: registrar opção de compra de cliente para posterior aquisição
- Ação: alocar
- Classificação: temporária
- Atributos: info (L), preço (L), cliente (E)
- Volatilidade: info-EST, preço-PVL, cliente-MVL
- Compartilhamento: info-REP, preço-CMP, cliente-MEX
- Processamento: processamento de cadeias de caracteres, suporte a transações, armazenamento temporário, controle de acesso

Jambreiro: Funcional Requisitos

- aritmética
- processamento de cadeias de caracteres
- leitura arquivos
- suporte a transações
- armazenamento temporário
- consultas atributo-valor
- controle de acesso
- armazenamento persistente

Jambreiro: Funcional Serviço de Entrada

- Servidor WWW:
 - lê arquivo contendo página inicial
- Servidor Transações:
- Servidor Banco de Dados:

Jambreiro: Funcional Serviço de Navegação

- Servidor WWW:
 - repassa requisição
- Servidor Transações:
 - lê arquivo template, gera página
- Servidor Banco de Dados:
 - consulta atributo-valor

Jambreiro: Funcional Serviço de Busca

- Servidor WWW:
 - repassa requisição
- Servidor Transações:
 - lê arquivo template, gera página
- Servidor Banco de Dados:
 - consulta atributo-valor

Jambreiro: Funcional Serviço de Colocação na Cesta

- Servidor WWW:
 - controle de acesso
- Servidor de Transações:
 - manutenção da cesta, gera página, lê arquivo template
- Servidor Banco de Dados:
 - consulta dados cliente

Jambreiro: Funcional Serviço de Compra

- Servidor WWW:
 - controle de acesso
- Servidor Transações:
 - gera página
- Servidor Banco de Dados:
 - armazena dados transação

Jambreiro: Funcional Estratégia de Armazenamento

- Produto
 - armazenamento persistente no SGBD
 - replicáveis no servidor de transações
- Cliente
 - armazenamento persistente no SGBD
- Esqueleto de página
 - servidor de transações
- Imagem
 - servidor WWW

Jambreiro: Execução

- Arquitetura do servidor
 - arquitetura de três níveis
- Ambiente de execução
 - WWW, ODBC, cgi-bin
- Endereçamento:
 - URL, atributos-valor e tuplas relacionais
- Protocolos:
 - baseados em HTTP e CGI
- Ferramentas:
 - servidor WWW, SGBD

Jambreiro Arquitetura do Servidor

Localização de dados

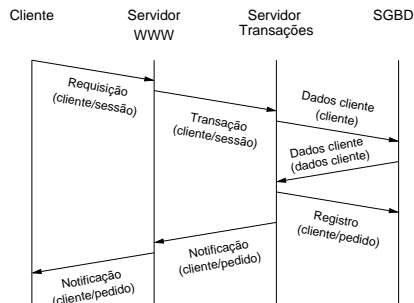
- Produto
 - armazenamento persistente no SGBD
- Cliente
 - armazenamento persistente no SGBD e replicado no servidor de aplicações
- Esqueleto de página
 - servidor de transações, pois são utilizados por requisições dinâmicas
- Imagem
 - armazenadas no servidor WWW

Jambreiro Ambiente de Execução

- Cliente - WWW:
 - HTTP
- WWW-Transação:
 - CGI-BIN+socket
- Transação-SGBD:
 - ODBC

Jambreiro Modelo de Execução

Diagrama temporal do serviço de compra



Jambreiro Endereçamento

- HTTP: URL `http://jambreiro/servico/prm1=val1&...&prmN=valN`
- CGI-BIN: par atributo-valor
`SERVER="jambreiro"`
`PATH="servico"`
`QUERY_STRING="prm1=val1&...&prmN=valN"`
- ODBC: SQL `select * from db where prm1=val1 and ... and prmN=valN;`

Jambreiro Execução de Compra

- Tipos de mensagens:
 - Requisição de compra
 - Transação de compra
 - Obtenção de dados de cliente
 - Envio de dados de cliente
 - Gravação da sessão
 - Envio de confirmação

Jambreiro Execução de Compra

Requisição de Compra

- Finalidade: cliente efetua compra
- Trajetos: cliente para servidor WWW
- Formato:
 - `http://loja/compra?client_id=jj1&session_id=1`
- Conteúdo:
 - client_id
 - session_id

Jambreiro Execução de Compra

Transação de Compra

- Finalidade: servidor WWW repassa req.
- Trajeto: WWW para transações
- Formato:
 - QUERY_STRING="client_id=jj1&session_id=1"
 - PATH="/compra?"
- Conteúdo:
 - client_id
 - session_id

Jambreiro Execução de Compra

Envio de dados do cliente

- Finalidade: notificação dos dados do cliente
- Trajeto: SGBD para transações
- Formato:
 - atr1 val1
- Conteúdo:
 - client_id
 - nome
 - endereço
 - cartão de crédito

Jambreiro Execução de Compra

Gravação da Sessão

- Finalidade: registra compra
- Trajeto: Transações para SGBD
- Formato:
 - insert client_id, pedido into compras
- Conteúdo:
 - client_id: identificador do cliente
 - pedido: dados do pedido atendido

Jambreiro Execução de Compra

Envio da confirmação

- Finalidade: informa cliente da compra
- Trajeto: Transações para cliente (via WWW)
- Formato:
 - HTML
- Conteúdo:
 - order_id
 - purchase_id

Jambreiro: Execução Ferramentas

- Apache: servidor WWW
- MySQL: SGBD
- Perl: linguagem de script