INF016 – Arquitetura de Software

01 - Introdução

Sandro Santos Andrade
sandroandrade@ifba.edu.br

Instituto Federal de Educação, Ciência e Tecnologia da Bahia
Departamento de Tecnologia Eletro-Eletrônica
Graduação Tecnológica em Análise e Desenvolvimento de Sistemas

Introdução

- A área de Arquitetura de Software estuda como sistemas de *software* são projetados e construídos
- Arquitetura de Software:
  - Conjunto formado pelas principais decisões de projeto tomadas durante seu desenvolvimento e qualquer evolução subsequente
- Desenvolvimento de *software* centrado em arquiteturas
- Qualidade de projeto → Qualidade de Software
- Famílias de produtos de *software*

Software e Construção Civil

- Analogia forte e de fácil compreensão
- As fases são similares (requisitos, projeto, etc)
- Outras similaridades:
  - Projeto arquitetural com foco nas necessidades dos usuários
  - Permite especialização de trabalho
  - Planos e progressos podem ser avaliados em pontos intermediários
- Mas não é só isso ...

Software e Construção Civil

1) Toda construção tem uma arquitetura, separada, porém relacionada, à estrutura física

- Esta arquitetura pode ser descrita, discutida e comparada com as de outras construções
- A arquitetura antecipadamente projetada pode ser comparada com a arquitetura resultante do processo de construção
- De forma similar, a arquitetura de um *software* existe de forma independente, porém relacionada, ao código-fonte que a implementa

Software e Construção Civil

"Prática é a contemplação frequente e contínua do modo de execução de um trabalho, ou da mera operação das mãos, que converte o material da melhor e mais imediata forma possível"

Marcus Vitruvius Pollio, século 1 D.C.

Software e Construção Civil

2) Propriedades das estruturas são induzidas pelo projeto das suas arquiteturas:

- Castelo medieval: paredes altas e espessas e janelas estreitas, se existentes. Induz propriedades defensivas
- Propriedades de um *software*, como resiliência a tipos particulares de ataques, são determinadas pelo projeto de suas arquiteturas

Software e Construção Civil

- Objetivos de uma boa arquitetura:
  - Força: fundações para um assoalho sólido, escolha apropriada de materiais sem economia
  - Utilidade: distribuição sensata das partes, com seus propósitos devidamente atendidos e situação apropriada
  - Beleza: aparência agradável, boa percepção do “todo” e dimensões proporcionais entre as partes

Software e Construção Civil

3) Reconhecimento do papel distinto e característico do arquiteto – pessoa que cria a arquitetura

- Exige ampla formação:
  - Aspectos de engenharia
  - Senso apurado de estética
  - Conhecer o modo como as pessoas trabalham, comem, brincam e moram ajuda a projetar construções satisfatórias e que funcionam bem ao longo das estações e dos anos
  - Habilidades simples de programação não são suficientes para a criação de sistemas complexos que efetivamente funcionam

Software e Construção Civil

"Um arquiteto deve ser engenhoso e competente na aquisição do conhecimento. Não será um mestre perfeito se for deficiente em uma dessas qualidades. Deve ser bom escritor, hábil relator, versado em ótica e geometria, especialista em números, familiarizado com história, informado sobre os princípios da filosofia natural e moral, um tanto músico, não ignorante das ciências da lei e da física, nem dos movimentos, leis e inter-relacionamentos dos corpos celestes"

Vitruvious, I, 1, 3

Software e Construção Civil

4) O processo não é tão importante quanto a arquitetura

- Isso não quer dizer que o processo não é importante, somente que ele não é garantia de sucesso
- O processo existe para servir um fim – o projeto e a qualidade da infra-estrutura – não para ser um fim em si próprio

Software e Construção Civil

5) A arquitetura (de *software*) amadureceu, ao longo dos anos, como uma disciplina

- Uma base de conhecimentos está disponível, capturando as experiências e lições de projeto prévios
- Foco no reuso de conhecimento, de projeto de sub-sistemas e de ferramentas
- Benefício de uso de materiais, partes e tamanhos padronizados

Software e Construção Civil

- Estilos Arquiteturais:
  - Vila Romana
  - Catedral Gótica
  - Estilo fazenda
  - Chalé Suiço
  - Arranha-céu
- Tentam encontrar um conjunto comum de requisitos e acomodar as restrições de topologia local, clima e materiais, ferramentas e mão-de-obra disponíveis
- Um estilo coloca restrições ao desenvolvimento, o que leva a qualidades particulares desejáveis

Software e Construção Civil

- Limitações da analogia:
  - Conhecemos muito sobre prédios e não tanto sobre *software*
  - Natureza essencial dos materiais totalmente diferente
  - O *software* é mais “maleável” do que os materiais físicos de construção
  - A indústria da construção civil é mais consolidada
  - O fase de implantação não existe na construção civil
  - Caráter extremamente dinâmico do *software*

Software e Construção Civil

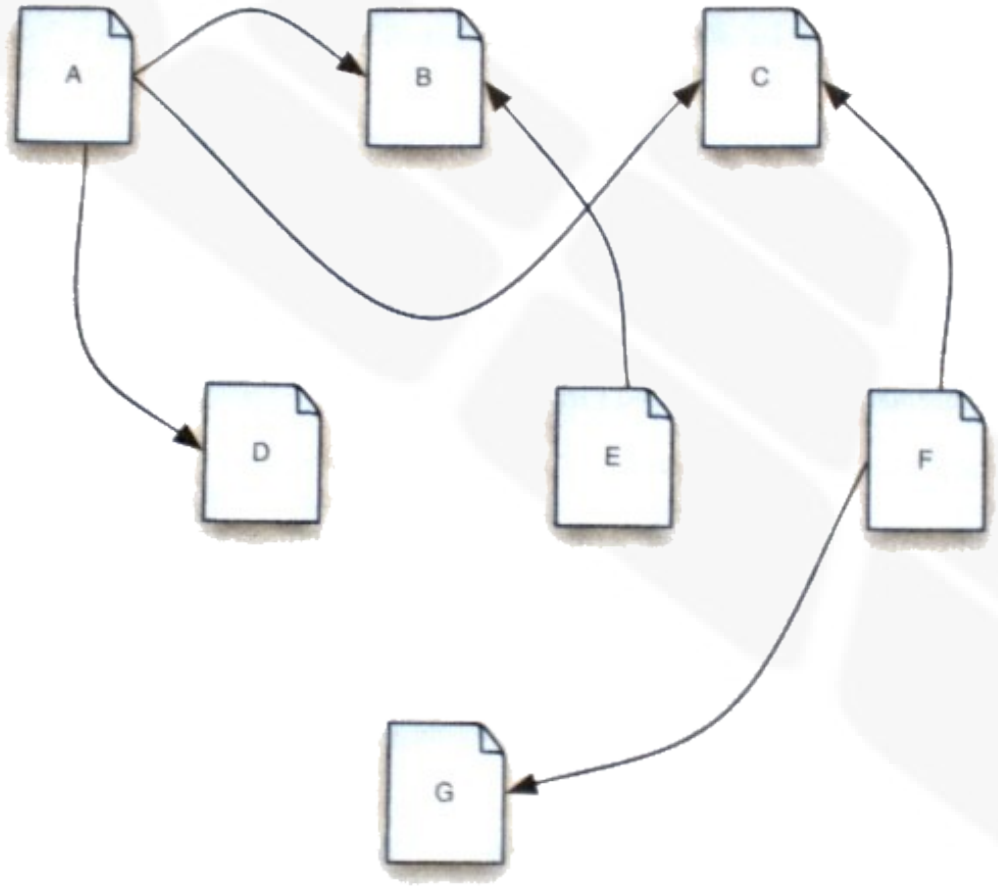
- Resumindo:
  - A arquitetura do *software* deve ser o centro do projeto e desenvolvimento de sistemas, mais importante que o processo, análise e até mesmo programação
  - Ao dar proeminência à arquitetura obtém-se: controle intelectual, integridade conceitual, base adequada e efetiva para reuso, comunicação efetiva no projeto e gerenciamento de um conjunto de sistemas variantes, porém relacionados
  - O foco na arquitetura deve estar presente em **todas** as fases do projeto

Exemplos

- Exemplo 1: Arquitetura da web:
  - O que é a web ? Como ela é construída ? Como você projetaria um *software* para um site de comércio eletrônico ?
  - A arquitetura do sistema fornece o vocabulário e os meios para responder as questões acima, em particular o estilo arquitetural adotado para a web

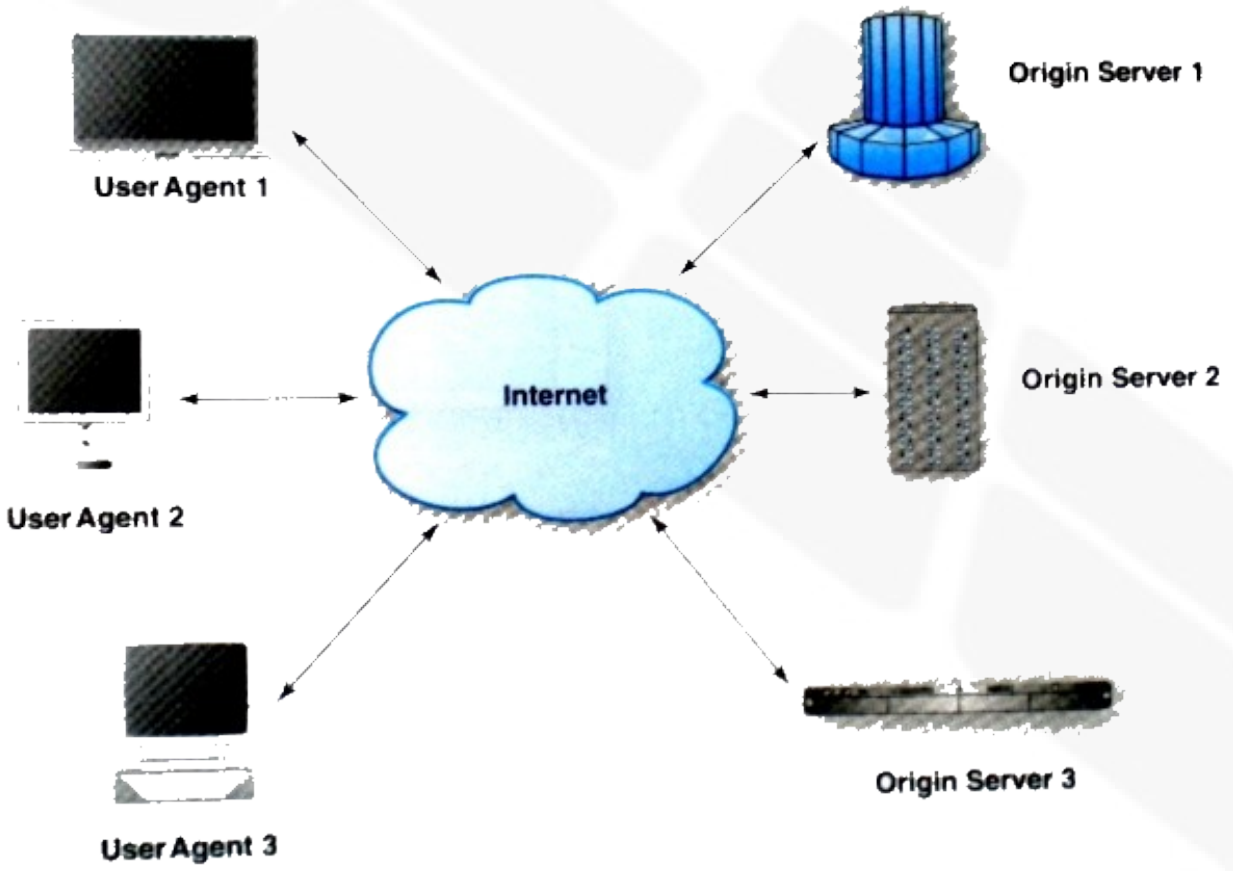
Exemplos

- Visão do usuário: conjunto dinâmico de relacionamentos entre coleções de informação



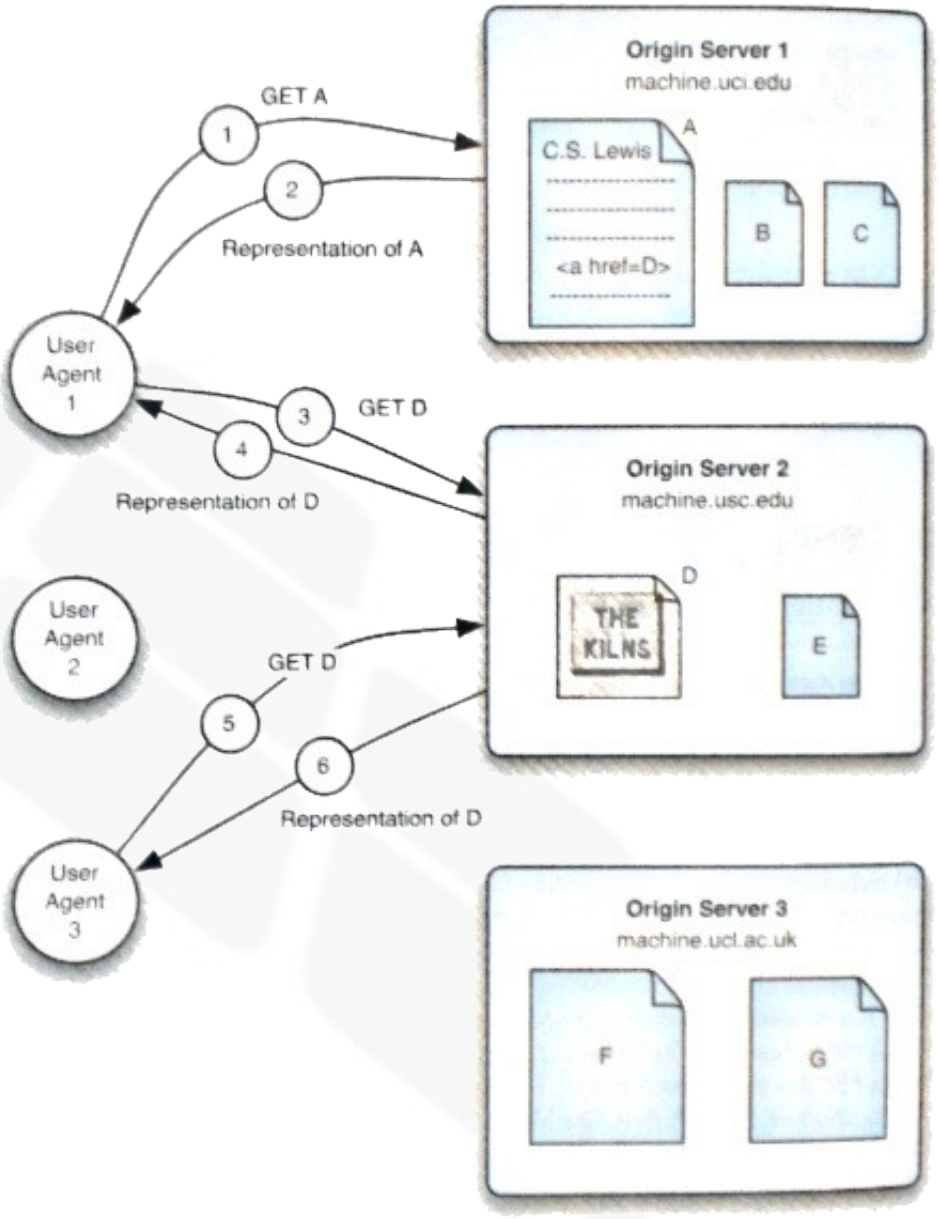
Exemplos

- Visão de rede: coleção de máquinas independentemente apropriadas e operadas, que se comunicam via rede



Exemplos

- Visão do desenvolvedor: coleção de programas independentemente desenvolvidos que se comunicam através dos padrões HTTP, URI, MIME e HTML



Exemplos

- Estas visões não explicam como a web funciona
- Uma estratégia melhor é apresentar um conjunto de definições e restrições que caracterizam a web:
  - Coleção de *resources*, identificados unicamente por uma URL
  - Cada *resource* denota uma informação, como um documento, imagem, serviço, coleção de outros *resources*, etc
  - URL's podem ser utilizadas para determinar a identidade da máquina que contém o *resource*
  - Toda comunicação é iniciada pelos clientes (*user agents*), realizando requisições aos servidores

Exemplos

- Uma estratégia melhor é apresentar um conjunto de definições e restrições que caracterizam a web:
  - *Resources* podem ser manipulados através de suas *representações*. O HTML é a linguagem de representação mais comum da web
  - Toda comunicação entre clientes e servidores é realizada através de um protocolo extremamente simples (HTTP), com poucas primitivas, tais como GET e POST
  - Toda comunicação entre clientes e servidores é *context-free*, ou seja, o servidor responde à requisição baseando-se somente na informação presente na própria requisição. Nenhum histórico de operações é mantido

Exemplos

- Exemplo 2: *Shell Script*

ls invoices | grep -e August | sort

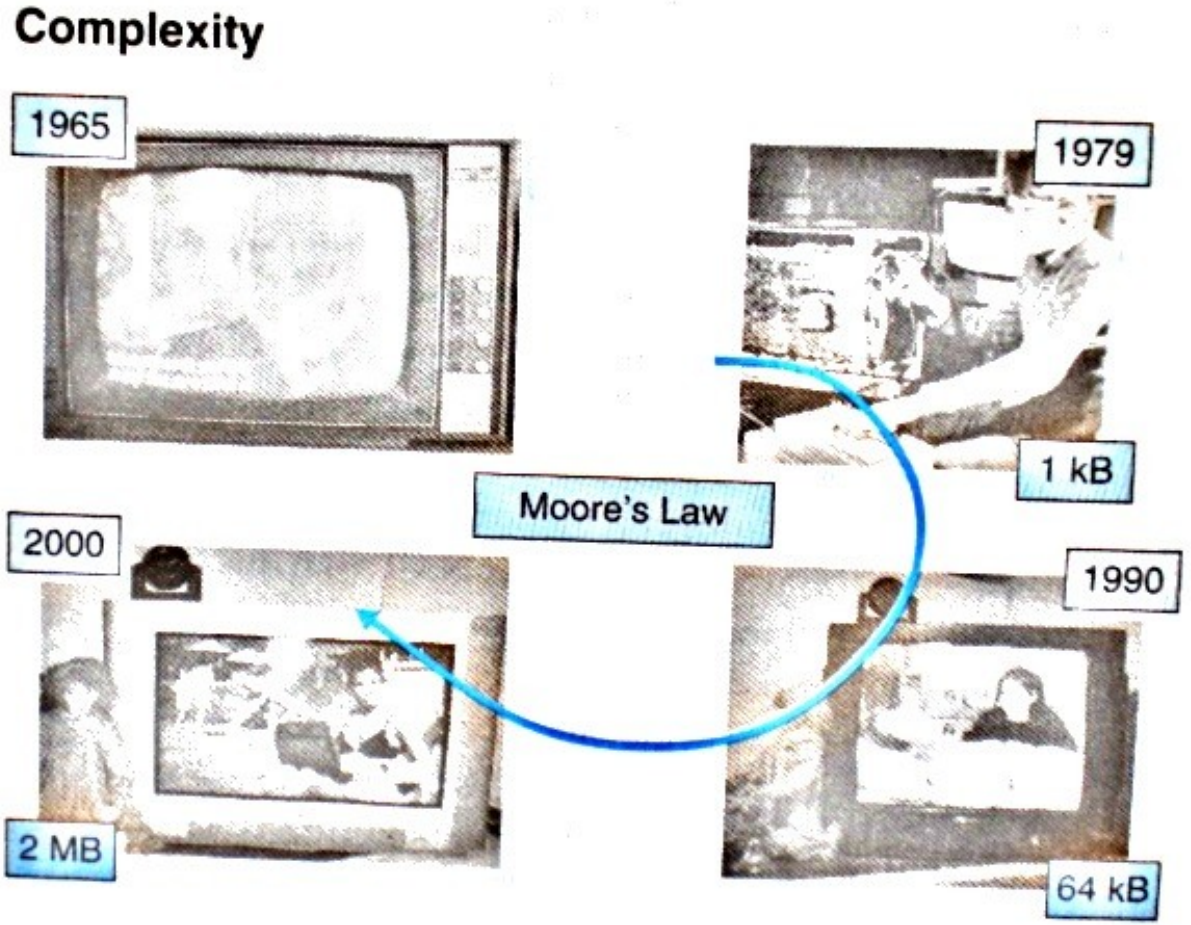
  - Um *filtro* é um programa que recebe um fluxo de caracteres como entrada e produz um fluxo de caracteres como saída. Filtros podem ser parametrizados
  - Um *pipe* é uma forma de conectar dois filtros, onde a saída do primeiro filtro é conectada à entrada do segundo
  - Conhecendo os filtros e *pipes* utilizados pode-se facilmente compreender o programa e criar outros
  - O conjunto particular de regras aqui aplicado define um estilo arquitetural conhecido como *Pipe-and-Filter*
  - Pode ser utilizado em qualquer sistema

Exemplos

- Exemplo 3: Linhas de Produto
  - Famílias de produto são conjuntos de programas independentes que possuem um alto potencial de compartilhamento de estrutura e componentes constituintes
  - Ex: HD TV 35” com *DVD player* com sinal ATSC, HD TV 35” sem *DVD player* com sinal ATSC, HD TV 35” com *DVD player* com sinal DVB-T
  - Reutilizar estruturas, comportamentos e implementações simplifica o desenvolvimento, reduz prazos e custos e melhora a confiabilidade geral do sistema
  - Arquiteturas de *software* são abstrações essenciais para o gerenciamento de variações e de pontos em comum

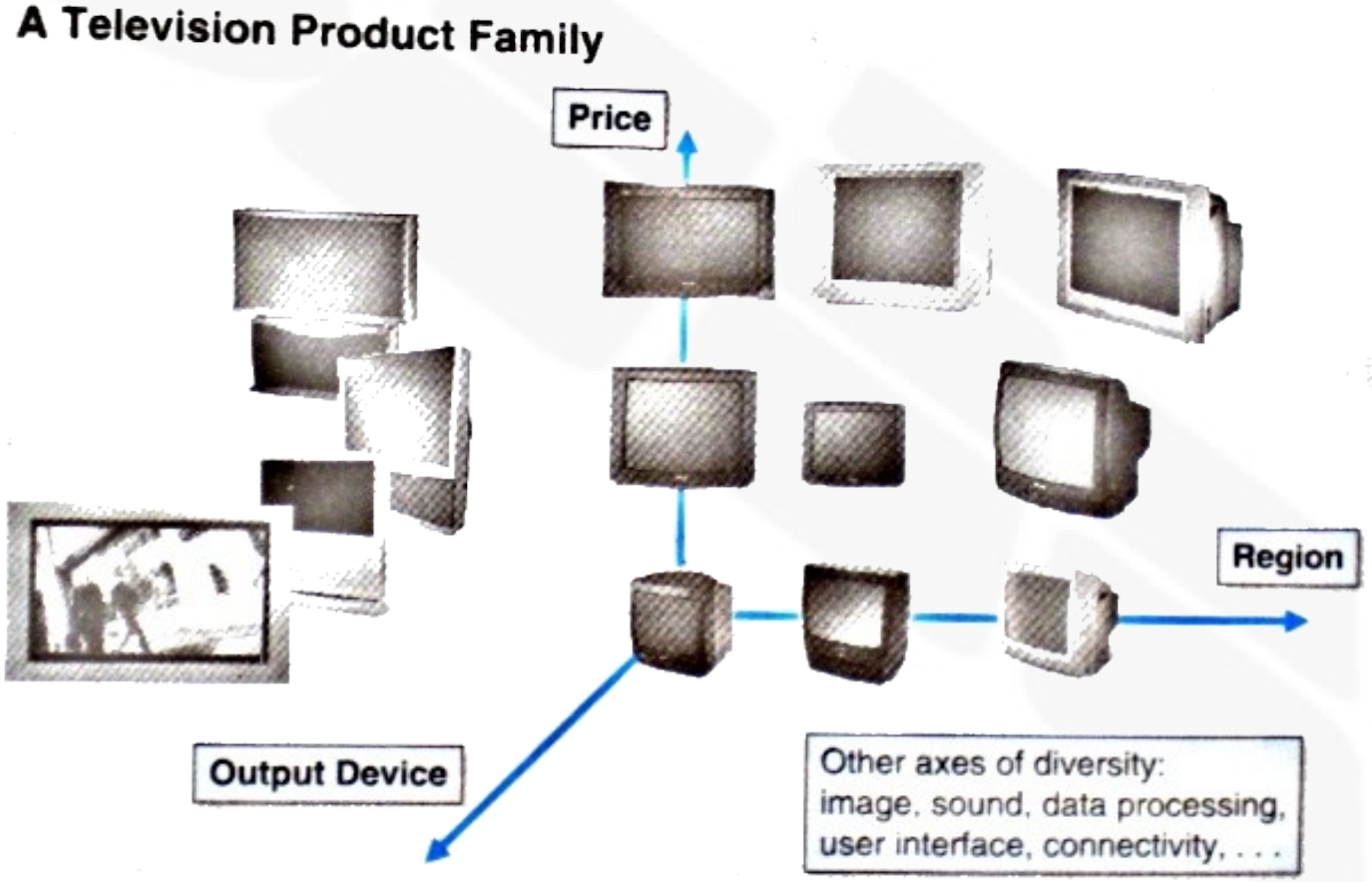
Exemplos

- Linha de produtos da Philips (iniciada no final da década de 90)



Exemplos

- Linha de produtos da Philips
  - Metodologia arquitetural: *Koala*

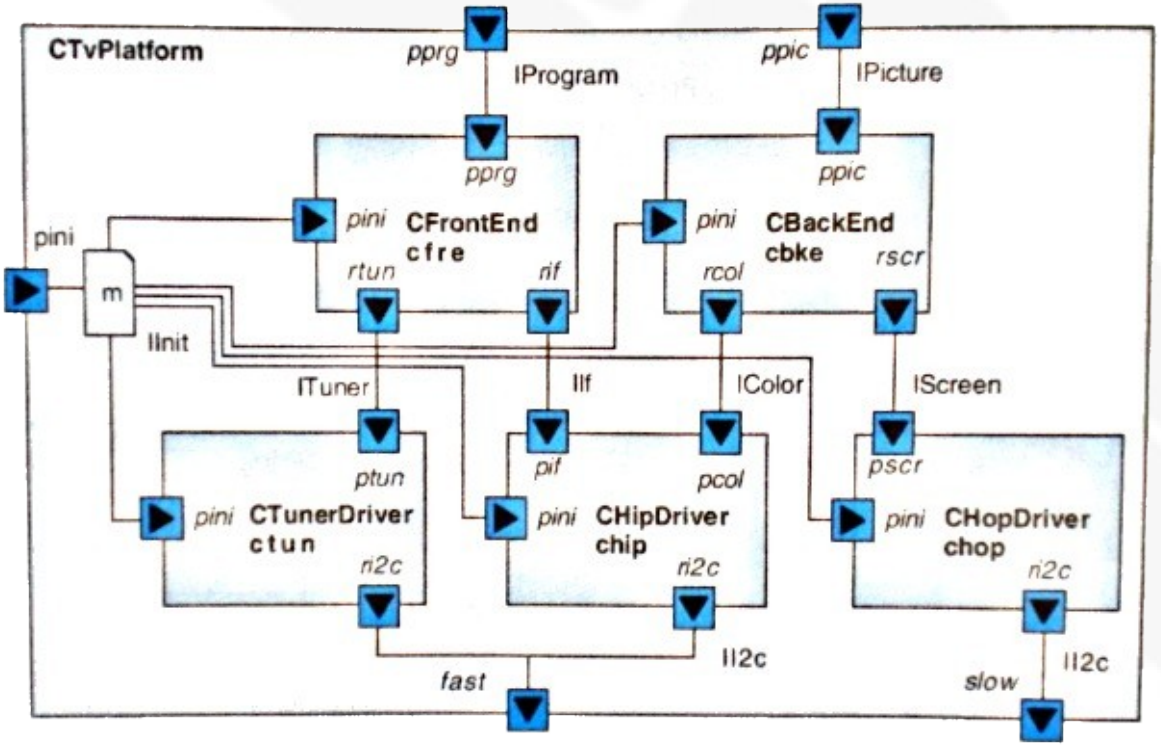


Exemplos

- *Koala*:
  - Modela e implementa o *software* como uma coleção de componentes que interagem entre si
  - Cada componente exporta um conjunto de serviços através de um conjunto de *provided interfaces*
  - Cada componente explicitamente define suas dependências com o ambiente (*hardware* ou *software*) através de um conjunto de *required interfaces*

Exemplos

- Arquitetura exemplo de uma plataforma de TV da Philips:
  - Compatibilidade de interfaces
  - *Composite*

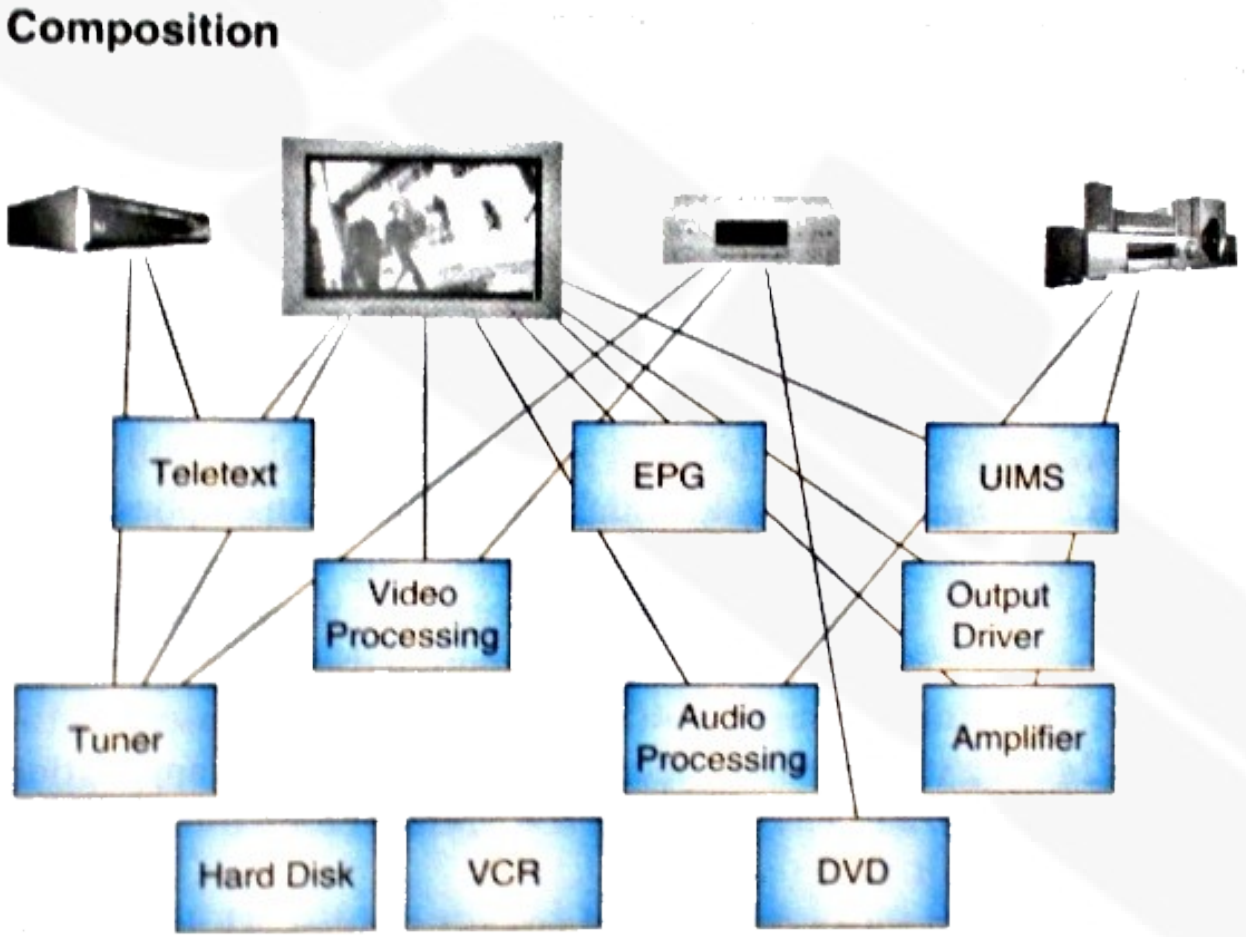


Exemplos

- Koala: mecanismos para gerenciamento da variabilidade:
  - *Diversity interfaces*: mecanismo para parametrizar um componente. Permite que um componente importe propriedades específicas da configuração a partir de elementos do Koala que implementam esta interface. São externos ao componente
  - *Switches*: elemento de conexão que permite que um componente interaja com apenas um dentre um conjunto de componentes, dependendo do valor de um parâmetro obtido em *run-time*
  - *Optional interfaces*: provê ou requer funcionalidades presentes em apenas alguns produtos da família

Exemplos

- Koala: população de produtos



Exemplos

- Famílias de produtos demandam mudanças nos processos e práticas organizacionais
- A abordagem padrão de desenvolvimento não suporta linhas de produto de forma eficiente
- O *Koala* é a manifestação concreta da experiência corporativa, conhecimentos e vantagens competitivas da empresa
- Arquiteturas de *software* evitam a dependência de uma estrutura social (permanência de funcionários, etc) e suportam o alcance de níveis maiores de produtividade

INF016 – Arquitetura de Software

01 - Introdução

Sandro Santos Andrade
sandroandrade@ifba.edu.br

Instituto Federal de Educação, Ciência e Tecnologia da Bahia
Departamento de Tecnologia Eletro-Eletrônica
Graduação Tecnológica em Análise e Desenvolvimento de Sistemas