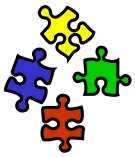
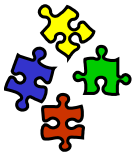


ADLs

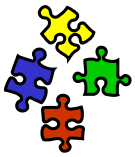
- Em geral cada ADL oferece capacidades específicas
  - **AESOP**: permite o uso de estilos arquiteturais
  - **ADAGE**: permite a descrição de frameworks arquiteturais para sistemas de aviação
  - **C2**: permite a descrição de sistemas de interface com usuário usando um estilo baseado em mensagem
  - **Wright**: suporta a especificação e análise de interações entre componentes arquiteturais



ACME

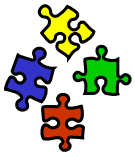
- Architecture Description Language
- Motivação:
  - Proliferação de ADLs
  - Necessidade de um formato “padrão”

Linguagem para Intercâmbio de Descrições Arquiteturais = ACME



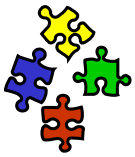
ACME
Objetivos Principais

- Permitir a integração de diferentes ferramentas oferecendo uma forma comum de intercâmbio arquitetural
- Permitir o desenvolvimento de arquitetura compatível com múltiplas ADLs
- Esclarecer o relacionamento entre diferentes ADLs



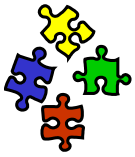
ACME
Objetivos Secundários

- Oferecer um esquema para representação de arquiteturas que irá permitir o desenvolvimento de novas ferramentas para análise e visualização de estruturas arquiteturais
- Oferecer uma fundamentação para o desenvolvimento de ADLs
- Servir como veículo para criar convenções e padrões para informação arquitetural
- Oferecer descrições expressivas que são fáceis de ler e escrever



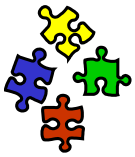
ACME

- Formato de Intercâmbio Genérico + facilidades de anotação para informações adicionais específicas de ADLs



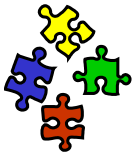
ACME
Características

- ONTOLOGIA arquitetural: com sete elementos para design arquitetural
- Mecanismo de ANOTAÇÃO: suportando associação de informação não estrutural usando sublinguagens definidas externamente
- Mecanismo de TEMPLATE
- Framework de semântica aberta



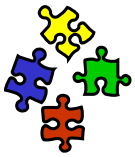
ACME
Elementos

- Sete tipos de entidades para representação arquitetural:
  - Componentes
  - Conectores
  - Sistemas
  - Portas
  - Papéis
  - Representações
  - Rep-Maps



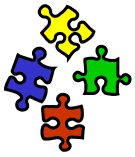
ACME
Componentes

- Elemento computacional primário
- Exemplos típicos:
  - Clientes
  - Servidores
  - Filtros
  - Objetos



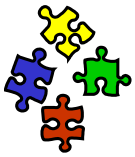
ACME
Conectores

- Representam interações entre componentes
- Mediam a comunicação e coordenação entre componentes
- Exemplos:
  - Pipes
  - RPC
  - Eventos
  - Protocolos Cliente-Servidor



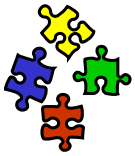
ACME
Sistemas

- Representam Configurações de componentes e conectores



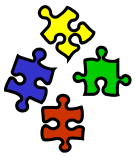
ACME
Interfaces

- Portas => Componentes
- Papéis => Conectores



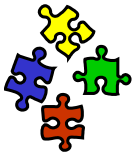
ACME
Portas

- Interfaces dos componentes são definidas por **PORTAS**
- Cada porta identifica um ponto de interação entre o componente e seu ambiente
- Um componente pode prover múltiplas interfaces usando diferentes **tipos de portas**

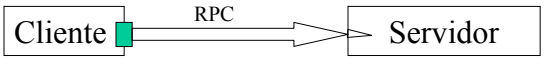


ACME
Papéis

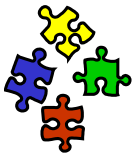
- Interfaces dos Conectores são definidas por papéis
- Cada papel de um conector define um participante da interação representada pelo conector
- Conectores Binários têm dois papéis:
  - Em RPC: Caller e Callee
  - Em um Pipe: Leitura e Escrita
- Outros tipos de conectores podem ter mais que dois papéis. Exemplo: Broadcast de eventos pode ter um papel de um *announcer* e um número arbitrário de papeis de *receiver*.



ACME
Exemplo

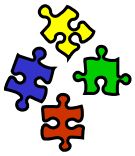



```
System simple_cs = {
    Component client = {Port send-request }
    Component server = {Port receive-request }
    Connector rpc = { Roles {caller, callee} }
    Attachments: {
            client.send-request to rpc.caller ;
            server.receive-request to rpc.callee }
}
```



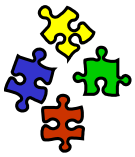
ACME
Representação

- Usado para descrição da representação hierárquica de arquiteturas
- O uso de múltiplas representações permite expressar múltiplas visões das entidades arquiteturais



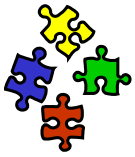
ACME
Rep-Map

- Usado para definir a correspondência entre a representação interna do sistema e a interface externa do componente ou conector
- Descreve uma associação entre portas externas e portas internas (para conectores define uma associação entre papéis internos e papéis externos)



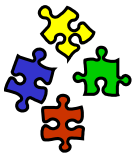
ACME
Propriedades

- Para acomodar a ampla variedade de informação auxiliar, ACME suporta anotação de estruturas arquiteturais com **listas de propriedades**
- Cada propriedade tem: nome, tipo opcional e valor
- Qualquer um dos sete tipos de entidades arquiteturais ACME podem ser anotadas

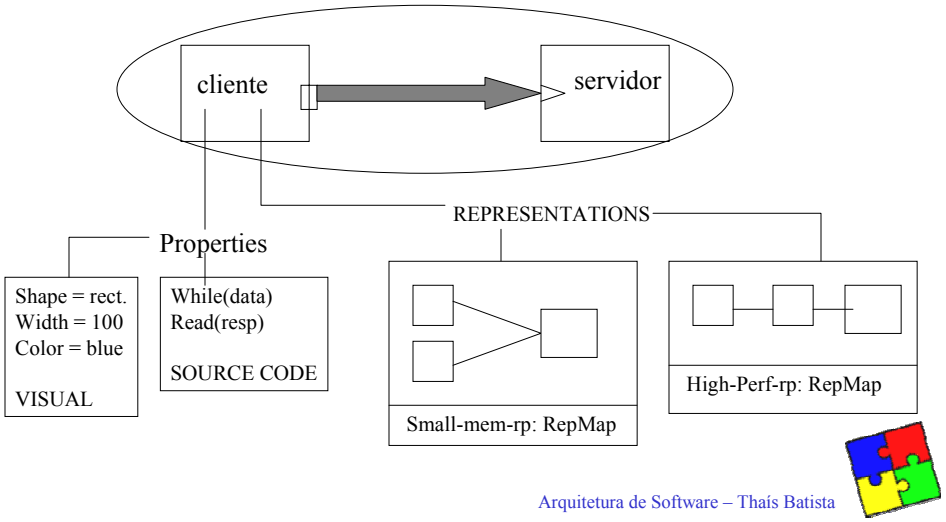


ACME
Exemplo com Propriedades

```
System simple_cs = {
    Component client = {
        Port send-request;
        Properties { Aesop-style: style-id = client-server;
                     Unicon-style: style-id = cs;
                     source-code: external = “CODE-LIB/client.c” }}
    Component server = {
        Port receive-request
        Properties {  idempotence: boolean = true;
                      max_concurrent_clients : integer = 1;
                      source-code: external = “CODE-LIB/server.c” }}
    Connector rpc = { Roles {caller, callee} }
        Properties {  synchronous : boolean = true;
                      max_roles : integer = 2;
                      protocol =  Wright = “...” }}
    Attachments: {
         client.send-request to rpc.caller ;
         server.receive-request to rpc.callee } }
```



Exemplo


cliente
servidor
REPRESENTATIONS
Properties
Shape = rect.
Width = 100
Color = blue
VISUAL
While(data)
Read(resp)
SOURCE CODE
Small-mem-rp: RepMap
High-Perf-rp: RepMap