

# *tecnologias web e gestão de identidade*

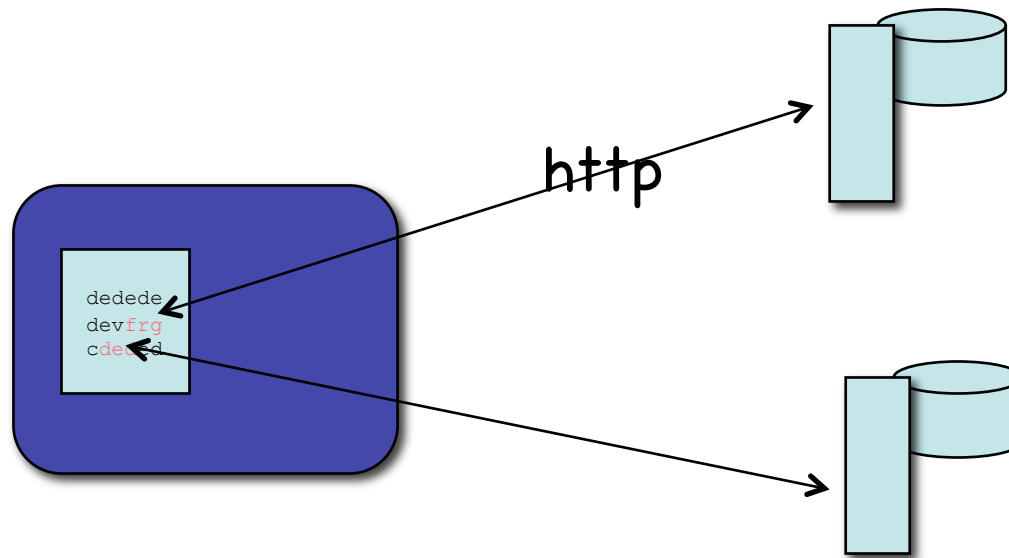


# *histórico*

- serviços da era inicial da Internet
  - telnet
  - ftp
  - gopher - estruturas hierárquicas
  - www - hipertexto
    - estudos sobre hipertexto e hipermídia
    - sistemas locais



# www - modelo básico

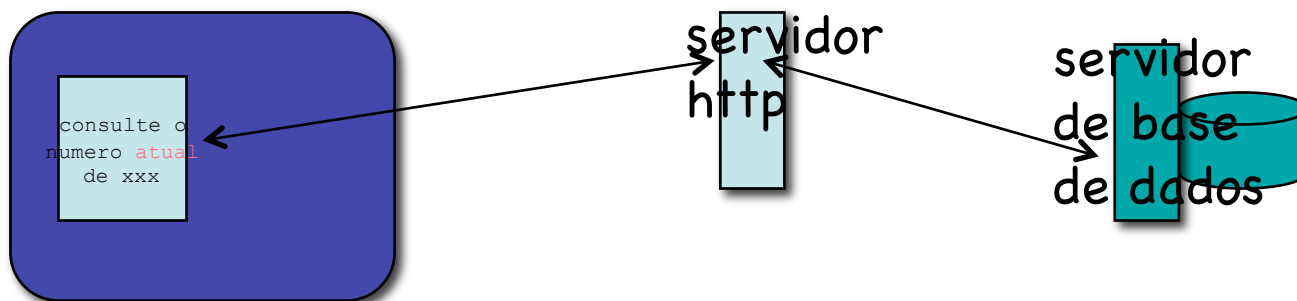


- usuário ↔ aplicação
- leitura de texto estático
  - browsers
  - http e html



# geração de páginas dinâmicas

- arquiteturas de três camadas



- tecnologias CGI e outras
  - problemas com ausência de estado



# *servidores proxy*

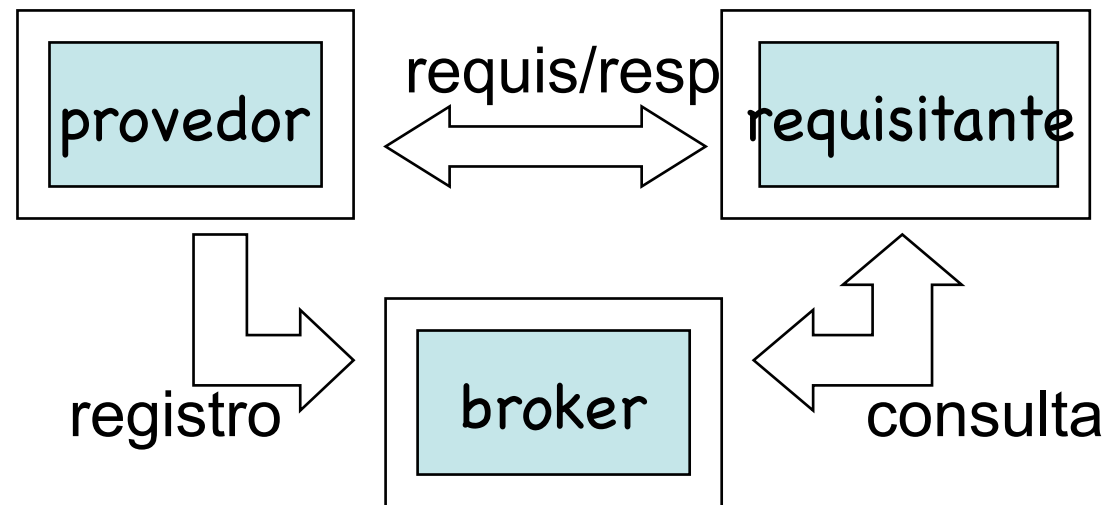


- segurança e caching



# Web services

- aplicação ↔ aplicação
- integração e interoperabilidade...



→ arquitetura básica semelhante a outros sistemas de chamadas remotas

- ★ *“Web services are self-contained, modular applications that can be described, published, located, and invoked over a network.”*

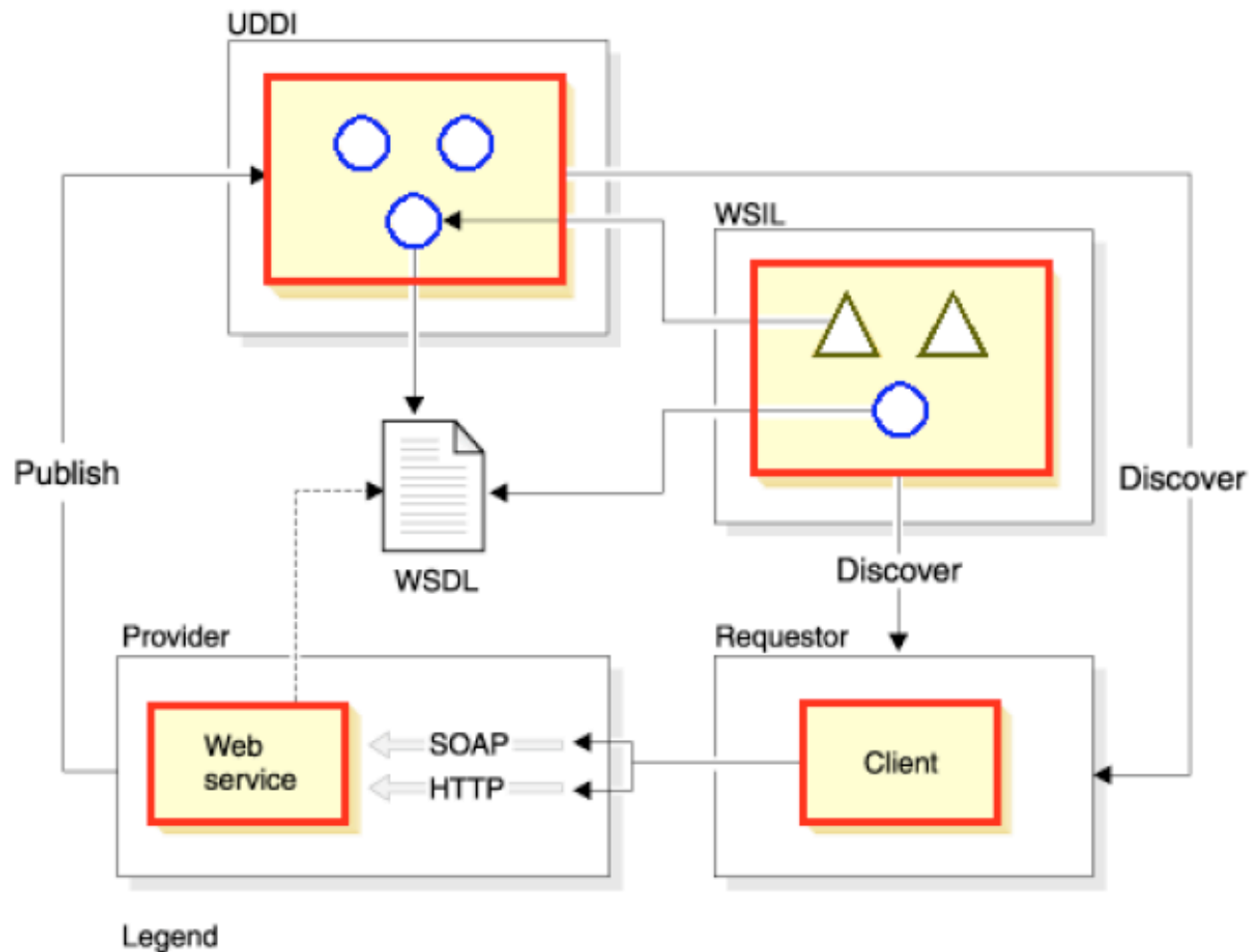


## *padronização - núcleo*

- **XML** (Extensible Markup Language) - linguagem de marcação genérica.
- **SOAP** (Simple Object Access Protocol) - protocolo para chamada de serviços, em XML.
- **WSDL** (Web Services Description Language) - linguagem (em XML) de descrição de interfaces e implementações.
- **UDDI** (Universal Description, Discovery, and Integration) - API para cliente e implementação de servidor SOAP que armazena e recupera informações sobre serviços e servidores



# *web services - arquitetura*



## *muitos outros padrões...*

- transações
- segurança
- mensagens
- eventos
- confiabilidade
- ...



# SOAP

- envelope de endereçamento, regras de codificação de dados, convenções de descrição de chamada e retorno
- em princípio independente de plataforma
  - na prática usado sobre http
- troca de informações serializadas
  - passagem sempre por cópia



## *exemplo: pedido de serviço*

```
POST /webapp/servlet/rpcrouter HTTP/1.1
Host: www.messages.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: ""
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd=http://www.w3.org/2001/XMLSchema
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns1:getMessage xmlns:ns1="urn:NextMessage"
      soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <UserID xsi:type="xsd:string">JDoe</UserID>
      <Password xsi:type="xsd:string">0JDOE0</Password>
    </ns1:getMessage>
  </soapenv:Body>
</soapenv:Envelope>
```



chamada a **getMessage (UserID, Password)**



## *exemplo: resposta*

HTTP/1.1 200 OK

Content-Type: text/xml; charset="utf-8"

Content-Length: nnnn

```
<soapenv:Envelope
  xmlns:soapenv="..." xmlns:xsd="..." xmlns:xsi="...">
  <soapenv:Body>
    <ns1:getResponseMessage xmlns:ns1="urn:NextMessage"
      soapenv:encodingStyle=
        "http://schemas.xmlsoap.org/soap/encoding/">
      <return xsi:type="xsd:string">Call mom!</return>
    </ns1:getResponseMessage>
  </soapenv:Body>
</soapenv:Envelope>
```



## *exemplo: chamada de Java*

```
public class MySoapClient {
    public static void main(String[] args) {
        Call call = new Call();
        call.setEncodingStyleURI(Constants.NS_URI_SOAP_ENC);
        call.setTargetObjectURI ("urn:NextMessage");
        call.setMethodName ("getMessage");
        Vector params = new Vector();
        Parameter userIDParam = new Parameter(
            "UserID", String.class, "JDoe", Constants.NS_SOAP_ENC);
        params.addElement(userIDParam);
        Parameter passwordParam = new Parameter(
            "Password", String.class, "0JDOE0", Constants.NS_SOAP_ENC);
        params.addElement(passwordParam);
        call.setParams(params);
        Response resp = null;
        URL url = new URL ("http://www.messages.com/soap/servlet/rpcrouter");
        resp = call.invoke (url, "urn:NextMessage"); // url, soapActionURI
```



## *nomes*

- uso de URIs (uniform resource identifiers)
  - inicialmente distinção entre URLs (uniform resource locators) e URNs (uniform resource names)

<scheme name> : <hierarchical part> [ ? <query> ] [ # <fragment> ]



## *segurança*

- uso de SSL ou TLS para criar canais criptografados
  - normalmente autenticação só de servidor
- pontos intermediários conhecem parceiros de comunicação
  - criptografia só para a camada de aplicação
- problemas como negação de serviço e estouro de pilha



## *controle de acesso*

- diversos domínios administrativos
- problema de autenticação e controle de acesso distribuídos



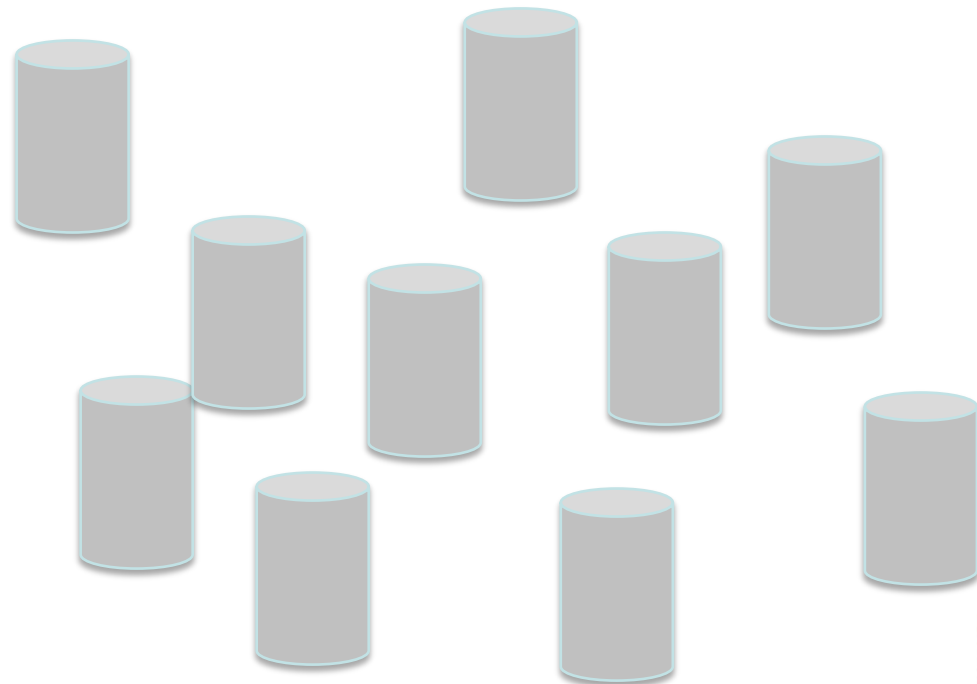
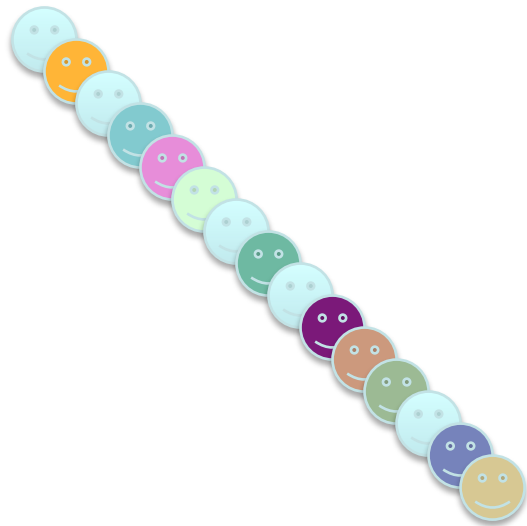
## *autenticação em diferentes domínios*

- cadastro individual de cada usuário em cada serviço
  - ônus para administrador de serviço
    - cadastro de cada usuário e de seus direitos
  - ônus para usuário:
    - senha (ou outra coisa) para cada serviço?
- conta única para todos os usuários de certa instituição
  - ônus para administrador de serviço:
    - não há como fazer auditoria
  - ônus para usuário
    - não há como diferenciar direitos



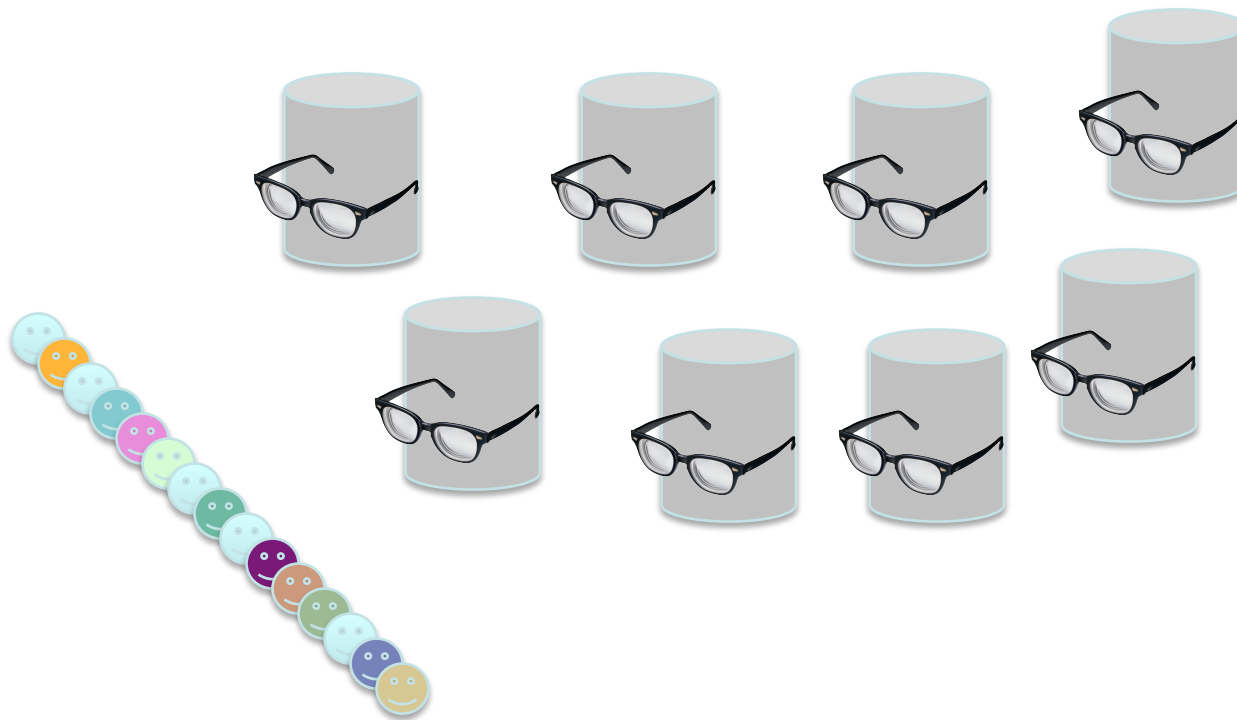
# *autenticação e controle de acesso*

- mapeamento de identidade a direitos?
- dificuldades de gerenciamento
  - escala
  - manutenção



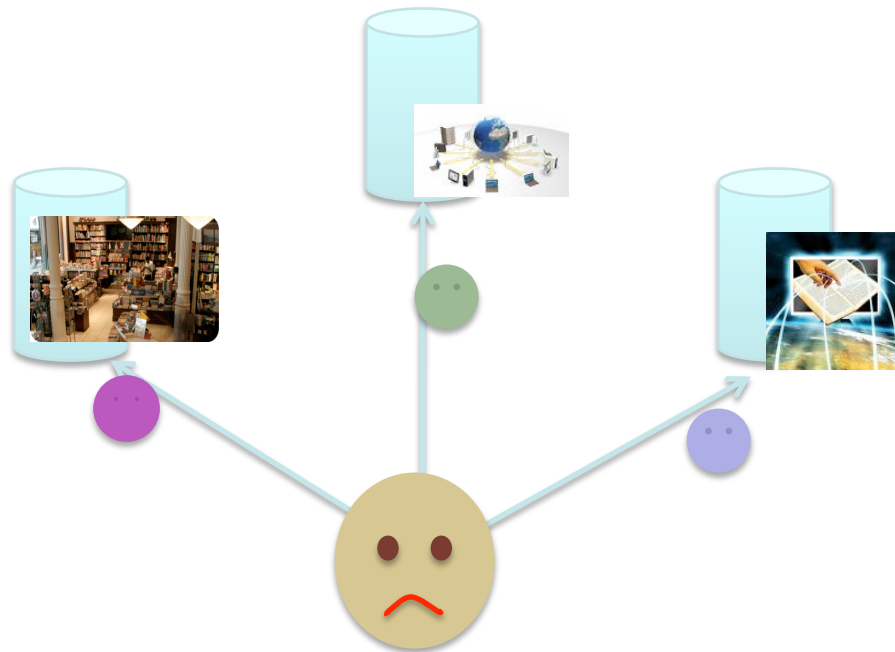
# *autenticação e controle de acesso*

- privacidade
  - cada serviço precisa saber quem é o usuário????

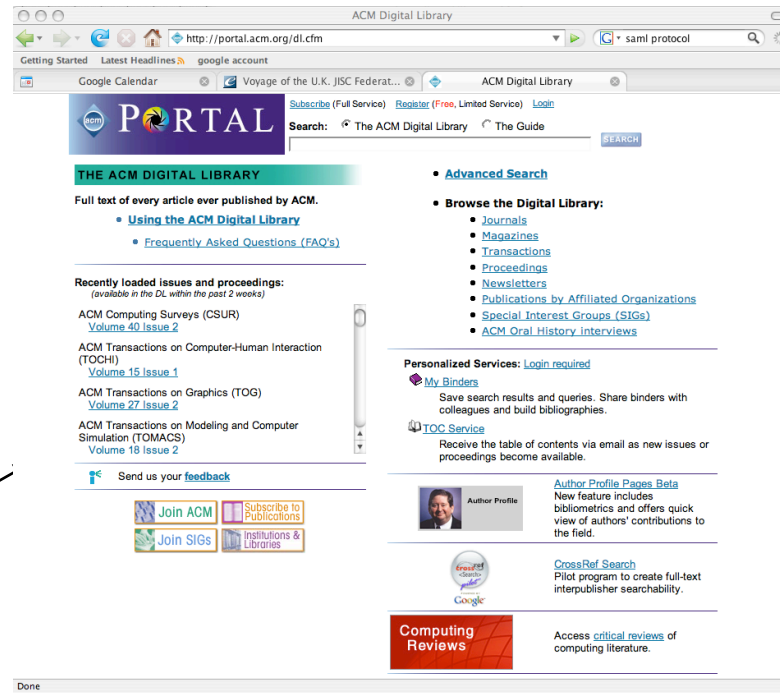


# *autenticação e controle de acesso*

- comodidade de uso
  - apresentação de credenciais a cada servidor???



# exemplos



acesso a editoras online

- reconhecimento de usuários de intuições cadastradas



# exemplos



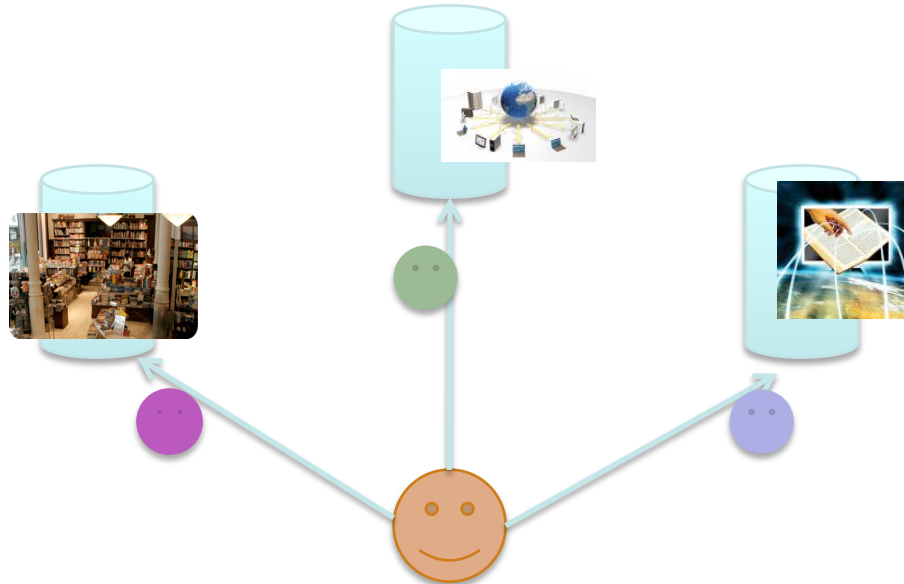
vendas com descontos p/  
estudantes

- como saber que usuário é estudante?
- certificado? mas serviço tem que conhecer cada usuário? e certificado tem que armazenar o que??



# *controle de acesso baseado em papéis*

- usuários e papéis
  - estudante
  - sócio da SBC
  - participante de projeto
  - ...



# *sistemas de gestão de identidade*

- autenticação de usuário por provedor de identidade
- provedor de serviço recebe uma garantia de autenticação
  - rede de confiança entre provedores de serviço e provedores de identidade
- provedor de serviço pode requisitar atributos do usuário para controle de acesso
  - relação com papéis
  - preocupação com privacidade
- soluções específicas para web



# uso de provedores de identidade

- provedores de serviços confiam em algumas fontes de identidade



provedor de identidade



serviço em qualquer lugar

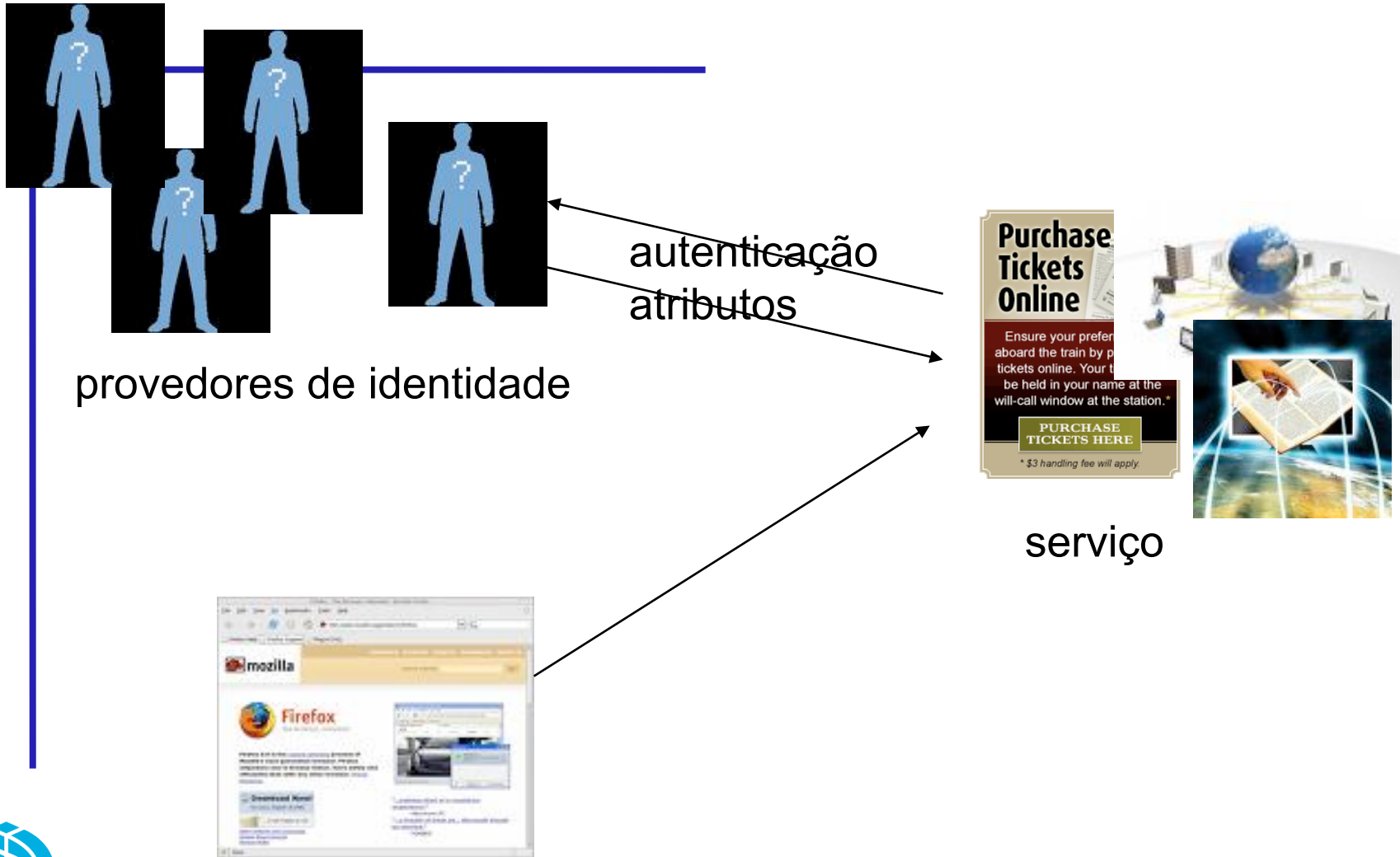


# *Federações acadêmicas*

- redes de confiança:
  - universidades e outras instituições de ensino e pesquisa
    - atuação como provedores de identidade e como provedores de serviço
  - editoras e outras
    - atuação como provedor de serviço



# Federações em comunidades acadêmicas

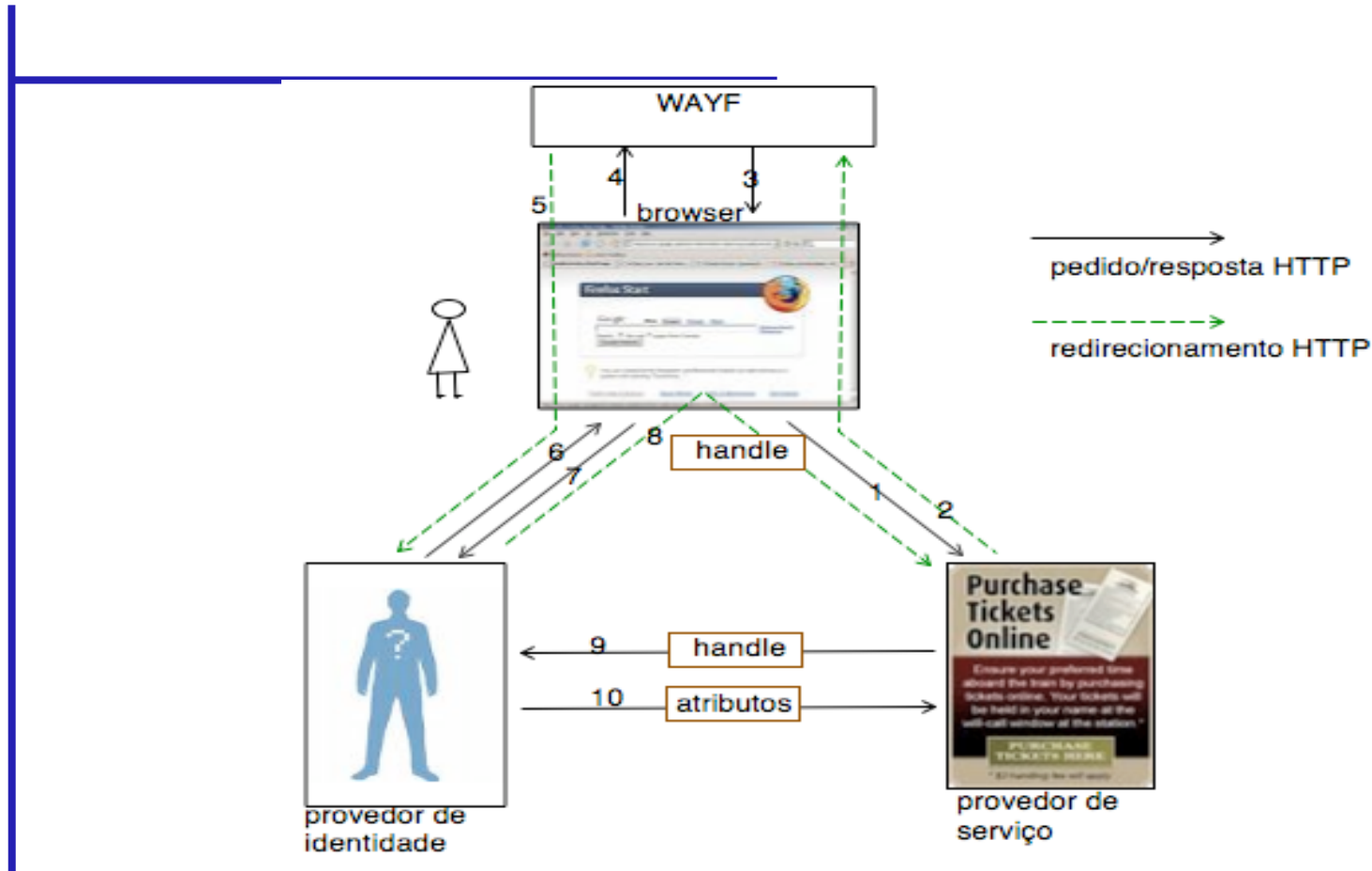


# *Sistemas de gestão de identidade - evolução*

- soluções iniciais: protocolos próprios para comunicação entre provedores de serviço e provedores de identidade
- evolução para padronização: SAML
  - security assertion markup language
  - definição de vários perfis de comunicação
  - diversas implementações baseadas ou compatíveis com SAML
    - shibboleth (Internet2)
    - simplesaml (Uninnet & wayf.dk)
    - Liberty Alliance
    - ...



## exemplo: uso de shibboleth



- protocolo SAML usado na comunicação (shib 2.x)



# Privacidade

- provedor de serviço deve receber o mínimo de informação necessário para estabelecer direitos de acesso
  - “estudante da PUC-Rio”
  - “sócio da SBC”
  - ...
- acordos entre provedores de serviço e provedores de identidade para liberação de atributos
- controle do usuário



## *Federação CAFe (RNP)*

- Instituições originárias do projeto piloto
  - CEFET-MG, UFC, UFF, UFMG, UFRGS
- Instituições que aderiram ao Serviço Experimental em 2008.2
  - UFPA, UFPE, UFV, UFMS, UNIVASF, USP
- Metas para 2009:
  - 6 novas instituições serão convidadas
    - Incluindo a RNP e algumas UPs
  - 2 provedores de serviço



