

OpenACS e as Comunidades Virtuais

Eduardo Santos

eduardo.edusantos@gmail.com
eduardo.santos@planejamento.gov.br

www.softwarepublico.gov.br

O que é o OpenACS

- Uma comunidade de desenvolvedores
- Um livro sobre aplicações para a Internet
- Uma tecnologia para desenvolvimento de aplicações Web
- Um repositório de software livre e código aberto bastante antigo
- Um consórcio de instituições e empresas

Comunidade OpenACS

- Conteúdo editado por experts ou voluntários
- Presença constante de meios de colaboração
- Moderação e cultura da comunidade
- Baseada nos ideais de software livre desde o começo

Histórico

- 1996: ACS (ArsDigita Community System)
- 1998: surgimento do “toolkit” ACS
- 1999: Projeto ACS/pg no Source Forge
- 2000: ACS é reescrito em Java
- 2001: ArsDigita prioriza ACS4 (Java) e considera ACS3 obsoleto

Histórico

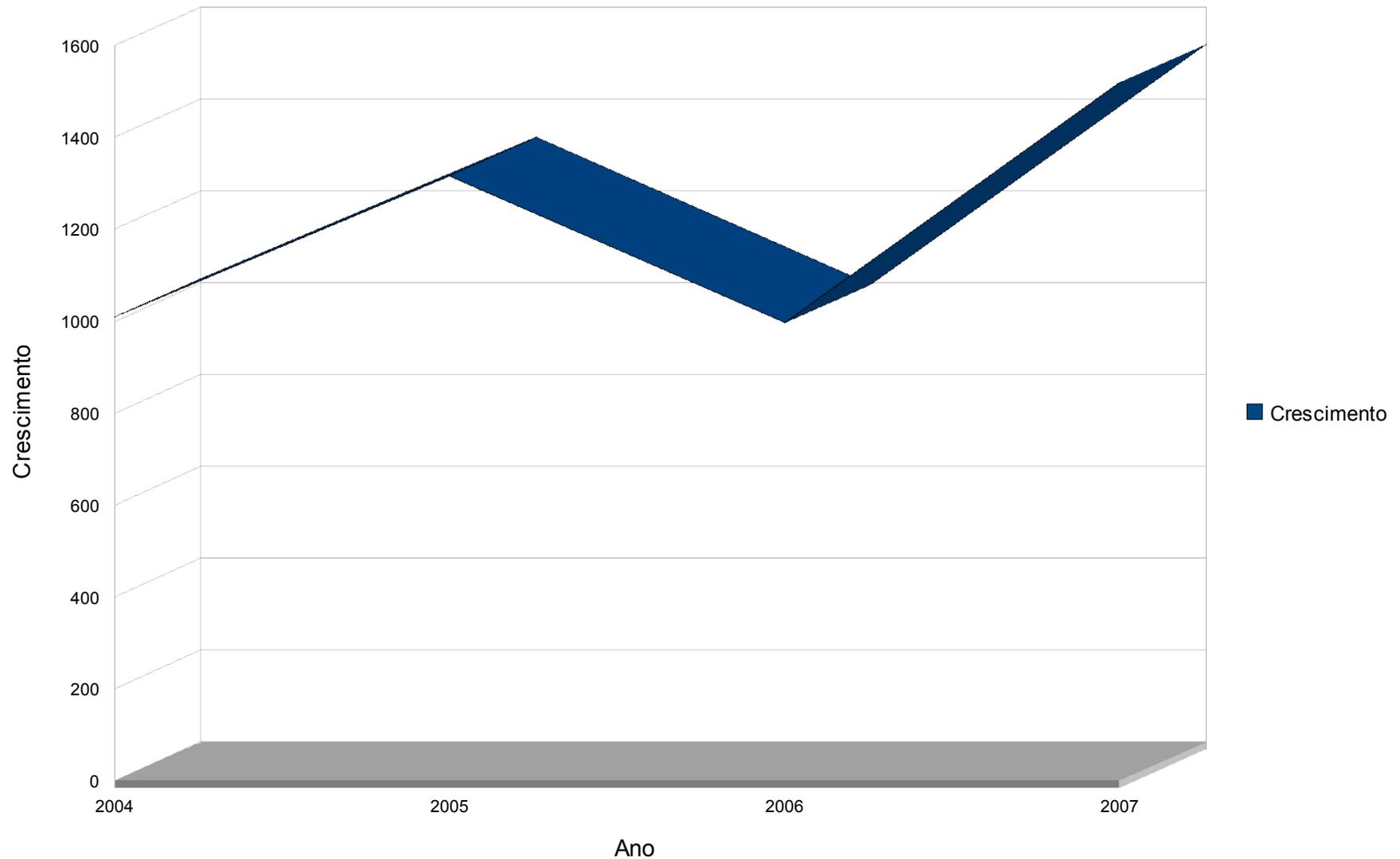
- 2001: Fork do ACS3, ainda em TCL (agora OpenACS4) supera ACS4 em desenvolvimento
- ainda 2001: Saída de Greenspun, tentativa de fechar o código, recusa de parceira com Microsoft
- final 2001: ACS4.6 (Java) tem sua licença modificada e deixa de ser livre, excluindo a comunidade OpenACS

Histórico

- Fevereiro de 2002: ArsDigita é comprada pela Red Hat e fecha as portas
- 28 de outubro de 2002: OpenACS 4.6.0 é liberado

Crescimento da comunidade

Crescimento da comunidade Internacional



PANDA

philip.greenspun.com/panda

- Philip and Alex's Guide to Web Publishing
- Publicado pela primeira vez em setembro de 1998
- Interessante histórico sobre o começo da Internet

SEIA

philip.greenspun.com/seia

- Software Engineering for Internet Applications
- Publicado em 2006
- Ótima guia sobre como construir aplicações para a Internet
- Serve para qualquer ambiente de desenvolvimento

Tecnologia

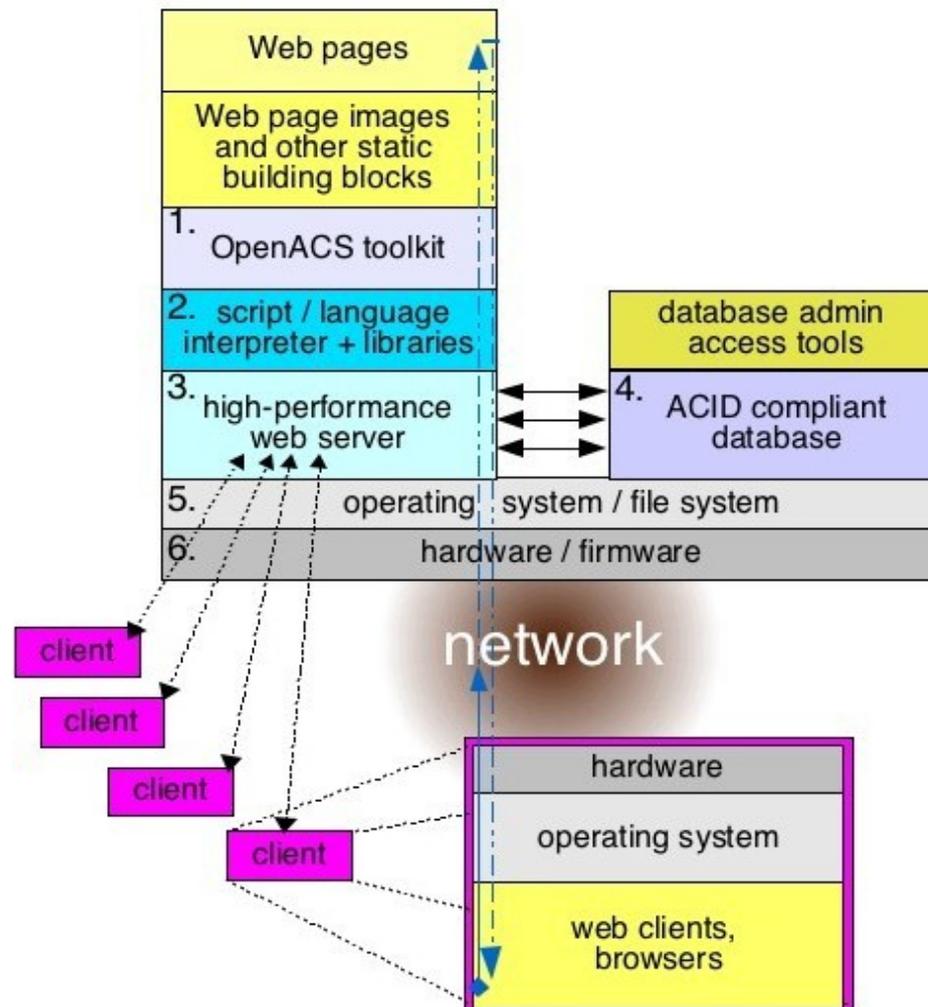
“No desenvolvimento do software, um framework ou arcabouço é uma estrutura de suporte definida em que um outro projeto de software pode ser organizado e desenvolvido.”

Wikipedia

Tecnologia

- O OpenACS é um framework de desenvolvimento Web, em muitas características
- É também um “toolkit” para desenvolvimento de aplicações de comunidades
- É um ambiente agregador nos princípios de modularidade e empacotamento

Tecnologia



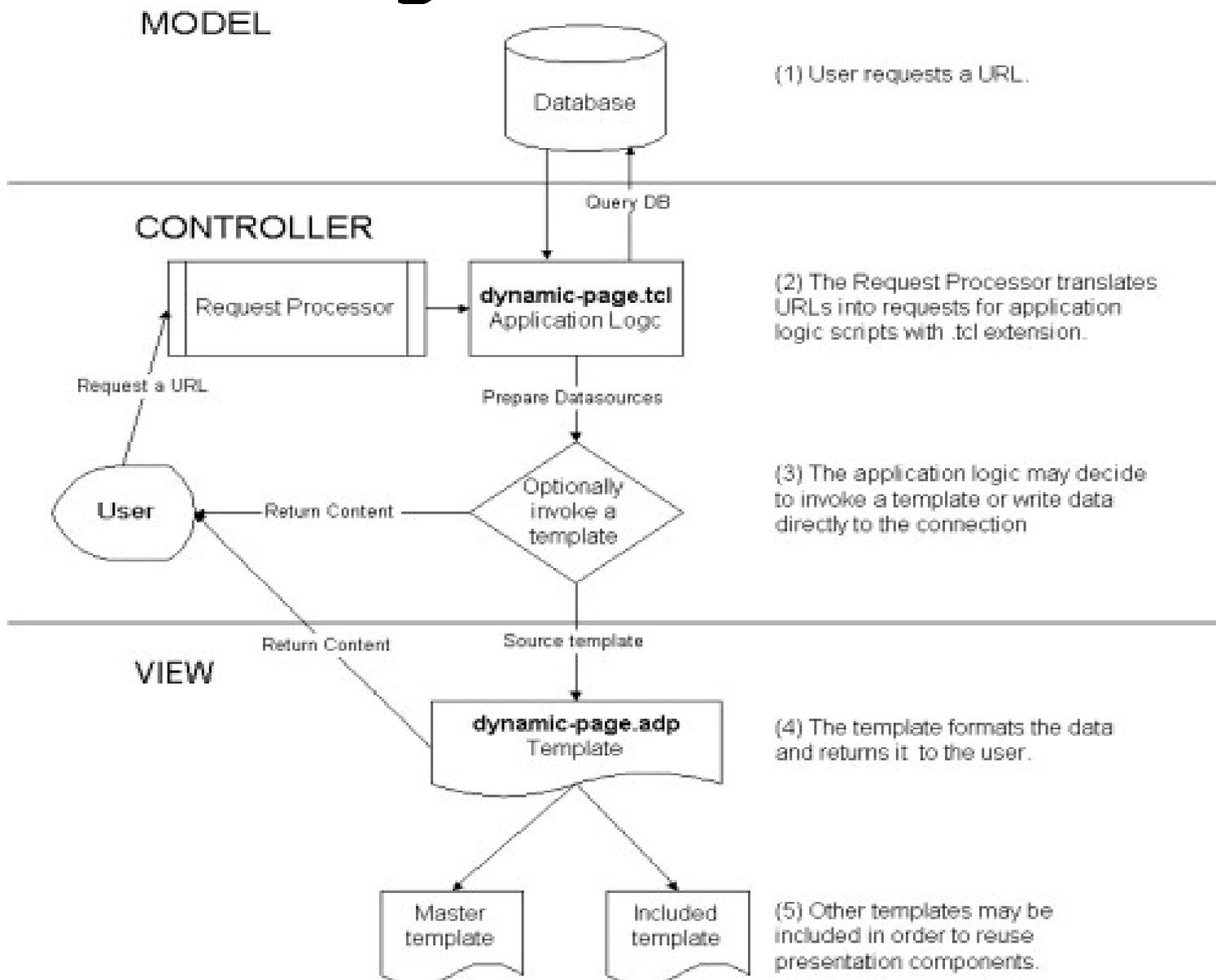
Arquitetura

- Arquitetura Web
- Modelo MVC
- Metadados e geração automática de código
- Gerenciamento de papéis e perfis
- Arquitetura multi-pool do banco de dados

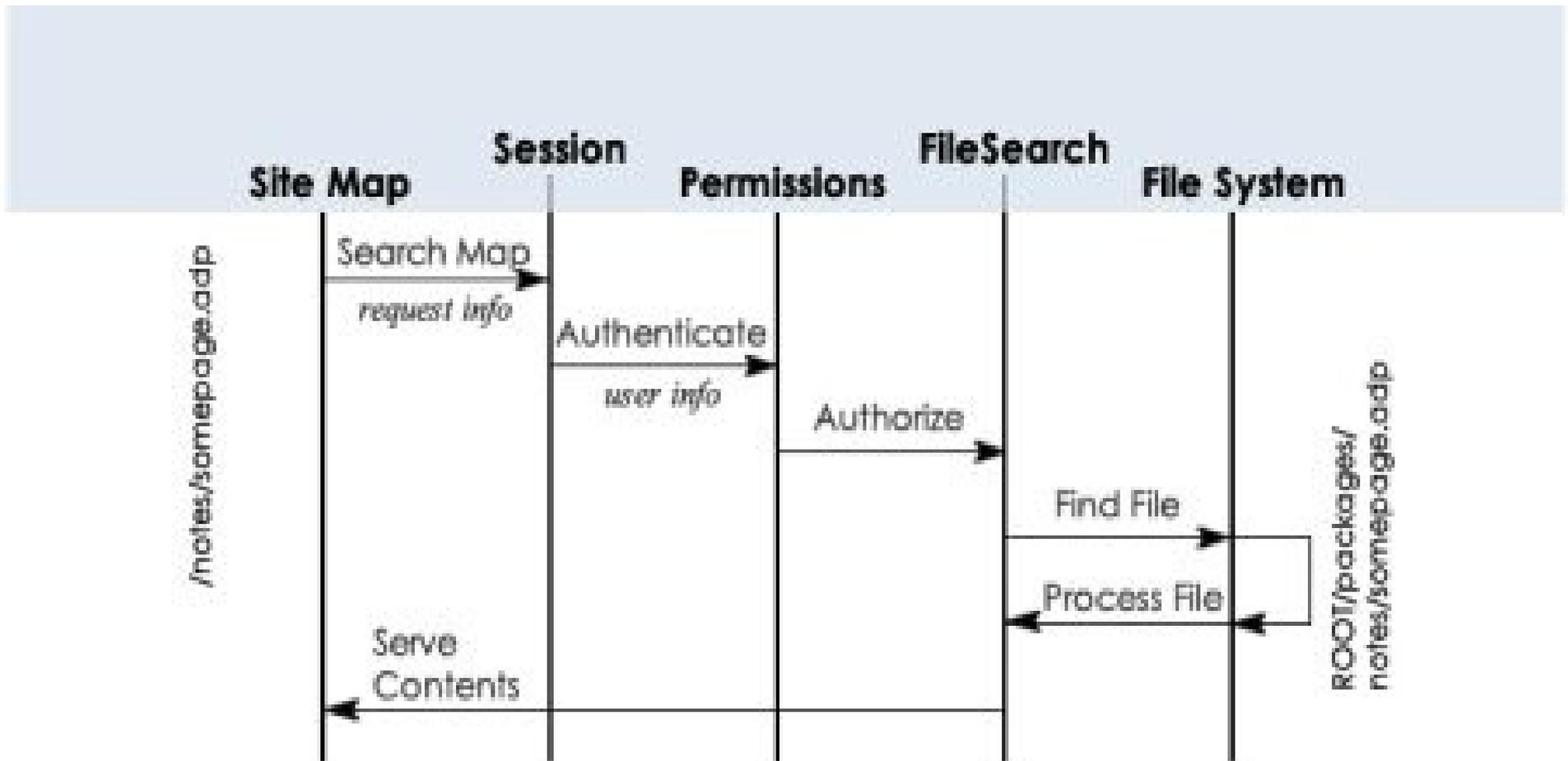
Modelo MVC

- Model-View-Controller
 - Separação entre código e design
 - Geração automática de código
 - Camada de banco de dados separada
 - sistema de cache de queries
- Request Processor
 - Tratamento das requisições de página feitas pelo usuário

Diagrama MVC



Request Processor



Request Processor

- Possibilidade de criar um mapa virtual do site
- Tratamento de autenticação por usuário e por sessão
- Sistema de permissões por arquivo ou URL
- Tratamento da lógica e conteúdo de maneira eficiente

Modelagem de dados

- Sistema baseado em objetos
 - Tipos de objetos = ~ Classes
 - Atributos de dados armazenados em mais de uma tabela
 - Tipos de objetos podem ter dados “herdados” do pai
 - Modelado a objeto X Orientado a objeto (BD X Programa)

Metadados

- Dados que geram dados (geração automática de código)
 - Maior facilidade de integrar “atores” no sistema
 - Relacionar vários “atores” sem necessidade de criar novas tabelas específicas

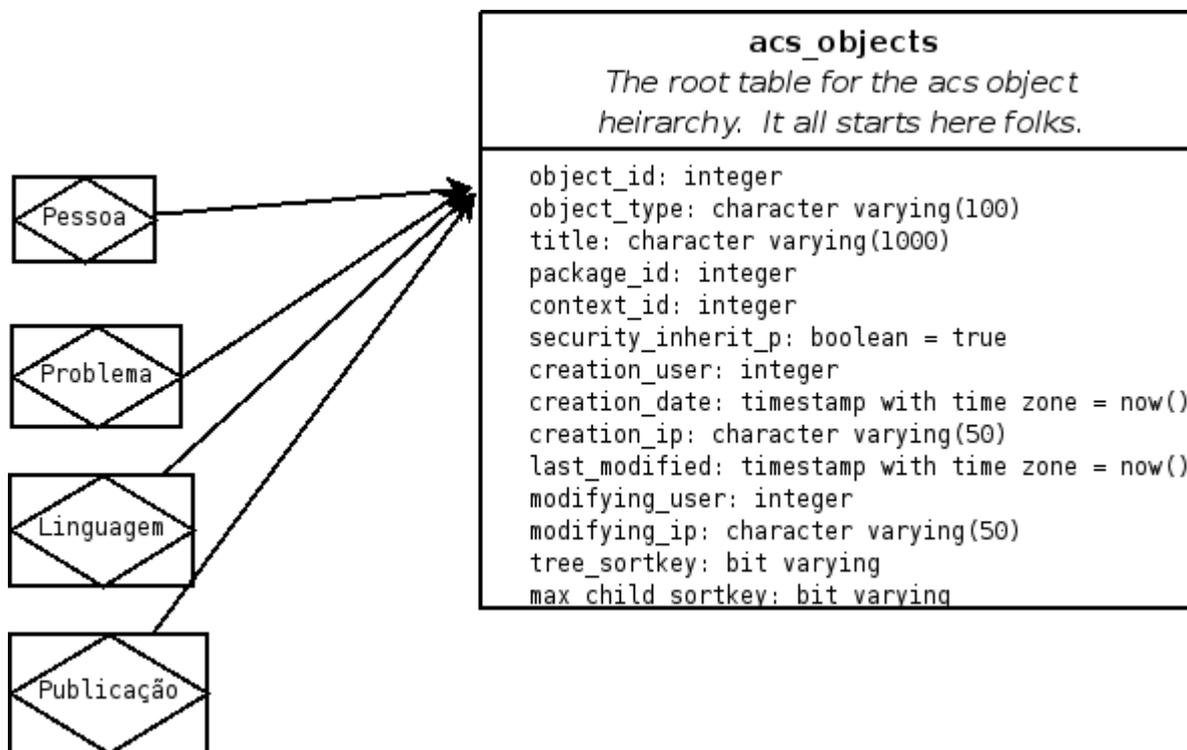
Metadados

- Vamos assumir os seguintes tipos como exemplo:



Metadados

- Definimos todos os tipos como objetos numa tabela

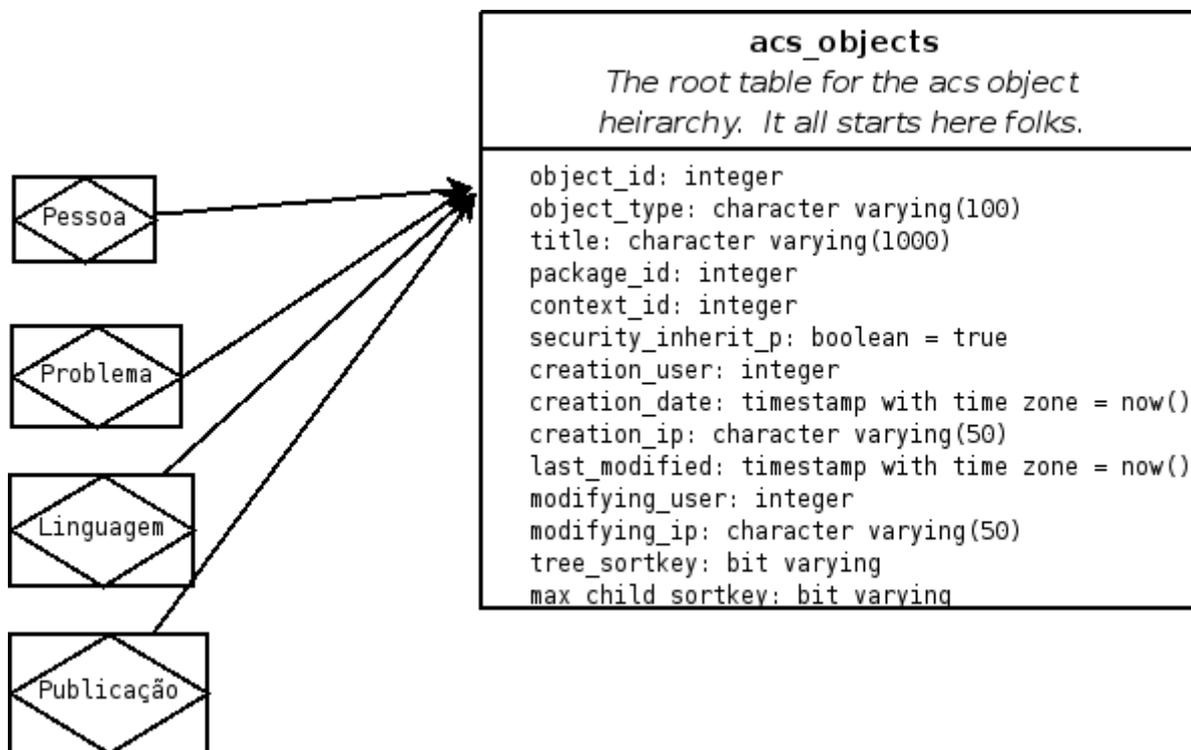


Metadados

- Exemplo: João criou a linguagem LISP
 - Objeto: Linguagem LISP
 - Criador: João
- Pergunta: por que não referenciar a tabela de usuários?
- Resposta: O criador pode não fazer mais parte do sistema

Metadados

- Para cada objeto, precisamos de várias informações

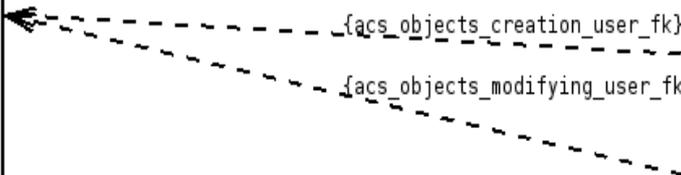


Metadados

- Além disso, precisamos de informações específicas do objeto

| users |
|---|
| <i>The creation_date and creation_ip columns inherited from acs_objects indicate when and from where the user registered. How do we apply a constraint ("email must not be null") to the parent type?</i> |
| <code>user_id: integer authority_id: integer username: character varying(100) screen_name: character varying(100) priv_name: integer priv_email: integer = 5 email_verified_p: boolean = true email_bouncing_p: boolean = false no_alerts_until: timestamp with time zone last_visit: timestamp with time zone second_to_last_visit: timestamp with time zone n_sessions: integer = 1 password: character(40) salt: character(40) password_question: character varying(1000) password_answer: character varying(1000) password_changed_date: timestamp with time zone auth token: character varying(100)</code> |

| acs_objects |
|--|
| <i>The root table for the acs object heirarchy. It all starts here folks.</i> |
| <code>object_id: integer object_type: character varying(100) title: character varying(1000) package_id: integer context_id: integer security_inherit_p: boolean = true creation_user: integer creation_date: timestamp with time zone = now() creation_ip: character varying(50) last_modified: timestamp with time zone = now() modifying_user: integer modifying_ip: character varying(50) tree_sortkey: bit varying max child sortkey: bit varying</code> |



Metadados

- A estrutura de metadados me permite gerar códigos genéricos para qualquer tipo de dado
- Através das API's do sistema, toda a estrutura de relacionamentos obedece um padrão

Metadados

- XQL query dispatcher: independência de banco de dados
- Service Contract API
 - Reusabilidade de código
 - Integração de aplicações
 - Extensibilidade de pacotes

Metadados

```
v_user_id := person__new(  
    v_user_id,  
    p_object_type,  
    p_creation_date,  
    p_creation_user,  
    p_creation_ip,  
    p_email,  
    p_url,  
    p_first_names,  
    p_last_name,  
    p_context_id  
);
```

```
select acs__add_user(  
    :user_id,  
    'user',  
    now(),  
    null,  
    :peeraddr,  
    :authority_id,  
    :username,  
    :email,  
    :url,  
    :first_names,  
    :last_name,  
    :hashed_password,  
    :salt,  
    :screen_name,  
    :email_verified_p,  
    :member_state  
);
```

Gerenciamento de papéis e perfis

- Relacionamento entre objetos
- Perfis entre grupos de objetos
- Gerenciamento de papéis

acs_rels

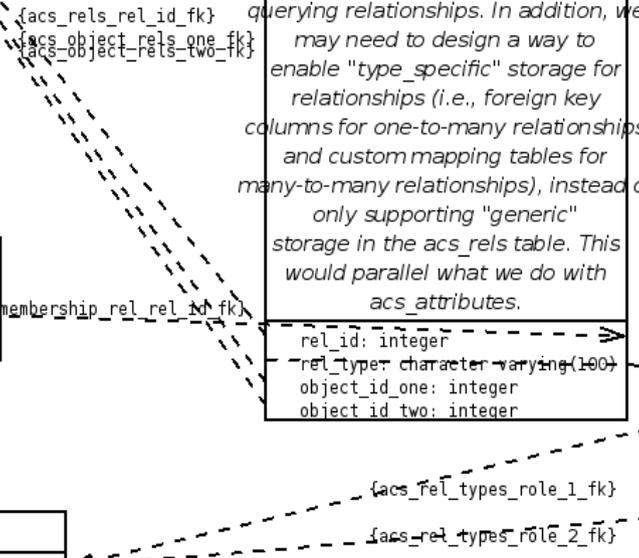
The `acs_rels` table is essentially a generic mapping table for `acs` objects. Once we come up with a way to associate attributes with relationship types, we could replace many of the ACS 3.x mapping tables like `user_content_map`, `user_group_map`, and `user_group_type_modules_map` with this one table. Much application logic consists of asking questions like "Does object X have a relationship of type Y to object Z?" where all that differs is X, Y, and Z. Thus, the value of consolidating many mapping tables into one is that we can provide a generic API for defining and querying relationships. In addition, we may need to design a way to enable "type-specific" storage for relationships (i.e., foreign key columns for one-to-many relationships and custom mapping tables for many-to-many relationships), instead of only supporting "generic" storage in the `acs_rels` table. This would parallel what we do with `acs_attributes`.

| acs_objects |
|--|
| The root table for the <code>acs</code> object heirarchy. It all starts here folks. |
| <pre> object_id: integer object_type: character varying(100) title: character varying(1000) package_id: integer context_id: integer security_inherit_p: boolean = true creation_user: integer creation_date: timestamp with time zone = now() creation_ip: character varying(50) last_modified: timestamp with time zone = now() modifying_user: integer modifying_ip: character varying(50) tree_sortkey: bit varying max child sortkey: bit varying </pre> |

| acs_rel_types |
|---|
| Each row in <code>acs_rel_types</code> represents a type of relationship between objects. For example, the following DML statement: |
| <pre> <blockquote><pre> insert into acs_rel_types (rel_type, object_type_one, role_one, min_n_rels_one, max_n_rels_one, object_type_two, role_two, min_n_rels_two, max_n_rels_two) values ('employment', 'person', 'employee', 0, null, 'company', 'employer', 0, null) </pre></blockquote> </pre> |
| defines an "employment" relationship type that can be expressed in natural language as: |
| <pre> <blockquote> A person may be the employee of zero or more companies, and a company may be the employer of zero or more people. </blockquote> </pre> |
| <pre> rel_type: character varying(100) object_type_one: character varying(100) role_one: character varying(100) min_n_rels_one: integer max_n_rels_one: integer object_type_two: character varying(100) role_two: character varying(100) min_n_rels_two: integer max_n_rels_two: integer acs_rel_types_max_n_1_ck(CHECK ((max_n_rels_one >= 0))) acs_rel_types_max_n_2_ck(CHECK ((max_n_rels_two >= 0))) acs_rel_types_min_n_1_ck(CHECK ((min_n_rels_one >= 0))) acs_rel_types_min_n_2_ck(CHECK ((min_n_rels_two >= 0))) acs_rel_types_n_rels_one_ck(CHECK ((min_n_rels_one <= max ... one))) acs_rel_types_n_rels_two_ck(CHECK ((min_n_rels_two <= max ... two))) </pre> |

| membership_rels |
|---|
| <pre> rel_id: integer member state: character varying(20) membership_rel_mem_ck(CHECK ((member_state)::text = ... []))) </pre> |

| acs_rel_roles |
|---|
| <pre> role: character varying(100) pretty_name: character varying(100) pretty_plural: character varying(100) </pre> |

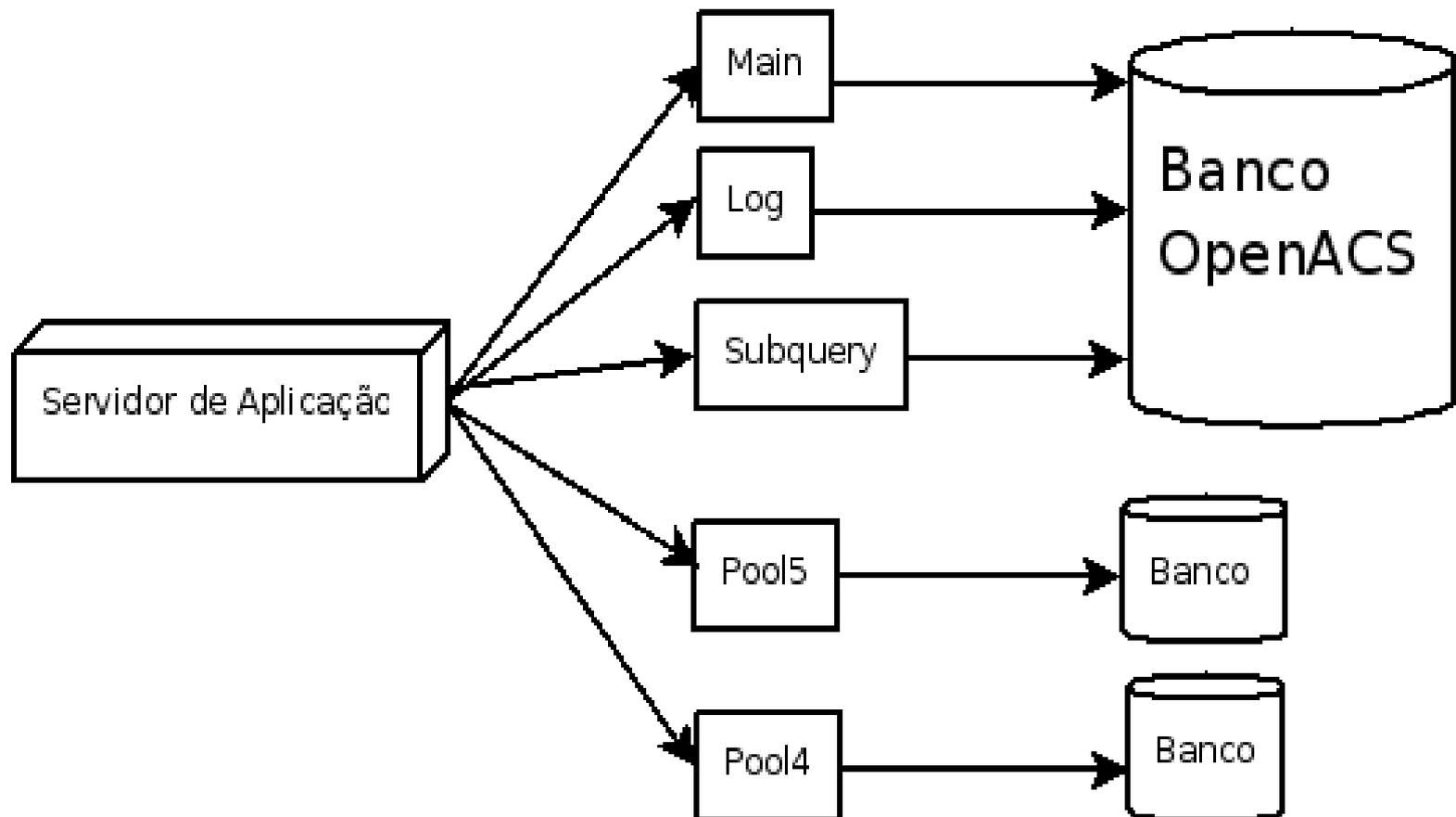
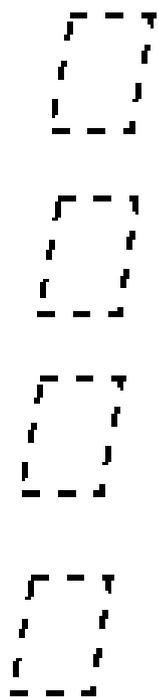


Arquitetura multi-pool

- Sempre que o sistema é iniciado, três pools de conexão são abertos:
 - Main: executa a maior parte das rotinas direto no banco
 - Subquery: permite executar uma query mesmo um um pool já aberto
 - Log: Normalmente grava os registros das transações sendo executadas no momento
 - etc...

Arquitetura multi-pool

Threads



Modularidade

- A estrutura modular permite a criação de pacotes
- Pacotes são aplicações construídas utilizando o núcleo do OpenACS

Estrutura de Arquivos

```
ROOT/  
  bin/  
    Various executables and scripts for server maintenance.  
  content-repository-content-files/  
    content repository content stored in the filesystem.  
  etc/  
    Installation scripts and configuration files.  
  packages/  
    acs-admin/  
    acs-api-browser/  
    ... many many more...  
    workflow/  
  log/  
    Server error and access logs  
  tcl/  
    bootstrap code  
  www/  
    Pages not in packages (static content, customized pages)
```

Estrutura de um pacote

```

ROOT/
+-- packages/      APM Root
+-- notes/         Package Root
|
|   +-- notes.info      Package Specification File
|   +-- sql/
|   |   +-- oracle/
|   |   |   +-- notes-create.sql      Data Model Creation Script for Oracle
|   |   |   +-- notes-drop.sql       Data Model Drop Script
|   |   |   +-- *.sql                 Data Model Files
|   |   |   +-- upgrade/
|   |   |   |   +-- upgrade-4.1-4.5.sql Data Model Upgrade Scripts
|   |   +-- postgresql/
|   |   |   +-- notes-create.sql      Data Model Creation Script for PostgreSQL
|   |   |   +-- notes-drop.sql       Data Model Drop Script
|   |   |   +-- *.sql                 Data Model Files
|   |   |   +-- upgrade/
|   |   |   |   +-- upgrade-4.1-4.5.sql Data Model Upgrade Scripts
|   +-- tcl/
|   |   +-- *.tcl                    Tcl Library Files
|   +-- lib/
|   |   +-- *.tcl                    Includable page logic
|   |   +-- *.adp                    Includable page templates
|   +-- www/
|   |   +-- admin/
|   |   |   +-- tests/
|   |   |   |   +-- index.tcl         Regression Tests
|   |   |   |   +-- ...              Regression Test Index Page
|   |   |   |   +-- ...              Regression Tests
|   |   |   +-- index.tcl           Administration UI Index Page
|   |   |   +-- ...                 Administration UI Pages
|   |   +-- doc/
|   |   |   +-- index.html           Documentation
|   |   |   +-- ...                 Documentation Index Page
|   |   +-- resources/
|   |   |   +-- ...                 Administration Pages
|   |   |   +-- ...                 Static Content
|   |   |   +-- ...                 Static Content files
|   |   +-- *.tcl                   UI Logic Scripts
|   |   +-- *.adp                   UI Templates
|   |   +-- *-oracle.xml             Oracle-specific Queries
|   |   +-- *-postgresql.xml        PostgreSQL-specific Queries
|
+-- Other package directories.

```

Pacotes

- Uma aplicação completa pode conter mais de um pacote
- É possível fornecer um arquivo XML que sirva de “instalador”

```
<?xml version="1.0"?>
```

```
<!-- This must be moved to the OpenACS page root when making a release tarball -->
```

```
<application name="dotlrn" pretty-name=".LRN" home="http://openacs.org/projects/dotlrn" min_stack_size="512">
```

```
<actions>
```

```
<!-- Installing all applets gives us what we need as applets depend on both dotlrn, the portlets
```

```
and the respective applications (i.e. forums file-storage etc.) -->
```

```
<install package="dotlrn-bm"/>
```

```
<install package="dotlrn-calendar"/>
```

```
<install package="dotlrn-dotlrn"/>
```

```
<install package="dotlrn-faq"/>
```

```
<install package="dotlrn-forums"/>
```

```
<install package="dotlrn-fs"/>
```

```
<install package="dotlrn-homework"/>
```

```
<install package="dotlrn-news"/>
```

```
<install package="dotlrn-portlet"/>
```

```
<install package="dotlrn-static"/>
```

```
<install package="dotlrn-assessment"/>
```

```
<install package="dotlrn-evaluation"/>
```

```
<install package="theme-selva"/>
```

```
<mount package="dotlrn" mount-point="/dotlrn" instance-name="dotLRN"/>
<set-parameter url="/" name="DefaultMaster"
value="/packages/dotlrn/www/dotlrn-master"/>
<set-parameter package-key="acs-kernel" name="IndexRedirectUrl" value="/dotlrn/
index"/>
<set-parameter package-key="acs-kernel" name="CommunityMemberURL"
value="/dotlrn/community-member"/>
<set-parameter package-key="acs-kernel" name="CommunityMemberAdminURL"
value="/dotlrn/admin/user"/>
<set-parameter package-key="new-portal" name="default_theme_name"
value="#new-portal.sloan_theme_name#"/>
<set-parameter package-key="acs-bootstrap-installer"
name="post_installation_message" value="When the server restarts click <a
href='/dotlrn'>here</a> to configure .LRN"/>
</actions>

</application>
```

Aplicações Verticais

-]Project Open[
 - Módulo ERP (*Enterprise Resource Planning*) do OpenACS;
 - Objetivos: administração de custos e colaboração entre os membros da equipe;
 - Wiki e chat integrados (estilo OpenACS);
 - Módulo de workflow específico;
 - Gerencia “papéis” dentro de uma empresa.

Aplicações Verticais

- Bug-tracker, Ticket-tracker e Workflow
 - Ferramenta para gerência de projetos de software;
 - Exemplo de workflow simplificado;
 - Possibilidade de gerência da agenda de trabalho;
 - Gerência de tickets;
 - Possibilidades de implementação

Aplicações Verticais

- dotLRN (.LRN)
 - Ambiente de Aprendizagem Virtual (AVA)
 - Ambiente virtual para professores e alunos compartilhar tecnologias da informação e aliá-las ao processo de ensino-aprendizagem;
 - *Learning Management System* (LMS)
 - Software que automatiza a administração dos eventos de treinamento;
 - Desenvolvido para lidar com cursos de múltipla publicação.

Aplicações Verticais

- Gerido por um consórcio internacional, responsável por:
 - Fazer o controle de qualidade;
 - Realizar pesquisas educacionais;
 - Promover o uso do software no mundo;
 - Gerenciar a comunidade de desenvolvedores.
- Câmara de diretores:
 - <http://www.dotlrn.org/about/board/>
- Equipe de liderança:
 - <http://www.dotlrn.org/about/leadership/>

Aplicações Verticais

- www.catir.sede.embrapa.br

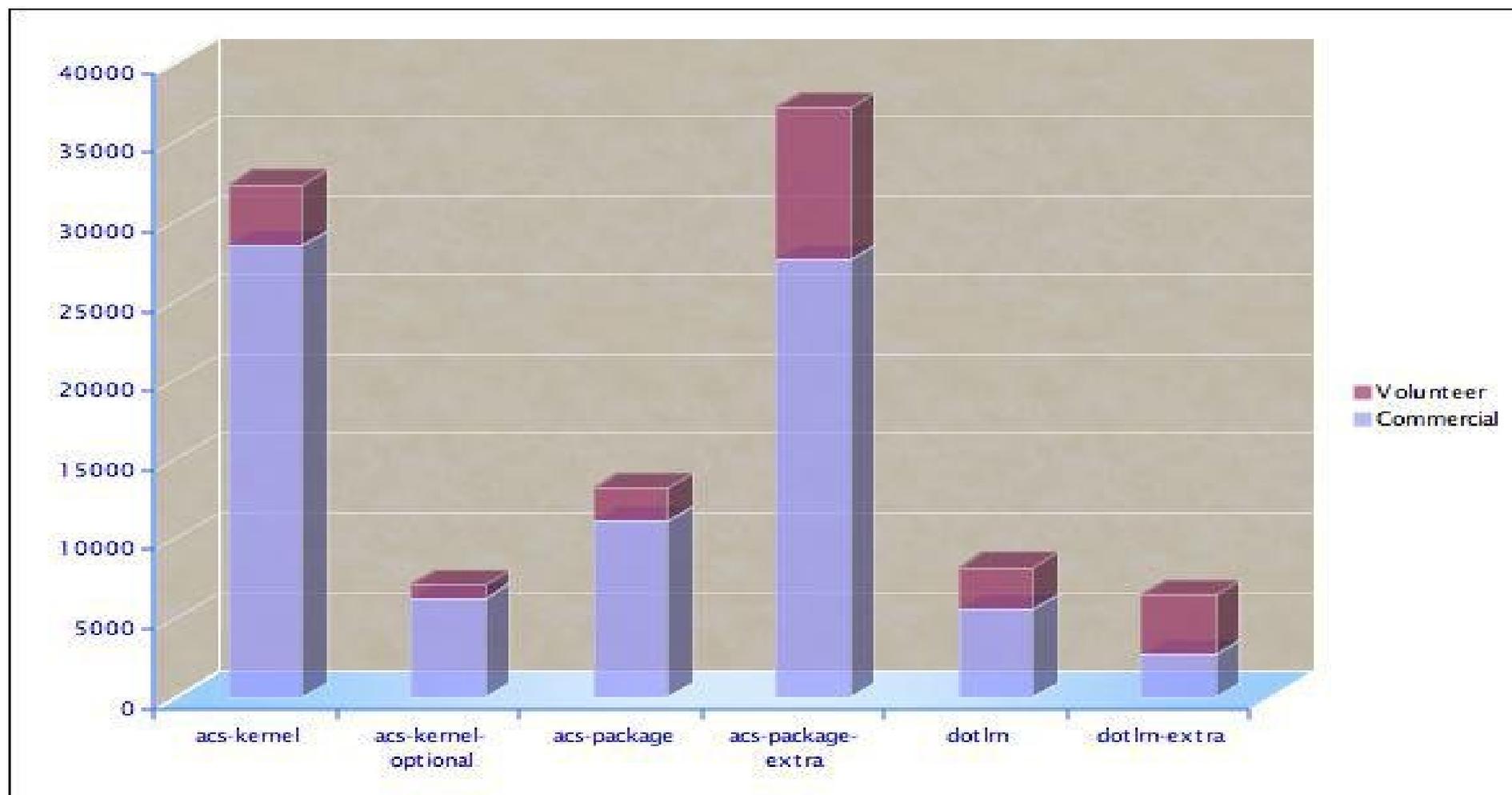
Aplicações Verticais

- www.softwarepublico.gov.br

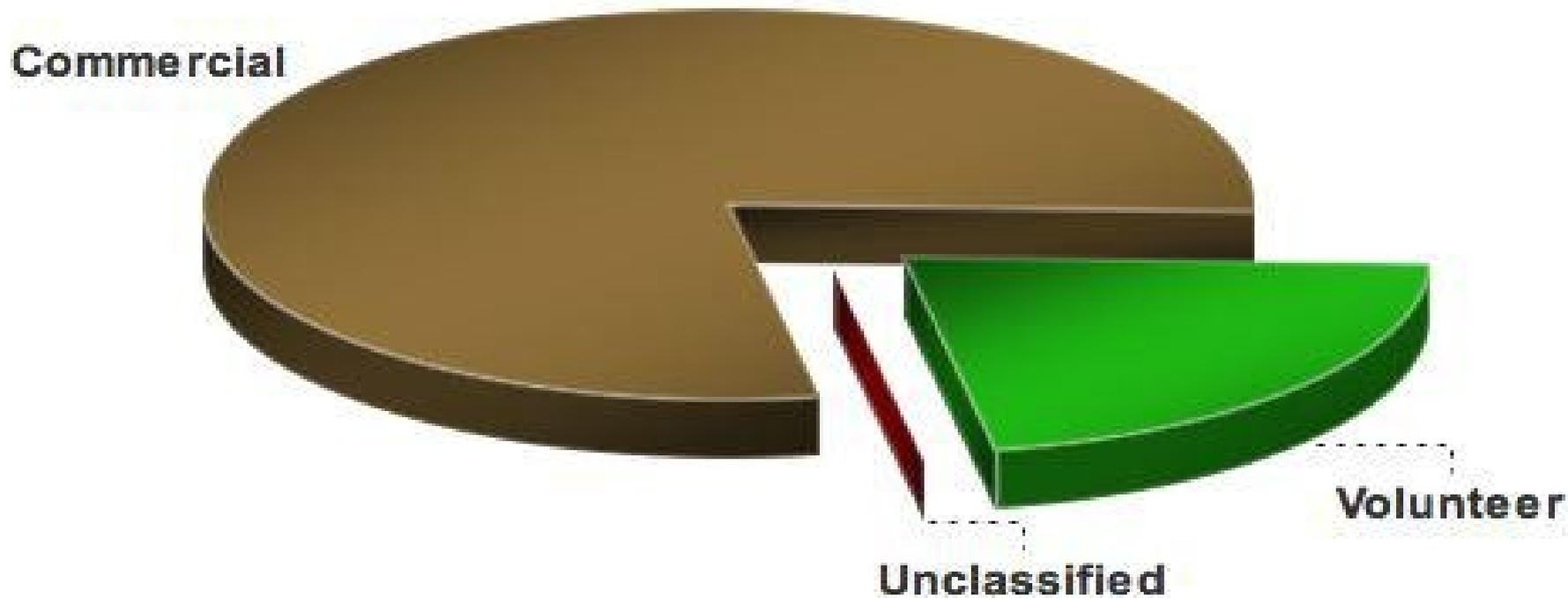
OpenACS: Comunidade

- *“OpenACS is more than a technology, it is a vibrant community”* - Jade Rubick
- Cooperação / moderação / confiança
- Governança
- Componentes testados e maduros
- Suporte comercial / não comercial
- Documentação

OpenACS: Comunidade



OpenACS: Comunidade



OpenACS: Comunidade

- Custo do projeto:
 - Base de código: 1.391.619 linhas de código
 - Esforço estimado: 387 anos de trabalho para uma pessoa
 - Considerando um salário anual médio de US\$ 55.000,00
- Custo estimado: US\$ 21.275.089,00

Fonte: <http://www.ohloh.net/projects/3877?p=OpenACS>

Considerações finais

- Inspirado na apresentação de Orzenil Silva Júnior em:
 - www.softwarepublico.gov.br/dotlrn/clubs/openacs
- Links para conhecer mais:
 - www.dotlrn.org
 - www.openacs.org
 - www.softwarepublico.gov.br
- Comunidade brasileira:
 - www.softwarepublico.gov.br/dotlrn/clubs/openacs

Obrigado

Eduardo Santos

eduardo.edusantos@gmail.com
eduardo.santos@planejamento.gov.br

www.softwarepublico.gov.br