ARVIN MILANEZ JÚNIOR
CARLOS FELIPE GASPARIN GERONASSO
EDSON FERNANDO MARINHO DE MELO
LUCAS RAVEDUTTI LUCIO MACHADO

SISTEMA DE CONFERÊNCIA CHAT E ÁUDIO PARA A ÁREA MÉDICA

Curitiba – Paraná
Janeiro de 2006

ARVIN MILANEZ JÚNIOR
CARLOS FELIPE GASPARIN GERONASSO
EDSON FERNANDO MARINHO DE MELO
LUCAS RAVEDUTTI LUCIO MACHADO

SISTEMA DE CONFERÊNCIA CHAT E ÁUDIO PARA A ÁREA MÉDICA

Monografia apresentada como requisito parcial à obtenção do grau de Tecnólogo em Informática, curso de Tecnologia em Informática, Escola Técnica da Universidade Federal do Paraná.

Orientadora: Profa MsC Jeroniza Nunes Marchaukoski

Curitiba - Paraná
Janeiro de 2006

SUMÁRIO

1. PUBLIC

1.1. index.html

```
<meta http-equiv="refresh" content="1;URL=sistema.php?modulo=usuario&action=usuario.index">
```

1.2. style.css

```
/* estruturais */

body table td {
  font-family: "Trebuchet MS", "Bitstream Vera Sans", tahoma;
  font-size: 13px;
  background-color: #FFFFFF;
}

input, textarea, select {
    padding: 1px;
    color: #555555;
}

input:focus, textarea:focus, select:focus {
    border-bottom: #ffdead solid 2px;
    border-right: #ffdead solid 2px;
    border-left: #c07300 solid 2px;
    border-top:  #c07300 solid 2px;
    color: #000;
}

/* gerais */

table.geral {
    background-color: #FFFFFF;
    width: 100%;
}

td.geral {
    padding: 10px;
    vertical-align: top;
}
/* Especificações do Título do form*/
p.title {
    background-color: #a0bedd;
    border: 1px;
    padding: 2px;
    font-size: 20px;
    font-weight: bold;
}

/* Especificações do Form */
table.form {
    background-color: #E4E3E0;
    border: 0px;
    border-color: #FFFFFF;
    padding: 2px;
    width: 80%;
}

table.form_peq {
    background-color: #E4E3E0;
    border: 0px;
    border-color: #FFFFFF;
    color : #1C5078
    padding: 2px;
    width: 70%;
}
```

```
table.form_peq2 {
    background-color: #CFCFCE;
    border: 0px;
    border-color: #FFFFFF;
    color : #1C5078
    padding: 2px;
    width: 100%;
}

table.empty {
    background-color: #FFFFFF;
    border: 1px solid silver;
    border-color: #B0B0B0;
    padding: 2px;
    width: 50%;
}


td.form_label {
  background-color:#E4E3E0;
  width: 30%;
  text-align: right;
  color :#1C5078;
  vertical-align: top;
  font-weight: bold;
}

td.form_field {
  background-color: #E4E3E0;
  width: 70%;
}

td.error {
  background-color: #C5DBE9;
}

div.error {
  padding: 2px;
  padding-left: 10px;
  border: 1px solid #FF0000;
  background-color: #FFFF99;
  font-weight: bold;
  color: #FF0000;
  width: 100%;
  margin-left: 0%;
}

div.nome {
  font-weight: bold;
  color: #22457F;
}


/* Especificações das Listas */

/* Cor do fundo da tabela listar */
table.list {
    background-color: #FFFFFF;
    border: 0px;
```

```
    border-color: #599EB5;
    padding: 2px;
    width: 100%;
}

table.list_peq {
    background-color: #FFFFFF;
    border: 0px;
    padding: 0px;
    width: 90%;
}

td.list_header {
    background-image : url('/workspace/base/public/imagem/barra_cinza.gif');
    padding: 2px;
    border: 0px;
    font-weight: bold;
    color: #294165;
}

td.list_header1 {
    background-image : url('/workspace/base/public/imagem/barra_amarela.gif');
    padding: 2px;
    font-weight: bold;
    color: #294165;
}

/* Listas mais escuras do Listar */
td.list_body1 {
    background-color: #CDCECF;
    color : #2C5370;
    padding: 0px;
    vertical-align:top;
}

/* Listas mais claras do Listar */
td.list_body2 {
    background-color: #E4E3E0;
    color : #2C5370;
    padding: 0px;
    vertical-align:top;
}

td.message {
    background-color: #E8E8E8;
    vertical-align: middle;
}

/* Listas mais escuras do Listar */
td.list_upload {
    background-color: #D5DDE3;
    color : #2C5370;
    padding: 0px;
    vertical-align:top;
}

div.message {
  font-weight: bold;
}
```

```
/* Especificações do Pé da página */
td.footer {
  padding-left: 10px;
  padding-right: 10px;
  text-align: right;
  border: 1px solid silver;
  width: 100%;
}

/* Especificações para todos os botões */
input.button {
  background-image : url("/workspace/base/public/imagem/xp_button_on.gif");
  font-family:tahoma;
  font-size:12;
  border:1px;
  border-style:solid;
  border-width:1;
  border-color:#073A78;
}

/* Células do Topo */
td.topo_chat {
  background-color: #D5DEE3;
  color : #1D3D4F;
}

/* Células do Form */
td.form_chat {
  background-color: #328CBE;
}

/* Título do chat, no pop-up do chat */
td.form_titulo {
  background-color: #328CBE;
  color:#FFFFFF;
  font-size:12px;
}

/* Células do Form */
td.usuarios_chat {
  background-color: #D8D8D8;
}

/* Configurações da Linha <hr> */
hr {
    border: none 0;
    border-top: 1px solid #B6B6B6;
    width: 99%;
    height: 0px;
}

/* Configurações dos Links */
a {
  color: #ad0d02;
  text-decoration: none;
}
a:active {
  color: #ad0d02;
}
a:visited {
```

```
  color: #ad0d02;
}
a:hover {
  color: #388925;
}
```

1.3. config.php

```
<?php


////////////////////////////////////////////////////////////////////////
//// FUNÇÃO QUE PRINTA_R
function p($var) {
    echo "<hr><pre><font color='blue' size='5'>";
    print_r($var);
    echo "<hr></font>";
}
////////////////////////////////////////////////////////////////////////

/**
 * Classe que armazena as configuraçoes necessárias para execução do sistema em MVC.
 *
 * @package config
 * @todo verificar o diretório das imagens _IMAGEM_DIR_ que depende da zona de DNS de cada
servidor
 */

// ---------- CONFIGURACOES RELATIVAS AO PATH DE CADA ELEMENTO

/** Diretório raíz do sistema. Armazena as configurações e os módulos. */
define("_ROOT_PATH_", dirname(__FILE__) . "/../");

/** Diretório que armazena as bibliotecas comuns do sistema. */
define("_LIB_PATH_", _ROOT_PATH_ . "lib/");

////////////////////////////////////////////////////////////////////////////////

/** Nome do servidor (diretório visível do apache). */
define("_HTTP_NAME_", "http://" . $_SERVER["SERVER_NAME"] .
dirname($_SERVER["PHP_SELF"]));
////////////////////////////////////////////////////////////////////////////////

/** Diretório que armazena os módulos do sistema. */
define("_MODULE_PATH_", _ROOT_PATH_ . "modulos/");

/** Nome do módulo base/default do sistema (responsável pelo <kbd>{@link Controller}</kbd> e
permissões). */
define("_MODULE_BASE_", "_base");

/** Nome do diretório utilizado para armazenar as classes dentro de cada módulo. */
define("_CLASSES_DIR_", "classes");

/** Nome do diretório utilizado para armazenar os templates dentro de cada módulo. */
define("_TEMPLATES_DIR_", "templates");

/** Nome do arquivo de configuração que deve estar presente dentro de cada módulo. */
define("_CONFIG_FILE_", "config.xml");

/** Nome do diretório utilizado para armazenar as imagens. */
define("_IMAGEM_DIR_", "./imagem/");

/** Nome do diretório utilizado para armazenar os sons. */
define("_SOUND_DIR_", "./sound/");
```

```
////////////////////////////////////////////////////////////////////////////////
/** Nome do diretório utilizado para armazenar javascripts. */
define("_JAVASCRIPT_DIR_", _HTTP_NAME_ . "/javascript/");

// ---------- CONFIGURACOES DO BANCO DE DADOS
////////////////////////////////////////////////////////////////////////////////

/** Configuração do Banco de Dados: SGBD (utilizado na URL de conexão do PDO - ex.: pgsql). */
define("_DB_SBGD_", "pgsql");

/** Configuração do Banco de Dados: Máquina em que ele está armazenado. */
define("_DB_HOST_", "localhost");

/** Configuração do Banco de Dados: Nome da Base de Dados. */
define("_DB_NAME_", "academico");

/** Configuração do Banco de Dados: Usuário para acesso à Base de Dados. */
define("_DB_USER_", "academico");

/** Configuração do Banco de Dados: Senha para acesso à Base de Dados. */
define("_DB_PASS_", "academico");

// ---------- CONFIGURACOES RELATIVAS AOS DIRETORIOS UTILIZADOS PELO SMARTY

/** Diretório utilizado pelo Smarty para armazenamento de configurações. */
define("_SMARTY_CONFIG_DIR_",  _LIB_PATH_ . "Smarty/smarty_config");

/** Diretório utilizado pelo Smarty para armazenamento de cache. */
define("_SMARTY_CACHE_DIR_",   _LIB_PATH_ . "Smarty/smarty_cache");

/** Diretório utilizado pelo Smarty para armazenamento de templates compilados. */
define("_SMARTY_COMPILE_DIR_", _LIB_PATH_ . "Smarty/smarty_templates_c");

// ---------- CONSTANTES GERAIS

/** Parâmetro utilizado no retorno da {@link Action}: armazena o resultado da execução (sucesso,
erro, input, etc). */
define("PARAM_ACTION_RESULT", "actionResult");

////////////////////////////////////////////////////////////////////////////////

/** Parâmetro utilizado no retorno da {@link Controller}: verifica qual o parametro passado para a URL
(FORWARD). */
define("FORWARD_PARAM", "forwardParam");

////////////////////////////////////////////////////////////////////////////////

/** Parâmetro utilizado no retorno da {@link Controller}: indica quando não será utilizado o Header e o
Footer Padrão. */
define("REQUEST_HEADER_FOOTER", "requestHeaderFooter");

////////////////////////////////////////////////////////////////////////////////

/** Parâmetro de configuração do título do sistema */
define("_SISTEMA_TITLE_", "..:: Disciplina Virtual ::..");
////////////////////////////////////////////////////////////////////////////////

/** Parâmetro utilizado no retorno da {@link Action}: armazena os possíveis erros de validação. */
define("PARAM_ERRORS", "errors");
```

```
// ---------- MENSAGENS

/** Mensagem padrão: retorno de lista vazia. */
define("MESSAGE_EMPTY_LIST", "Nenhum registro encontrado.");

/**
 * Carrega as configurações do módulo a partir do arquivo solicitado.
 *
 * Esse arquivo deve armazenas as configurações de actions e views do módulo, constantes, etc.
 * Deve estar no formato XML na documentação externa do MVC. Algumas configurações (definição
de
 * constantes, p.ex.) são carregadas em memória, enquanto outras são retornadas (o mapeamento
das
 * {@link Action}s do módulo).
 *
 * @param string $file Nome do arquivo para carregamento das configurações.
 * @return array Retorna um <kbd>array</kbd> contendo os mapeamentos das {@link Action}s do
módulo.
 */
function _loadConfig($file) {
    $xml = simplexml_load_string(file_get_contents($file));

    // carrega as constantes do modulo
    foreach ($xml->constant as $constant) {
        $name = (string)$constant["name"];
        define($name, $value);
    }

    // ultimo passo: carrega os action mappings e retorna
    $actionMapping = array();
    foreach ($xml->action as $tmpAction) {
        $actionName = (string)$tmpAction["name"];
        $actionClass = (string)$tmpAction["class"];
        $actionMethod = (string)$tmpAction["method"];
        $actionResults = array();

        foreach ($tmpAction->result as $tmpResult) {
            $resultName = (string)$tmpResult["name"];
            $resultType = (string)$tmpResult["type"];
            $resultPath = (string)$tmpResult;

            $actionResults[$resultName] = array("name" => $resultName,
                                                "type" => $resultType,
                                                "path" => $resultPath);
        }

        $actionMapping[$actionName] = array("name" => $actionName,
                                            "class" => $actionClass,
                                            "method" => $actionMethod,
                                            "results" => $actionResults);
    }

    return $actionMapping;
}

/**
 * Retorna um valor armazenado no escopo solicitado.
 *
 * @param string $param Nome do parâmetro desejado.
 */
```

```
function getParameter($param, $arr) {

//     echo "Praram: $param <hr>";
//     echo "<pre>";
//     print_r($arr);
//     echo "<hr>";

    return (array_key_exists($param, $arr) ? $arr[$param] : "");
}

/**
 * Retorna um valor armazenado no escopo request.
 *
 * @param string $param Nome do parâmetro desejado.
 */
function getRequestParameter($param) {
    return getParameter($param, $_REQUEST);
}

/**
 * Função padrão do PHP para carregamento de classes quando ocorre alguma falha no
 * seu carregamento. É utilizado como um mecanismo para o carregamento das classes
 * sem haver a necessidade de <kbd>include</kbd>. A classe solicitada é buscada
 * primeiramente no módulo base e, não sendo encontrada, em seguida no módulo corrente.
 *
 * @param string $className Nome da classe em questão.
 */
function __autoload($className) {
    // array contendo os diretórios dos módulos
    $arrModuleDir = array_slice(scandir(_MODULE_PATH_),2);
    foreach ($arrModuleDir as $dirModuleName) {
        $dirModule = _MODULE_PATH_ . $dirModuleName . "/" . _CLASSES_DIR_;
        _loadClassFromDir($dirModule, $className);
    }
}

//function __autoload($className) {
//    $dirBase = _MODULE_PATH_ . _MODULE_BASE_ . "/" . _CLASSES_DIR_;
//    $dirModule = _MODULE_PATH_ . MODULE_DIR . "/" . _CLASSES_DIR_;
//
//    $found = _loadClassFromDir($dirBase, $className); // varrendo modulo base
//    if ((!$found) && ($dirBase != $dirModule)) {
//        $found = _loadClassFromDir($dirModule, $className); // varrendo modulo corrente
//    }
//
//    if (!$found) {
//        throw new Exception("Classe ou Interface não-encontrada: $className.");
//    }
//}

/**
 * Pesquisa pela classe solicitada em um determinado diretório.
 *
 * Atua da seguinte forma: lê o conteúdo do diretório e verifica os elementos
 * encontrados. Se for um arquivo correspondente à classe solicitada, carrega a classe;
 * se for um diretório, pesquisa nesse diretório recursivamente; se
 * for diferente disso, ignora.
 *
 * @param string $dir Diretório a ser varrido.
 * @param string $className Nome da classe a ser carregada.
```

```
 * @return bool Retorna <kbd>true</kbd> caso consiga carregar a classe, <kbd>false</kbd> em caso
contrário.
 */
function _loadClassFromDir($dir, $className) {
    $classFile = $className . ".class.php";
    $found = false;
    $arr = scandir($dir);
    for ($i = 0; ($i < count($arr)) && (!$found); $i++) {
        $f = $arr[$i];
        $path = "$dir/$f";
        if (($f == $classFile) && (is_file($path))) {
            require_once($path);
            $found = true;
        } else if (($f != ".") && ($f != "..") && (is_dir($path))) {
            $found = _loadClassFromDir($path, $className);
        }
    }

    return $found;
}

////////////////////////////////////////////////////////////////////////////////
////////////////////////// CONFIGURACOES DO ENVIO DE EMAIL ////////////////////////////

/** Configuração do envio de email: Remetente da mensagem do email. */
define("_MAIL_FROM_", "arvin@ufpr.br");

/** Configuração do envio de email: Remetente da mensagem do email. */
define("_MAIL_ASSUNTO_", "[WorldConf]");

// ---------- CONFIGURACOES DO ENVIO DE EMAIL
/** Configuração do Servidor de SMTP. */
define("_SMTP_HOST_","vulcao.ufpr.br");

// ---------- CONFIGURACOES DO ENVIO DE EMAIL
/** Configuração da Porta do Servidor de SMTP. */
define("_SMTP_PORTA_","25");

// ---------- CONFIGURACOES DO ENVIO DE EMAIL
/** Configuração do Usuário de Autenticação no Servidor de SMTP. */
define("_SMTP_USER_","");

// ---------- CONFIGURACOES DO ENVIO DE EMAIL
/** Configuração da Senha do Usuário de Autenticação no Servidor de SMTP. */
define("_SMTP_PASS_","");

////////////////////////// FINAL CONFIGURAÇÃO DO ENVIO DE EMAIL ////////////////////////
////////////////////////////////////////////////////////////////////////////////


?>
```

1.4. sistema.php

```
<?php

/**
 * Carrega as configurações do módulo solicitado e actiona o {@link Controller} para
 * processamento da requisição.
 *
 * @package config
 */

// -------- includes
require_once("config.php");
session_start();

// verificar o modulo e action solicitados
if (($_GET["modulo"] == "usuario") AND ($_GET["action"] == "chat.list") AND $_SESSION["opcao"] ==
2) {
   $modulo = "chat";
   $action = getParameter("action", $_GET);
}
elseif (($_GET["modulo"] == "usuario") AND ($_GET["action"] == "chat.list") AND
$_SESSION["opcao"] == 3) {
   $action = "usuario.list";
   $modulo = array_key_exists("modulo", $_GET) ? $_GET["modulo"] : _MODULE_BASE_;
}
else{
   $modulo = array_key_exists("modulo", $_GET) ? $_GET["modulo"] : _MODULE_BASE_;
   $action = getParameter("action", $_GET);
}




/** Diretório raíz do módulo corrente (solicitado nesta requisição). */
define("MODULE_DIR", $modulo); // TODO verificar escopo (application ou request?)

// obter configs do modulo selecionado
$configFile = _MODULE_PATH_ . "$modulo/" . _CONFIG_FILE_;
$actionMapping = _loadConfig($configFile);

// criar um controller especifico para o modulo
$controller = new Controller($actionMapping);

// tratar requisicao
$controller->handleRequest($action);

?>
```

2. MÓDULO BASE – CONFIGURAÇÃO

2.1. config.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<config>

    <!-- CONSTANTES VALIDAS PARA ESTE MODULO -->

    <!-- constante obrigatoria: nome do modulo -->
    <constant name="MODULE_NAME" value="Módulo Base"/>

    <!-- ACTIONS VALIDAS PARA ESTE MODULO -->

    <!-- Action Inicial -->
    <action name="index" class="IndexAction" method="execute">
        <result name="success" type="smarty">index.tpl</result>
    </action>

</config>
```

3. MÓDULO BASE – CLASSES PACOTE CONTROL

3.1. Action.class.php

```
<?php

/**
 * @package control
 */

/**
 * Classe que define uma action. Toda as actions devem herdar desta classe.
 *
 * Os métodos das classes Action que sejam responsáveis pelo tratamento das solicitações
 * devem seguir o padrão de (1) não receberem parâmetros e (2) retornarem um <kbd>array</kbd>
 * contendo os dados obtidos no processamento da Action (mais detalhes na documentação do
método
 * {@link Action::execute()}.
 *
 * @package control
 */
class Action {

    // ---------- ACTION RESULTS PRÉ-CONFIGURADOS

    /** Action Result pré-configurado: SUCCESS. */
    const SUCCESS = "success";

    /** Action Result pré-configurado: INPUT. */
    const INPUT   = "input";

    /** Action Result pré-configurado: ERROR. */
    const ERROR   = "error";

    // ---------- RESULT TYPES SUPORTADOS PELO SISTEMA

    /** Result Type suportado pelo sistema: SMARTY (processamento de template do Smarty). */
    const RESULT_TYPE_SMARTY = "smarty";

    /** Result Type suportado pelo sistema: ACTION (redirect para outra action). */
    const RESULT_TYPE_ACTION = "action";

    /**
     * Método de execução default das Actions. Caso a configuração não determine nenhum método
     * específico para o processamento de determinada action requisitada, este será o método
utilizado.
     *
     * Todo método utilizado para processamento de actions (este ou qualquer outro não-default,
     * definido através do XML de configuração do módulo) deve retornar um <kbd>array</kbd>,
     * contendo obrigatoriamente o resultado da execução (SUCCESS, INPUT, ERROR ou qualquer
outro
     * personalizado), com o qual será verificado pelo sistema qual deve ser a view a ser utilizada.
     *
     * Neste <kbd>array</kbd>, além do elemento resultado (que deve ficar sob a chave
     * {@link PARAM_ACTION_RESULT}), os dados a serem exibidos na view devem ser enviados.
     *
     * @return array Retorna os dados resultantes do processamento da Action.
     */
    public function execute() {
```

```
        return array(PARAM_ACTION_RESULT => Action::SUCCESS);
    }

    /**
     * Efetua a validação de um objeto do Modelo. Retorna um <kbd>array</kbd> de erros de validação
-
     * caso o <kbd>array</kbd> esteja vazio não houve nenhum erro.
     *
     * @return array Retorna os erros de validação (caso não ocorram erros, o <kbd>array</kbd>
estará vazio).
     */
    public function validate($model) {
        return array();
    }

}

?>
```

3.2. Controller.class.php

```
<?php

/**
 * @package control
 */

/**
 * Controlador responsável pela coordenação da execução das {@link Action}s e
 * renderização das views adequadas.
 *
 * Este é o coração da camada Control do MVC. A camada Control faz a ponte entre a
 * camadas View e Model no padrão MVC.
 *
 * @package control
 */
class Controller {

    /** Armazena os mapeamentos das actions do módulo que está sendo executado. */
    private $actionMapping;

    /**
     * Construtor da classe. Recebe os mapeamentos do módulo que deverá tratar.
     *
     * @param array $actionMapping Mapeamentos do módulo a ser tratado.
     */
    public function Controller($actionMapping) {
        $this->actionMapping = $actionMapping;

//         echo "Controller.class.php->Controller<pre>";
//         print_r($actionMapping);
//
// echo MODULE_NAME;
//         foreach ($actionMapping as $chave => $valor) {
//             echo "<hr>Chave: $chave";
//         }

    }

    /**
     * Efetua o tratamento de uma requisição. Este é o método principal - ele recebe como
     * parâmetro a action requisitada, realiza o processo de execução da mesma segundo as
     * informações da requisição e no decorrer de sua execução provê a geração/renderização
     * da view adequada para a action e para o resultado de sua execução.
     *
     * @param string $actionName Action para a qual o tratamento da requisição deve ser delegado.
     */
    public function handleRequest($actionName) {

        try {

            // obtem dados da action solicitada e instancia classe adequada
            $mapping = $this->getActionMappingFromName($actionName);

            $class = $mapping["class"];
            $method = $mapping["method"];

            // solicita execucao da action adequada e armazena dados de retorno
            $output = $this->executeAction($class, $method);
```

```
            // verifica resultado da execucao
            $result = $output[PARAM_ACTION_RESULT];

            // verifica qual o parametro passado para a URL (FORWARD)
            $param = $output[FORWARD_PARAM];

            // verifica qual a view correspondente para o resultado nas confs da action
            $view = $this->getView($mapping, $result, $param);

            // renderiza a view
            print($this->getDecoratedContent($view, $output));

        } catch (Exception $e) {

            Util::logException($e);

        }

    }

    /**
     * Retorna os dados da action solicitada através de pesquisa nos mapeamentos armazenados.
     *
     * @param string $name Action para pesquisa de configurações.
     * @return array Configuração da action solicitada.
     */
    private function getActionMappingFromName($name) {
        if (!array_key_exists($name, $this->actionMapping)) {
            throw new Exception("Action não-encontrada: $name.");
        }
        return $this->actionMapping[$name];
    }

    /**
     * Efetua a execução da action solicitada, com base nos nomes de classe e método passados
como parâmetro.
     *
     * @param string $class Classe Action a ser utilizada.
     * @param string $method Nome do método a ser utilizado (opcional).
     * @return array Retorna o <kbd>array</kbd> de dados retornados pela action (consulte método
{@link Action::execute()}).
     */
    private function executeAction($class, $method = null) {
        $action = new $class();
        if (($method == null) || ($method == "")) {
            $method = "execute"; // metodo default das actions
        }
        return $action->$method();
    }

    /**
     * Verifica a view que deve ser utilizada para apresentação dos resultados da execução de uma
action.
     *
     * Seu comportamento depende do tipo de resultado (Result Type) definido para a combinação de
Action
     * e Resultado (Action Result) passados como parâmetro:
     *
```

```
    * - {@ link Action::RESULT_TYPE_SMARTY Action::RESULT_TYPE_SMARTY}: retorna o nome
do template a ser processado.
    * - {@ link Action::RESULT_TYPE_ACTION Action::RESULT_TYPE_ACTION}: efetua um redirect
para a action configurada.
    *
    * @param array $mapping Mapeamento contendo configurações da Action em questão.
    * @param string $result Nome do Resultado obtido na execução da Action (Action Result).
    * @return string Retorna o template adequado no caso de result type "smarty". Para result type
"action",
    * o método não conclui sua execução, pois um redirecionamento é efetuado.
    */
   private function getView($mapping, $result, $param) {

      $actionName = $mapping["name"];
      $results = $mapping["results"];


      if ((!is_array($results)) || (count($results) == 0)) {
         throw new Exception("Action \"$actionName\" não tem configuração adequada de Views.");
      }

      if (!array_key_exists($result, $results)) {
         throw new Exception("Action \"$actionName\" não tem Result \"$result\" configurado.");
      }

      $resultType = $results[$result]["type"];
      $resultPath = $results[$result]["path"];

      switch ($resultType) {
         case Action::RESULT_TYPE_SMARTY :
            return $resultPath;
         case Action::RESULT_TYPE_ACTION :
            header("location: " . Util::getURL($resultPath, $param));
            exit();
         default :
            throw new Exception("Result Type desconhecido: $resultType.");
      }

   }

   /**
    * Processa o template solicitado e retorna o resultado decorado pelos conteúdos default, como
cabeçalho e rodapé.
    *
    * @param string $viewName Nome do template a ser processado.
    * @param array $output <kbd>Array</kbd> contendo dados a serem processados pelo template.
    * @return string Retorna o resultado do processamento do template solicitado, incluindo os
conteúdos utilizados
    * para decorar a view.
    */
   private function getDecoratedContent($viewName, $output) {


      // conteudo principal
      $smarty = new SmartyView(MODULE_DIR);
      $smarty->assign($output);
      $content = $smarty->fetch($viewName);

      // decoracoes
      $smartyBase = new SmartyView();
```

```
        $smartyBase->assign($output);

        /**
         * Condição IF Responsável pela Utilização do Header e Footer Padrões
         * A passagem de qualquer parâmetro, implicará na utilização da página sem o template Header
e o Footer
         */

        // flag para a utilização do Header e Footer Padrões
        if ($output[REQUEST_HEADER_FOOTER] == "1") {
            $header = $smartyBase->fetch("header_visitante.tpl");
            $footer = $smartyBase->fetch("footer_visitante.tpl");
        }
        elseif ($output[REQUEST_HEADER_FOOTER] == "2") {
            $header = $smartyBase->fetch("header_usuario.tpl");
            $footer = $smartyBase->fetch("footer.tpl");
        }
        elseif ($output[REQUEST_HEADER_FOOTER] == "3") {
            $header = $smartyBase->fetch("header_admin.tpl");
            $footer = $smartyBase->fetch("footer_admin.tpl");
        }

        return $header . $content . $footer;

    }

}

?>
```

3.3. IndexAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action básica que apenas retorna SUCCESS (segue comportamento default da classe
 * {@link Action}).
 *
 * @see Action
 * @package control
 */
class IndexAction extends Action {
}

?>
```

4. MÓDULO BASE – CLASSES PACOTE UTIL

4.1. Util.class.php

```
<?php

/**
 * @package util
 */

/** Dependência de biblioteca externa: Log. */
require_once(_LIB_PATH_ . "Log/Log.php");
/** Requerimento da Classe Mail do PEAR */
require_once("Mail.php");

/**
 * Classe auxiliar. Contém métodos <kbd>static</kbd> utilitários para log, formatação de URLs, etc.
 *
 * @static
 * @package util
 */
class Util {

    /**
     * Efetua log em nível ERROR dos dados da exceção passada como parâmetro.
     *
     * @param Exception $e Exceção a ser logada.
     */
    public static function logException($e) {
        $str = "Ocorreu a seguinte exceção:\n";
        $str .= "Message   : " . $e->getMessage() . "\n";
        $str .= "Code      : " . $e->getCode() . "\n";
        $str .= "File      : " . $e->getFile() . "\n";
        $str .= "Line      : " . $e->getLine() . "\n";
        $str .= "StackTrace:\n" . $e->getTraceAsString();

        Util::logError($str);
    }

    /**
     * Efetua log em nível ERROR da mensagem passada como parâmetro.
     *
     * @param string $msg Mensagem a ser logada.
     */
    public static function logError($msg) {
        Util::log($msg, PEAR_LOG_ERR);
    }

    /**
     * Efetua log em nível DEBUG da mensagem passada como parâmetro.
     *
     * @param string $msg Mensagem a ser logada.
     */
    public static function logDebug($msg) {
        Util::log($msg, PEAR_LOG_DEBUG);
    }

    /**
     * Efetua log da mensagem passada como parâmetro no nível solicitado. Este método
```

```
     * é privado pois não deve ser utilizado diretamente, apenas através dos métodos
     * públicos disponíveis para log.
     *
     * @param string $msg Mensagem a ser logada.
     * @param int $level Nível de log desejado.
     */
    private static function log($msg, $level) {
        $conf = array("error_prepend" => '<font color="#FF0000"><tt>',
                      "error_append"  => '</tt></font>');
        $logger = &Log::singleton("display", "", MODULE_NAME . " (" . MODULE_DIR . ")", $conf);
        $logger->log($msg, $level);
    }

    /**
     * Formata e retorna uma URL no formato esperado pelo sistema. Deve ser utilizado
     * pelos módulos para geração de URLs no formato adequado (que inclui o nome do módulo,
     * da {@link Action}, além dos parâmetros adicionais).
     *
     * A utilização deste método é importante, uma vez que os módulos não devem formatar
     * suas URLs manualmente, pois caso a base do sistema seja alterada as URLs ficarão
     * quebradas.
     *
     * A utilização do método é simples. Por exemplo, para acionar a action "curso.delete",
     * passando como parâmetro ID = 1 e cascata = "true", basta utilizar a URL gerada pela
     * seguinte chamada:
     *
     * <code>
     * Util::getURL("curso.delete", "id=1&cascata=true");
     * </code>
     *
     * @param string $action Action para a qual a URL vai apontar.
     * @param string $params Parâmetros adicionais a serem anexados à QUERY_STRING
(opcional).
     * Devem estar no formato QUERY_STRING (ou seja, param1=valor1&param2=valor2...).
     * @return string URL formatada para a action e parâmetros solicitados.
     */
    public static function getURL($action, $params = null) {
        $url = "sistema.php?modulo=" . MODULE_DIR . "&action=$action";
        if ($params != null) {
            $url .= "&$params";
        }
        return $url;
    }

    /**
     * Retorna uma conexão para a base de dados através da classe PDO. Para mais informações
     * sobre essa classe consulte a documentação dos módulos PEAR relacionados ao PDO (PHP
     * Data Objects).
     *
     * As conexões com banco de dados não devem ser abertas manualmente, mas apenas através de
     * chamadas a este método.
     *
     * @return PDO Instância PDO a ser utilizada para comunicação com o banco de dados.
     */
    public static function getPDOHandler() {

        // sqlite - apenas para testes
        //$pdo = new PDO("sqlite:/home/marcio/dvlp/tcc/sqlite_data/academico.data");

        // postgresql
```

```
        $str = sprintf("%s:host=%s dbname=%s", _DB_SBGD_, _DB_HOST_, _DB_NAME_);
        $pdo = new PDO($str, _DB_USER_, _DB_PASS_);

        $pdo->setAttribute(PDO_ATTR_ERRMODE, PDO_ERRMODE_EXCEPTION);

        $stmt = $pdo->prepare("SET DATESTYLE TO SQL,European");
        $stmt->execute();

        return $pdo;
    }

    /**
     * Método que irá validar a data, retorna se a data é válida ou não.
     * O formato da data poderá ser: dd/mm/aaaa, dd-mm-aaaa ou dd.mm.aaaa
     * @return boolean
     * @param $data
     */
    function validaData($data) {
        $data = trim($data);
        if (empty($data)) {
            return true;
            exit;
        }
        if (strlen($data)>10) {
            return true;
            exit;
        }
        $data = split ('[/.-]', $data);
        if (!(ereg("^[[:digit:]]+$", $data[0]))) {
            return true;
            exit;
        }
        if (!(ereg("^[[:digit:]]+$", $data[1]))) {
            return true;
            exit;
        }
        if (!(ereg("^[[:digit:]]+$", $data[2]))) {
            return true;
            exit;
        }

        // checkdate mm/dd/aaaa
        if (checkdate($data[1],$data[0],$data[2])) {
            return false;
        }
        else {
            return true;
        }
    }

     /** Método que irá validar o  cpf, retorna verdadeiro se for um cpf válido ou retornará falso  * para
cpf inválido.
      * @return boolean
      * @param $cpf
      */
     function validaCpf($cpf) {
        $erro = false;
        $aux_cpf = "";
```

```
      if ($cpf == "00000000000" || $cpf == "11111111111" || $cpf == "22222222222" || $cpf ==
"33333333333" || $cpf == "44444444444" || $cpf == "55555555555" || $cpf == "66666666666" || $cpf
== "77777777777" || $cpf == "88888888888" || $cpf == "99999999999") {
          $erro = true;
      }
      for ($j=0; $j<strlen($cpf); $j++)
          if (substr($cpf,$j,1) >= "0" AND substr($cpf,$j,1) <= "9")
              $aux_cpf .= substr($cpf,$j,1);
          if (strlen($aux_cpf) != 11)
              $erro = true;
          else {
              $cpf1 = $aux_cpf;
              $cpf2 = substr($cpf,-2);
              $controle = "";
              $start = 2;
              $end = 10;
      for ($i=1; $i<=2; $i++) {
          $soma = 0;
      for ($j=$start; $j<=$end; $j++)
          $soma += substr($cpf1,($j-$i-1),1) * ($end+1+$i-$j);
          if ($i == 2)
            $soma += $digito * 2;
             $digito = ($soma * 10) % 11;
          if ($digito == 10)
             $digito = 0;
             $controle .= $digito;
             $start = 3;
             $end = 11;
      }
          if ($controle != $cpf2)
             $erro = true;
  }
      return $erro;
  }

     /**
      * Método que irá validar o email, retorna verdadeiro se é um email válido
      * e falso se for um  email inválido.
      * @return boolean
      * @param $email
      */
     function validaEmail($email) {
       $nLen=strlen(trim($email));
         if($nLen==0) {
           return true;
 }
       $carac_prim=$email{0};
       $carac_ult=$email{$nLen-1};

   // email nao deve iniciar ou finalizar com @,.,-
   if(is_integer(strpos("@.-",$carac_prim)) || is_integer(strpos("@.-",$carac_ult))) {
       return true;
 }

   // email nao deve iniciar ou finalizar com numero
   if (is_integer($carac_prim) || is_integer($carac_ult)) {
       return true;
 }

   // verifica sequencia de  2 '..','--','__','@@' no email
```

```
    if((is_integer(strpos(trim($email),".."))) || (is_integer(strpos(trim($email),"--"))) ||
(is_integer(strpos(trim($email),"__"))) || (is_integer(strpos(trim($email),"@@")))) {
        return true;
  }


$caracteres_validos="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ012345678
9.-_@";

    for ($n=0;$n<strlen($email);$n++) {
        $carac_chek="";
        // verifica caracteres validos
        if (!is_integer(strpos($caracteres_validos,$email[$n]))) {
            return true;
    }

        // verifica se existe mais algun caractere depois do @
        if (substr_count(trim($email),"@")!=1) {
            return true;
    }
        // verifica se existe mais de um '.' depois do @
            if (substr_count(trim($email),".")==0) {
                return true;
    }
        // verifica se nao ha caracteres da lista abaixo
             $carac_chek=$carac_ant.trim($email[$n]);
             switch($carac_chek) {
                case '.@' :
                case '@.' :
                case '-@' :
                case '@-' :
                case '_@' :
                case '@_' :
                return true;
                break;
    }

        $carac_ant=trim($email[$n]);
  }

  return false;
}

public static function mailSender($mailFrom, $mailSubject, $mailBody) {

        $body = $mailBody;

        $headers['From'] =  _MAIL_FROM_;

        $headers['To'] = $mailFrom;

        $headers['Subject'] = $mailSubject;
        $headers['Date'] = date('r',time());
//         $headers['Content-Type'] = "text/html; charset=iso-8859-1";


        $recipients = array($headers['To']);

        $params['host'] = _SMTP_HOST_;
        $params['port'] = _SMTP_PORTA_;
```

```
        $params['auth'] = FALSE;
        $params['username'] = _SMTP_USER_;
        $params['password'] = _SMTP_PASS_;

        $mail_object = Mail::factory('smtp', $params);

        $msg = $mail_object->send($recipients, $headers, $body);
    }

}


?>
```

5. MÓDULO BASE – CLASSES PACOTE VIEW

5.1. SmartyView.class.php

```
<?php

/**
 * @package util
 */

/** Dependência de biblioteca externa: Log. */
require_once(_LIB_PATH_ . "Log/Log.php");
/** Requerimento da Classe Mail do PEAR */
require_once("Mail.php");

/**
 * Classe auxiliar. Contêm métodos <kbd>static</kbd> utilitários para log, formatação de URLs, etc.
 *
 * @static
 * @package util
 */
class Util {

    /**
     * Efetua log em nível ERROR dos dados da exceção passada como parâmetro.
     *
     * @param Exception $e Exceção a ser logada.
     */
    public static function logException($e) {
        $str = "Ocorreu a seguinte exceção:\n";
        $str .= "Message   : " . $e->getMessage() . "\n";
        $str .= "Code      : " . $e->getCode() . "\n";
        $str .= "File      : " . $e->getFile() . "\n";
        $str .= "Line      : " . $e->getLine() . "\n";
        $str .= "StackTrace:\n" . $e->getTraceAsString();

        Util::logError($str);
    }

    /**
     * Efetua log em nível ERROR da mensagem passada como parâmetro.
     *
     * @param string $msg Mensagem a ser logada.
     */
    public static function logError($msg) {
        Util::log($msg, PEAR_LOG_ERR);
    }

    /**
     * Efetua log em nível DEBUG da mensagem passada como parâmetro.
     *
     * @param string $msg Mensagem a ser logada.
     */
    public static function logDebug($msg) {
        Util::log($msg, PEAR_LOG_DEBUG);
    }

    /**
     * Efetua log da mensagem passada como parâmetro no nível solicitado. Este método
```

```
     * é privado pois não deve ser utilizado diretamente, apenas através dos métodos
     * públicos disponíveis para log.
     *
     * @param string $msg Mensagem a ser logada.
     * @param int $level Nível de log desejado.
     */
    private static function log($msg, $level) {
        $conf = array("error_prepend" => '<font color="#FF0000"><tt>',
                      "error_append"  => '</tt></font>');
        $logger = &Log::singleton("display", "", MODULE_NAME . " (" . MODULE_DIR . ")", $conf);
        $logger->log($msg, $level);
    }

    /**
     * Formata e retorna uma URL no formato esperado pelo sistema. Deve ser utilizado
     * pelos módulos para geração de URLs no formato adequado (que inclui o nome do módulo,
     * da {@link Action}, além dos parâmetros adicionais).
     *
     * A utilização deste método é importante, uma vez que os módulos não devem formatar
     * suas URLs manualmente, pois caso a base do sistema seja alterada as URLs ficarão
     * quebradas.
     *
     * A utilização do método é simples. Por exemplo, para acionar a action "curso.delete",
     * passando como parâmetro ID = 1 e cascata = "true", basta utilizar a URL gerada pela
     * seguinte chamada:
     *
     * <code>
     * Util::getURL("curso.delete", "id=1&cascata=true");
     * </code>
     *
     * @param string $action Action para a qual a URL vai apontar.
     * @param string $params Parâmetros adicionais a serem anexados à QUERY_STRING
(opcional).
     * Devem estar no formato QUERY_STRING (ou seja, param1=valor1&param2=valor2...).
     * @return string URL formatada para a action e parâmetros solicitados.
     */
    public static function getURL($action, $params = null) {
        $url = "sistema.php?modulo=" . MODULE_DIR . "&action=$action";
        if ($params != null) {
            $url .= "&$params";
        }
        return $url;
    }

    /**
     * Retorna uma conexão para a base de dados através da classe PDO. Para mais informações
     * sobre essa classe consulte a documentação dos módulos PEAR relacionados ao PDO (PHP
     * Data Objects).
     *
     * As conexões com banco de dados não devem ser abertas manualmente, mas apenas através de
     * chamadas a este método.
     *
     * @return PDO Instância PDO a ser utilizada para comunicação com o banco de dados.
     */
    public static function getPDOHandler() {

        // sqlite - apenas para testes
        //$pdo = new PDO("sqlite:/home/marcio/dvlp/tcc/sqlite_data/academico.data");

        // postgresql
```

```
        $str = sprintf("%s:host=%s dbname=%s", _DB_SBGD_, _DB_HOST_, _DB_NAME_);
        $pdo = new PDO($str, _DB_USER_, _DB_PASS_);

        $pdo->setAttribute(PDO_ATTR_ERRMODE, PDO_ERRMODE_EXCEPTION);

        $stmt = $pdo->prepare("SET DATESTYLE TO SQL,European");
        $stmt->execute();

        return $pdo;
    }

    /**
     * Método que irá validar a data, retorna se a data é válida ou não.
     * O formato da data poderá ser: dd/mm/aaaa, dd-mm-aaaa ou dd.mm.aaaa
     * @return boolean
     * @param $data
     */
    function validaData($data) {
        $data = trim($data);
        if (empty($data)) {
            return true;
            exit;
        }
        if (strlen($data)>10) {
            return true;
            exit;
        }
        $data = split ('[/.-]', $data);
        if (!(ereg("^[[:digit:]]+$", $data[0]))) {
            return true;
            exit;
        }
        if (!(ereg("^[[:digit:]]+$", $data[1]))) {
            return true;
            exit;
        }
        if (!(ereg("^[[:digit:]]+$", $data[2]))) {
            return true;
            exit;
        }

        // checkdate mm/dd/aaaa
        if (checkdate($data[1],$data[0],$data[2])) {
            return false;
        }
        else {
            return true;
        }
    }

     /** Método que irá validar o  cpf, retorna verdadeiro se for um cpf válido ou retornará falso  * para
cpf inválido.
      * @return boolean
      * @param $cpf
      */
     function validaCpf($cpf) {
        $erro = false;
        $aux_cpf = "";
```

```
        if ($cpf == "00000000000" || $cpf == "11111111111" || $cpf == "22222222222" || $cpf ==
"33333333333" || $cpf == "44444444444" || $cpf == "55555555555" || $cpf == "66666666666" || $cpf
== "77777777777" || $cpf == "88888888888" || $cpf == "99999999999") {
            $erro = true;
        }
        for ($j=0; $j<strlen($cpf); $j++)
            if (substr($cpf,$j,1) >= "0" AND substr($cpf,$j,1) <= "9")
                $aux_cpf .= substr($cpf,$j,1);
            if (strlen($aux_cpf) != 11)
                $erro = true;
            else {
                $cpf1 = $aux_cpf;
                $cpf2 = substr($cpf,-2);
                $controle = "";
                $start = 2;
                $end = 10;
        for ($i=1; $i<=2; $i++) {
            $soma = 0;
        for ($j=$start; $j<=$end; $j++)
            $soma += substr($cpf1,($j-$i-1),1) * ($end+1+$i-$j);
            if ($i == 2)
              $soma += $digito * 2;
               $digito = ($soma * 10) % 11;
            if ($digito == 10)
               $digito = 0;
               $controle .= $digito;
               $start = 3;
               $end = 11;
        }
            if ($controle != $cpf2)
               $erro = true;
    }
        return $erro;
    }

      /**
       * Método que irá validar o email, retorna verdadeiro se é um email válido
       * e falso se for um  email inválido.
       * @return boolean
       * @param $email
       */
      function validaEmail($email) {
        $nLen=strlen(trim($email));
          if($nLen==0) {
            return true;
  }
        $carac_prim=$email{0};
        $carac_ult=$email{$nLen-1};

    // email nao deve iniciar ou finalizar com @,.,-
    if(is_integer(strpos("@.-",$carac_prim)) || is_integer(strpos("@.-",$carac_ult))) {
        return true;
  }

    // email nao deve iniciar ou finalizar com numero
    if (is_integer($carac_prim) || is_integer($carac_ult)) {
        return true;
  }

    // verifica sequencia de  2 '..','--','__','@@' no email
```

```
    if((is_integer(strpos(trim($email),".."))) || (is_integer(strpos(trim($email),"--"))) ||
(is_integer(strpos(trim($email),"__"))) || (is_integer(strpos(trim($email),"@@")))) {
        return true;
  }


$caracteres_validos="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ012345678
9.-_@";

    for ($n=0;$n<strlen($email);$n++) {
        $carac_chek="";
        // verifica caracteres validos
        if (!is_integer(strpos($caracteres_validos,$email[$n]))) {
            return true;
    }

        // verifica se existe mais algun caractere depois do @
        if (substr_count(trim($email),"@")!=1) {
            return true;
    }
        // verifica se existe mais de um '.' depois do @
            if (substr_count(trim($email),".")==0) {
                return true;
    }
        // verifica se nao ha caracteres da lista abaixo
             $carac_chek=$carac_ant.trim($email[$n]);
             switch($carac_chek) {
                case '.@' :
                case '@.' :
                case '-@' :
                case '@-' :
                case '_@' :
                case '@_' :
                return true;
                break;
    }

        $carac_ant=trim($email[$n]);
  }

  return false;
}

public static function mailSender($mailFrom, $mailSubject, $mailBody) {

        $body = $mailBody;

        $headers['From'] =  _MAIL_FROM_;

        $headers['To'] = $mailFrom;

        $headers['Subject'] = $mailSubject;
        $headers['Date'] = date('r',time());
//         $headers['Content-Type'] = "text/html; charset=iso-8859-1";


        $recipients = array($headers['To']);

        $params['host'] = _SMTP_HOST_;
        $params['port'] = _SMTP_PORTA_;
```

```
        $params['auth'] = FALSE;
        $params['username'] = _SMTP_USER_;
        $params['password'] = _SMTP_PASS_;

        $mail_object = Mail::factory('smtp', $params);

        $msg = $mail_object->send($recipients, $headers, $body);
    }

}


?>
```

6. MÓDULO BASE - TEMPLATES

6.1. header_admin.tpl

```
<html>
<head>
<title>WorldConf - O Mundo fala aqui!</title>
<script language="JavaScript" type="text/JavaScript">
{literal}
<!--
function MM_preloadImages() { //v3.0
  var d=document; if(d.images){ if(!d.MM_p) d.MM_p=new Array();
    var i,j=d.MM_p.length,a=MM_preloadImages.arguments; for(i=0; i<a.length; i++)
    if (a[i].indexOf("#")!=0){ d.MM_p[j]=new Image; d.MM_p[j++].src=a[i];}}
}

function MM_swapImgRestore() { //v3.0
  var i,x,a=document.MM_sr; for(i=0;a&&i<a.length&&(x=a[i])&&x.oSrc;i++) x.src=x.oSrc;
}

function MM_findObj(n, d) { //v4.01
  var p,i,x;  if(!d) d=document; if((p=n.indexOf("?"))>0&&parent.frames.length) {
    d=parent.frames[n.substring(p+1)].document; n=n.substring(0,p);}
  if(!(x=d[n])&&d.all) x=d.all[n]; for (i=0;!x&&i<d.forms.length;i++) x=d.forms[i][n];
  for(i=0;!x&&d.layers&&i<d.layers.length;i++) x=MM_findObj(n,d.layers[i].document);
  if(!x && d.getElementById) x=d.getElementById(n); return x;
}

function MM_swapImage() { //v3.0
  var i,j=0,x,a=MM_swapImage.arguments; document.MM_sr=new Array; for(i=0;i<(a.length-2);i+=3)
   if ((x=MM_findObj(a[i]))!=null){document.MM_sr[j++]=x; if(!x.oSrc) x.oSrc=x.src; x.src=a[i+2];}
}
//-->
{/literal}
</script>
</head>
<link rel="stylesheet" type="text/css" href="style.css"/>
<body bgcolor="#F2F2F2" leftmargin="0" topmargin="0" marginwidth="0" marginheight="0">
<br>
<table align="center" width="780" height="435" border="0" cellpadding="0" cellspacing="0">
        <tr>
                <td colspan="3"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_42.jpg"
width="206" height="49"></td>
                <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_43.jpg"
width="222" height="49"></td>
                <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_44.jpg"
width="220" height="49"></td>
                <td><a href="./sistema.php?modulo=usuario&action=usuario.close"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('sair','','{$smarty.const._IMAGEM_DIR_}sysconf_75.jpg',0)"><img
src="{$smarty.const._IMAGEM_DIR_}sysconf_45.jpg" name="sair" width="132" height="49"
border="0"></a></td>
        </tr>
        <tr>
                <td colspan="3"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_46.jpg"
width="206" height="194"></td>
                <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_47.jpg"
width="222" height="194"></td>
```

```
		<td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_48.jpg"
width="220" height="194"></td>
		<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_49.jpg" width="132"
height="194"></td>
	</tr>
	<tr>
		<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_50.jpg" width="18"
height="56" border="0"></td>
		<td><a href="./sistema.php?modulo=chat&action=chat.load&cod=0"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('criachat','','{$smarty.const._IMAGEM_DIR_}sysconf_68.jpg',0)"><im
g src="{$smarty.const._IMAGEM_DIR_}sysconf_51.jpg" name="criachat" width="87" height="56"
border="0"></a></td>
		<td><a href="./sistema.php?modulo=chat&action=chat.list"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('listachat','','{$smarty.const._IMAGEM_DIR_}sysconf_69.jpg',0)"><im
g src="{$smarty.const._IMAGEM_DIR_}sysconf_52.jpg" name="listachat" width="101" height="56"
border="0"></a></td>
		<td><a href="./sistema.php?modulo=usuario&action=usuario.list"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('listausuario','','{$smarty.const._IMAGEM_DIR_}sysconf_70.jpg',0)">
<img src="{$smarty.const._IMAGEM_DIR_}sysconf_53.jpg" name="listausuario" width="120"
height="56" border="0"></a></td>
		<td><a href="./sistema.php?modulo=usuario&action=grupo.load&cod=0"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('criagrupo','','{$smarty.const._IMAGEM_DIR_}sysconf_71.jpg',0)"><i
mg src="{$smarty.const._IMAGEM_DIR_}sysconf_54.jpg" name="criagrupo" width="102" height="56"
border="0"></a></td>
		<td><a href="./sistema.php?modulo=usuario&action=grupo.list"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('listagrupo','','{$smarty.const._IMAGEM_DIR_}sysconf_72.jpg',0)"><i
mg src="{$smarty.const._IMAGEM_DIR_}sysconf_55.jpg" name="listagrupo" width="109" height="56"
border="0"></a></td>
		<td><a href="./sistema.php?modulo=chat&action=recurso.load&cod=0"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('criarecurso','','{$smarty.const._IMAGEM_DIR_}sysconf_73.jpg',0)">
<img src="{$smarty.const._IMAGEM_DIR_}sysconf_56.jpg" name="criarecurso" width="111"
height="56" border="0"></a></td>
		<td><a href="./sistema.php?modulo=chat&action=recurso.list"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('listarecurso','','{$smarty.const._IMAGEM_DIR_}sysconf_74.jpg',0)">
<img src="{$smarty.const._IMAGEM_DIR_}sysconf_57.jpg" name="listarecurso" width="132"
height="56" border="0"></a></td>
	</tr>
```

6.2. header_usuario.tpl

```
<html>
<head>
<title>WorldConf - O Mundo fala aqui!</title>
<script language="JavaScript" type="text/JavaScript">
{literal}
<!--
function MM_preloadImages() { //v3.0
  var d=document; if(d.images){ if(!d.MM_p) d.MM_p=new Array();
    var i,j=d.MM_p.length,a=MM_preloadImages.arguments; for(i=0; i<a.length; i++)
    if (a[i].indexOf("#")!=0){ d.MM_p[j]=new Image; d.MM_p[j++].src=a[i];}}
}

function MM_swapImgRestore() { //v3.0
  var i,x,a=document.MM_sr; for(i=0;a&&i<a.length&&(x=a[i])&&x.oSrc;i++) x.src=x.oSrc;
}

function MM_findObj(n, d) { //v4.01
  var p,i,x;  if(!d) d=document; if((p=n.indexOf("?"))>0&&parent.frames.length) {
    d=parent.frames[n.substring(p+1)].document; n=n.substring(0,p);}
  if(!(x=d[n])&&d.all) x=d.all[n]; for (i=0;!x&&i<d.forms.length;i++) x=d.forms[i][n];
  for(i=0;!x&&d.layers&&i<d.layers.length;i++) x=MM_findObj(n,d.layers[i].document);
  if(!x && d.getElementById) x=d.getElementById(n); return x;
}

function MM_swapImage() { //v3.0
  var i,j=0,x,a=MM_swapImage.arguments; document.MM_sr=new Array; for(i=0;i<(a.length-2);i+=3)
   if ((x=MM_findObj(a[i]))!=null){document.MM_sr[j++]=x; if(!x.oSrc) x.oSrc=x.src; x.src=a[i+2];}
}
//-->
{/literal}
</script>
</head>
<link rel="stylesheet" type="text/css" href="style.css"/>
<body bgcolor="#F2F2F2" leftmargin="0" topmargin="0" marginwidth="0" marginheight="0">
<br>

<script language="JavaScript" type="text/JavaScript">
{literal}

if (screen.width == 800){
document.write("<div id='usuario' style='position: absolute; top: 255; left: 1; text-align: right; vertical-
align: top' class='nome'>  Bem Vindo <font color='CC0000'>{/literal}{$login}{literal}</font>
 </div>");
}
else {
if (screen.width == 1024){
document.write("<div id='usuario' style='position: absolute; top: 255; left: 115; text-align: right; vertical-
align: top' class='nome'>  Bem Vindo <font color='CC0000'>{/literal}{$login}{literal}</font>
 </div>");
}
else {
if (screen.width == 1280){
document.write("<div id='usuario' style='position: absolute; top: 272; left: 250; text-align: right; vertical-
align: top' class='nome'>  Bem Vindo <font color='CC0000'>{/literal}{$login}{literal}</font>
 </div>");
}
}
}
```

```
{/literal}

 </script>

<table align="center" width="780" height="435" border="0" cellpadding="0" cellspacing="0">
          <tr>
                    <td colspan="3"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_01.jpg"
width="206" height="49"></td>
                    <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_02.jpg"
width="248" height="49"></td>
                    <td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_03.jpg" width="226"
height="49"></td>
      <td><a href="./sistema.php?modulo=usuario&action=usuario.close"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('sair','','{$smarty.const._IMAGEM_DIR_}sysconf_40.jpg',0)"><img
src="{$smarty.const._IMAGEM_DIR_}sysconf_04.jpg" name="sair" width="100" height="49"
border="0"></a></td>
          </tr>
          <tr>
                    <td colspan="3"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_05.jpg"
width="206" height="204"></td>
                    <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_06.jpg"
width="248" height="204"></td>
                    <td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_07.jpg" width="226"
height="204"></td>
                    <td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_08.jpg" width="100"
height="204"></td>
          </tr>
          <tr>
                    <td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_09.jpg" width="18"
height="46"></td>
      <td><a href="./sistema.php?modulo=chat&action=chat.load&cod=0"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('criachat','','{$smarty.const._IMAGEM_DIR_}sysconf_20.jpg',0)"><im
g src="{$smarty.const._IMAGEM_DIR_}sysconf_10.jpg" name="criachat" width="87" height="46"
border="0"></a></td>
                    <td><a href="./sistema.php?modulo=chat&action=chat.list"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('listachat','','{$smarty.const._IMAGEM_DIR_}sysconf_21.jpg',0)"><im
g src="{$smarty.const._IMAGEM_DIR_}sysconf_11.jpg" name="listachat" width="101" height="46"
border="0"></a></td>
                    <td><a href="./sistema.php?modulo=usuario&action=getpassword.load"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('trocasenha','','{$smarty.const._IMAGEM_DIR_}sysconf_22.jpg',0)">
<img src="{$smarty.const._IMAGEM_DIR_}sysconf_12.jpg" name="trocasenha" width="116"
height="46" border="0"></a></td>
                    <td><a href="./sistema.php?modulo=usuario&action=usuario.load"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('alteracadastro','','{$smarty.const._IMAGEM_DIR_}sysconf_23.jpg',0
)"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_13.jpg" name="alteracadastro" width="132"
height="46" border="0"></a></td>
                    <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_14.jpg"
width="326" height="46"></td>
          </tr>
     <!-- inicio conteudo -->
```

6.3. header_visitante.tpl

```
<html>
<head>
<title>WorldConf - O Mundo fala aqui!</title>
</head>
<link rel="stylesheet" type="text/css" href="style.css"/>
<body bgcolor="#F2F2F2" leftmargin="0" topmargin="0" marginwidth="0" marginheight="0">
<br>
<table align="center" width="780" height="436" border="0" cellpadding="0" cellspacing="0">
	<tr>
		<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_01.jpg" width="206"
height="49"></td>
		<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_02.jpg" width="248"
height="49"></td>
		<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_03.jpg" width="226"
height="49"></td>
	<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_41.jpg" width="100" height="49"
border="0"></a></td>
	</tr>
	<tr>
		<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_05.jpg" width="206"
height="204"></td>
		<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_06.jpg" width="248"
height="204"></td>
		<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_07.jpg" width="226"
height="204"></td>
		<td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_08.jpg" width="100"
height="204"></td>
	</tr>
	<tr>
		<td colspan="7"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_61.jpg"
width="780" height="46"></td>
	</tr>
    <!-- inicio conteudo -->
```

6.4. footer.tpl

```
        <tr>
                <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_17.jpg"
width="105" height="38"></td>
                <td colspan="4"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_18.jpg"
width="575" height="38"></td>
                <td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_19.jpg" width="100"
height="38"></td>
        </tr>
</table>
</body>
</html>
```

6.5. footer_admin.tpl

```
        <tr>
                <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_58.jpg"
width="105" height="38" alt=""></td>
                <td colspan="5"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_59.jpg"
width="543" height="38" alt=""></td>
                <td><img src="{$smarty.const._IMAGEM_DIR_}sysconf_60.jpg" width="132"
height="38" alt=""></td>
        </tr>
</table>
</body>
</html>
```

6.6. footer_vistante.tpl

```
   <tr>
                    <td colspan="7"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_76.jpg"
width="780" height="38"></td>
          </tr>
</table>
</body>
</html>
```

7. MÓDULO USUÁRIO – CONFIGURAÇÃO

7.1. config.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<config>

   <!-- CONSTANTES VALIDAS PARA ESTE MODULO -->

   <!-- constante obrigatoria: nome do modulo -->
   <constant name="MODULE_NAME" value="Módulo Usuário"/>

   <!-- ACTIONS VALIDAS PARA ESTE MODULO -->

   <!-- Usuário - Index -->
   <action name="usuario.index" class="UsuarioIndexAction">
      <result name="input" type="action">chat.list</result>
      <result name="success" type="smarty">usuario/index.tpl</result>
   </action>

   <!-- Usuário - Listar -->
   <action name="usuario.list" class="UsuarioListAction">
      <result name="success" type="smarty">usuario/list.tpl</result>
   </action>

   <!-- Usuário - Detalhes -->
   <action name="usuario.load" class="UsuarioLoadAction">
      <result name="success" type="smarty">usuario/load.tpl</result>
   </action>

   <!-- Usuário - Salvar -->
   <action name="usuario.save" class="UsuarioSaveAction">
      <result name="success" type="smarty">usuario/save.tpl</result>
      <result name="input" type="smarty">usuario/load.tpl</result>
   </action>

   <!-- Usuário - Excluir -->
   <action name="usuario.delete" class="UsuarioDeleteAction">
      <result name="success" type="action">usuario.list</result>
   </action>

   <!-- Usuário - Validar -->
   <action name="usuario.validate" class="UsuarioValidateAction">
      <result name="input" type="smarty">usuario/index.tpl</result>
      <result name="success" type="action">chat.list</result>
   </action>

   <!-- Usuário - Close-->
   <action name="usuario.close" class="UsuarioCloseAction">
      <result name="success" type="smarty">usuario/close.tpl</result>
   </action>

   <!-- Grupo - Listar -->
   <action name="grupo.list" class="GrupoListAction">
      <result name="success" type="smarty">grupo/list.tpl</result>
   </action>

   <!-- Grupo - Detalhes -->
   <action name="grupo.load" class="GrupoLoadAction">
```

```
        <result name="success" type="smarty">grupo/load.tpl</result>
    </action>

    <!-- Grupo - Salvar -->
    <action name="grupo.save" class="GrupoSaveAction">
        <result name="success" type="action">grupo.list</result>
        <result name="input" type="smarty">grupo/load.tpl</result>
    </action>

    <!-- Grupo - Excluir -->
    <action name="grupo.delete" class="GrupoDeleteAction">
        <result name="success" type="action">grupo.list</result>
    </action>

    <!-- GetPassword - Detalhes -->
    <action name="getpassword.load" class="GetPasswordLoadAction">
        <result name="success" type="smarty">getPassword/load.tpl</result>
    </action>

    <!-- GetPassword - Salvar -->
    <action name="getpassword.save" class="GetPasswordSaveAction">
        <result name="input" type="smarty">getPassword/load.tpl</result>
        <result name="success" type="smarty">getPassword/save.tpl</result>
    </action>

    <!-- EsqueceSenha - Detalhes -->
    <action name="esquecesenha.load" class="EsqueceSenhaLoadAction">
        <result name="success" type="smarty">esqueceSenha/load.tpl</result>
    </action>

    <!-- EsqueceSenha - Salvar -->
    <action name="esquecesenha.save" class="EsqueceSenhaSaveAction">
        <result name="input" type="smarty">esqueceSenha/load.tpl</result>
        <result name="success" type="smarty">esqueceSenha/mensagem.tpl</result>
    </action>-->

    <!-- AlteraStatus - Detalhes -->
    <action name="alterastatus.load" class="AlteraStatusLoadAction">
        <result name="success" type="smarty">alteraStatus/load.tpl</result>
    </action>

    <!-- AlteraStatus - Salvar -->
    <action name="alterastatus.save" class="AlteraStatusSaveAction">
        <result name="input" type="smarty">alteraStatus/load.tpl</result>
        <result name="success" type="action">usuario.list</result>
    </action>

</config>
```

8. MÓDULO USUÁRIO – CLASSES PACOTES ACTION

8.1. AlteraStatusLoadAction.class.php

```
<?php

   /**
    * Action responsável pela exibição do status do Usuário do sistema.
    *
    * @see Action
    * @package control
    */

class AlteraStatusLoadAction extends Action {

   /**
    * Efetua o tratamento das requisições. Pesquisa o STATUS do Usuário solicitado
    * de acordo com a solicitação.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
         $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
         <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
         <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
                  <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
         </tr></table></body></html>";
         echo $teste;
         exit();
      }

      $cod = $_REQUEST["cod"];

         if ((is_numeric($cod)) && ($cod > 0)) {
            $usuario = UsuarioManager::findByKey($cod);
         } else {
            $usuario = new Usuario();
            $usuario->setCod($cod);
            $usuario->setStatus($_REQUEST["status"]);
         }

      $output = array();
      $output["usuario"] = $usuario;
      $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      return $output;
   }
}
?>
```

8.2. AlteraStatusSaveAction.class.php

```
<?php
   /**
    * Action responsável pela atualização do Status do Usuários no sistema.
    * @see Action
    * @package control
    */
class AlteraStatusSaveAction extends Action {

   /**
    * Efetua o tratamento das requisições. Realiza a atualização do STATUS do Usuário solicitado.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
         $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
         <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
         <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
                  <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
         </tr></table></body></html>";
         echo $teste;
         exit();
      }

      $output = array();
      $usuario = new Usuario();
      $usuario->setCod($_REQUEST["cod"]);
      $usuario->setStatus($_REQUEST["status"]);
      $usuario->setNome($_REQUEST["nome"]);

      $errors = $this->validate($usuario);

      if (count($errors) > 0) {
         $output["usuario"] =  $usuario;
         $output[PARAM_ERRORS] = $errors;
         $output[PARAM_ACTION_RESULT] = Action::INPUT;
      }
      else {
         $key = $usuario->getCod();
         AlteraStatusManager::save($usuario);
         $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      }
      $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      return $output;
   }
}
?>
```

8.3. EsqueceSenhaLoadAction.class.php

```
<?php

    /**
     * Action responsável pela exibição do formulário esquece senha.
     *
     * @see Action
     * @package control
     */
class EsqueceSenhaLoadAction extends Action {

    /**
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();

        $output = array();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }
}
?>
```

8.4. EsqueceSenhaSaveAction.class.php

```
<?php

    /** Action responsável pelo envio de uma nova senha para o usuário.
     * Ela irá receber o email do usuário, irá verificar se este email existe e após a validação.
     * atualizará a senha do usuário no Banco de Dados e enviará a nova senha para o Usuário
     *
     * @see Action
     * @package control
     */
class EsqueceSenhaSaveAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa o email do Usuário solicitado e
     * atualiza a Senha do Usuario solicitado.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     * Caso a validação das informações do Usuario falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
     */
    public function execute() {

        session_start();

        $output = array();
        $esqueceSenha = new Usuario();
        $esqueceSenha->setEmail($_REQUEST["email"]);

        $arrUsuario = EsqueceSenhaManager::findByField($esqueceSenha->getEmail());

        if (count($arrUsuario) == 1) {
            $objUsuario = $arrUsuario[0];

            $novaSenha = substr(md5(date("H:i:s")),7,6);

            $objUsuario->setSenha(md5($novaSenha));

            $objUsuario = EsqueceSenhaManager::save($objUsuario);

            $assunto = _MAIL_ASSUNTO_ . " Recuperação de Senha";
            $corpoMsg = "Sr(a):".  $objUsuario->getNome()."\nLogin:".$objUsuario->getLogin()."\nSenha:
".$novaSenha;

            Util::mailSender($objUsuario->getEmail(), $assunto, $corpoMsg);

            $output["msgEmailEnviado"] = "Em instantes você receberá um email com uma nova senha!";
            $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

        }
        elseif ($esqueceSenha->getEmail() == "") {
            $errors[] = "É necessário preencher o campo Email!";
            $output[PARAM_ERRORS] = $errors;
            $output[PARAM_ACTION_RESULT] = Action::INPUT;
        }
        elseif (count($arrUsuario) == 0){
            $errors[] = "Email inexistente!";
            $output[PARAM_ERRORS] = $errors;
            $output[PARAM_ACTION_RESULT] = Action::INPUT;
```

```
        }
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }
}
?>
```

8.5. GetPasswordLoadAction.class.php

```
<?php

   /**
    * Action responsável pela exibição do formulário de troca de Senha.
    *
    * @see Action
    * @package control
    */

class GetPasswordLoadAction extends Action {

   /**
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] == ""){

         $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
         <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
         <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN!</td></tr>"."
               <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
         </tr></table></body></html>";
         echo $teste;
         exit();
      }

      $output = array();
      $loginUsuario = $_SESSION["loginUsuario"];
      $output["login"] = $loginUsuario;
      $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      return $output;
   }
}
?>
```

8.6. GetPasswordSaveAction.class.php

```
<?php

   /**
    * Action responsável pela alteração da senha do Usuario no sistema.
    * @see Action
    * @package control
    */
class GetPasswordSaveAction extends Action {

   /**
    * Efetua o tratamento das requisições. Pesquisa a senha antiga do Usuário solicitado.
    * Esta senha será alterada e atualizada no Banco de Dados.
    * Realiza a atualização da senha do Usuario solicitado.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    * Caso a validação das informações do Usuario falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] == ""){
        $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
        <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
        <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN!</td></tr>"."
              <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
        </tr></table></body></html>";
        echo $teste;
        exit();
      }

      $output = array();
      $getpassword = new GetPassword();
      $getpassword->setCod($_SESSION["codUsuario"]);
      $getpassword->setSenha($_REQUEST["senha"]);
      $getpassword->setConfirmaSenha($_REQUEST["confirma_senha"]);
      $getpassword->setSenhaAntiga($_REQUEST["senhaAntiga"]);

      $errors = $this->validate($getpassword);

      if (count($errors) > 0) {
         $output["getpassword"] =  $getpassword;
         $output[PARAM_ERRORS] = $errors;
         $output[PARAM_ACTION_RESULT] = Action::INPUT;
      }
      else {
         $key = $getpassword->getCod();
         $senhaNova = GetPasswordManager::FindByCod($key);
         $senha = $senhaNova->getSenha();
         $senhaAntiga = md5($getpassword->getSenhaAntiga());

         /* Compara se a senha digitada no campo SENHA é a mesma que se encontra no
         Banco de Dados.*/
         if (md5($senha) == md5($senhaAntiga)){
```

```
                GetPasswordManager::save($getpassword);
                $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
            }
            else {
                $output["getpassword"] =  $getpassword;
                $output[PARAM_ERRORS] = "Senha Antiga Não Confere";
                $output[PARAM_ACTION_RESULT] = Action::INPUT;
             }
        }
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }

    /**
     * Efetua a validação dos campos do formulário
     * @see Action::validate()
     * @param object $getpassword que será validado
     * @return array $errors
     */
    public function validate($getpassword) {

        $errors = array();

        // Valida se a senha digitada possui no mínimo 6 caracteres
        if (strlen($getpassword->getSenha()) < 6) {
           $errors[] = "Senha deve ter no mínimo 6 caracteres!";
        }

        // Valida se o campo CONFIRMA SENHA não está em branco
        if (($getpassword->getConfirmaSenha() == "")) {
            $errors[] = "Confirmação de Senha Inválida!";
        }

        // Valida se a senha digitada no campo SENHA esta igual à do campo CONFIRMA SENHA
        if (($getpassword->getSenha() != $getpassword->getConfirmaSenha())) {
            $errors[] = "Senhas não conferem!";
        }

        return $errors;
    }
}
?>
```

8.7. GrupoDeleteAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela exclusão de Grupos do sistema.
 *
 * @see Action
 * @package control
 */
class GrupoDeleteAction extends Action {

    /**
     * Efetua o tratamento das requisições. Exclui os dados do Grupo solicitado através do cod.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $cod = $_REQUEST["cod"];

        GrupoManager::delete($cod);

        $output = array();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }

}

?>
```

8.8. GrupoListAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela obtenção da lista de Usuários do sistema.
 *
 * @see Action
 * @package control
 */
class GrupoListAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa a lista de todos os Grupos do sistema.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
        $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf - O
Mundo fala aqui!</title></head>"."
            <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
            <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
                    <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
            </tr></table></body></html>";
            echo $teste;
            exit();
        }

        $output = array();
        $output["lista"] = GrupoManager::fetchAll();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }

}

?>
```

8.9. GrupoLoadAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela exibição dos detalhes de Grupos do sistema.
 *
 * @see Action
 * @package control
 */
class GrupoLoadAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa os dados do Grupos solicitado ou então
     * cria um novo Grupo, de acordo com a solicitação. Estes dados podem ser alterados e
     * posteriormente salvos com a action de inserção/atualização.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
            $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
            <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
            <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
                    <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
            </tr></table></body></html>";
            echo $teste;
            exit();
        }

        $cod = $_REQUEST["cod"];

        if ((is_numeric($cod)) && ($cod > 0)) {
            $grupo = GrupoManager::findByKey($cod);
        } else {
            $grupo = new Grupo();
            $grupo->setCod($cod);
        }

        $output = array();
        $output["grupo"] =  $grupo;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }

}

?>
```

8.10. GrupoSaveAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela inserção e atualização de Grupos do sistema.
 *
 * @see Action
 * @package control
 */
class GrupoSaveAction extends Action {

   /**
    * Efetua o tratamento das requisições. Realiza a inserção/atualização do Grupo solicitado.
    * Caso a validação das informações do Grupo falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
         $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
         <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
         <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
                  <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
         </tr></table></body></html>";
         echo $teste;
         exit();
      }

      $output = array();

      $grupo = new Grupo();
      $grupo->setCod($_REQUEST["cod"]);
      $grupo->setNome(ltrim($_REQUEST["nome"]));
      $grupo->setSigla(ltrim($_REQUEST["sigla"]));
      $grupo->setDescricao(ltrim($_REQUEST["descricao"]));

      $errors = $this->validate($grupo);

      if (count($errors) > 0) {
         $output["grupo"] =  $grupo;
         $output[PARAM_ERRORS] = $errors;
         $output[PARAM_ACTION_RESULT] = Action::INPUT;

      }
      else {
         GrupoManager::save($grupo);
         $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
```

```
        }
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }

     /**
     * Efetua a validação dos campos do formulário
     * @see Action::validate()
     * @param object $grupo que será validado
     * @return array $errors
     */
    public function validate($grupo) {
        $errors = array();

        if ($grupo->getNome() == "") {
            $errors[] = "O nome deve ser informado.";
        }
        if ($grupo->getSigla() == "") {
            $errors[] = "A sigla deve ser informada.";
        }
        if (($grupo->getDescricao() == "")) {
            $errors[] = "A descrição deve ser informada.";
        }

        return $errors;
    }

}

?>
```

8.11. UsuarioCloseAction.class.php

```
<?php

   /**
    * Action responsável pelo fechamento da Sessão do Usuário.
    *
    * @see Action
    * @package control
    */
class UsuarioCloseAction extends Action {

   /**
    * Efetua o tratamento das requisições. Fecha e Destrói a Sessão do Usuário.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      session_unset();
      session_destroy();

      $output = array();
      $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      return $output;
   }
}
?>
```

8.12. UsuarioDeleteAction.class.php

```
<?php

   /**
    * Action responsável pela exclusão de Usuários do sistema.
    *
    * @see Action
    * @package control
    */
class UsuarioDeleteAction extends Action {

   /**
    * Efetua o tratamento das requisições. Exclui os dados do Usuário solicitado através do Cod.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
        $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
        <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
        <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
              <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
        </tr></table></body></html>";
        echo $teste;
        exit();
      }
      $cod = $_REQUEST["cod"];

      UsuarioManager::delete($cod);

      $output = array();
      $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      return $output;
   }
}
?>
```

8.13. UsuarioIndexAction.class.php

```
<?php

   /**
    * Action responsável pela exibição da Tela Inicial.
    *
    * @see Action
    * @package control
    */
class UsuarioIndexAction extends Action {

   /**
    * Efetua o tratamento das requisições.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] != ""){
         $output = array();
         $output[PARAM_ACTION_RESULT] = Action::INPUT;
         return $output;
      }

      $_SESSION['opcao'] = 1;

      $output = array();
      $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      return $output;
   }
}
?>
```

8.14. UsuarioListAction.class.php

```
<?php

   /**
    * Action responsável pela obtenção da lista de Usuários do sistema.
    *
    * @see Action
    * @package control
    */
class UsuarioListAction extends Action {

   /**
    * Efetua o tratamento das requisições. Pesquisa a lista de todos os Usuários do sistema.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
        $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
        <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
        <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
              <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
        </tr></table></body></html>";
        echo $teste;
        exit();
      }

      $array1 = array();
      $array2 = array();

      $array1 = UsuarioManager::fetchAll();

      for($i = 0; $i < sizeof($array1); $i++){
            $usuario = $array1[$i];
            $usuario->setLogin(wordwrap($usuario->getLogin(),10,"<br>",1));
            $usuario->setEmail(wordwrap($usuario->getEmail(),30,"<br>",1));
            $usuario->setNome(wordwrap($usuario->getNome(),30,"<br>",1));
            $usuario->setInstituicao(wordwrap($usuario->getInstituicao(),25,"<br>",1));
            $usuario->setStatus(wordwrap($usuario->getStatus(),6,"<br>",1));
            $array2[] = $usuario;
      }

      $output["lista"] = $array2;
      $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      return $output;
   }
}
?>
```

8.15. UsuarioLoadAction.class.php

```
<?php

   /**
    * Action responsável pela exibição dos detalhes de Usuários do sistema.
    *
    * @see Action
    * @package control
    */
class UsuarioLoadAction extends Action {

   /**
    * Efetua o tratamento das requisições. Pesquisa os dados do Usuário solicitado ou então
    * cria um novo Usuário, de acordo com a solicitação. Estes dados podem ser alterados e
    * posteriormente salvos com a action de inserção/atualização.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if (($_SESSION["loginUsuario"] == "") AND ($_REQUEST["cod"] != 0)){

      $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf - O
Mundo fala aqui!</title></head>"."
        <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
        <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN!</td></tr>"."
              <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
        </tr></table></body></html>";
        echo $teste;
        exit();
       }

      if($_SESSION["loginUsuario"] == ""){
        $cod = 0;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      }
      elseif (($_SESSION["loginUsuario"] != "") AND ($_REQUEST["cod"] == 0)) {
        $cod = $_SESSION["codUsuario"];
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      }
      else {
        $cod = $_REQUEST["cod"];
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      }

      if ($cod > 0) {
         $usuario = UsuarioManager::findByKey($cod);
      }
      else {
         $usuario = new Usuario();
         $usuario->setCod($cod);
      }
      $output = array();
      $output["usuario"] =  $usuario;
      $output["opcao"] = $_SESSION["opcao"];
```

```
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }
}
?>
```

8.16. UsuarioSaveAction.class.php

```
<?php

   /**
    * Action responsável pela inserção e atualização de Usuarios do sistema.
    * @see Action
    * @package control
    */
class UsuarioSaveAction extends Action {

   /**
    * Efetua o tratamento das requisições. Realiza a inserção/atualização do Usuario solicitado.
    * Caso a validação das informações do Usuario falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();

      $output = array();
      $usuario = new Usuario();
      $usuario->setCod($_REQUEST["cod"]);
      $usuario->setLogin(ltrim($_REQUEST["login"]));
      $usuario->setSenha(ltrim($_REQUEST["senha"]));
      $usuario->setConfirmaSenha(ltrim(($_REQUEST["confirma_senha"])));
      $usuario->setNome(ltrim($_REQUEST["nome"]));
      $usuario->setRg($_REQUEST["rg"]);
      $usuario->setCpf($_REQUEST["cpf"]);
      $usuario->setDataNascimento($_REQUEST["data_nascimento"]);
      $usuario->setTelefone($_REQUEST["telefone"]);
      $usuario->setEmail($_REQUEST["email"]);
      $usuario->setEndereco(ltrim($_REQUEST["endereco"]));
      $usuario->setSexo($_REQUEST["sexo"]);
      $usuario->setCidade($_REQUEST["cidade"]);
      $usuario->setUf($_REQUEST["uf"]);
      $usuario->setCep($_REQUEST["cep"]);
      $usuario->setPais($_REQUEST["pais"]);
      $usuario->setInstituicao(ltrim($_REQUEST["instituicao"]));

      $errors = $this->validate($usuario);

      if (count($errors) > 0) {
         $output["usuario"] =  $usuario;
         $output[PARAM_ERRORS] = $errors;
         $output[PARAM_ACTION_RESULT] = Action::INPUT;
      }
      else {
         UsuarioManager::save($usuario);
         $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
          }
      $output["opcao"] = $_SESSION["opcao"];
      $loginUsuario = $_SESSION["loginUsuario"];
      $output["login"] = $loginUsuario;
      $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      return $output;
   }
```

```
    /**
     * Efetua a validação dos campos do formulário
     * @see Action::validate()
     * @param object $usuario que será validado
     * @return array $errors
     */
    public function validate($usuario) {
        $errors = array();

        /* Transformando o campo Login para letras minúsculas */
        $usuario->setLogin(strtolower($usuario->getLogin()));

        /* Transformando o campo Email para letras minúsculas */
        $usuario->setEmail(strtolower($usuario->getEmail()));

        // Atualização, a única excessão é a validação do email, para que não seja um email repetido. O
resto se mantem como na inserção.
        if ($usuario->getCod() > 0) {

            $email = $usuario->getEmail();
            if (UsuarioManager::findByEmail($email, $usuario->getCod())){
                $errors[] = "Este Email já existe! Tente outro.";
            }
        }
        // Inserção, a única excessão é a validação do login, para que não seja um login repetido.
        else {
            $login=$usuario->getLogin();
            if (UsuarioManager::findByLogin($login)){
                $errors[] = "Este Login já existe! Tente outro.";
            }

            if (strlen($usuario->getLogin()) < 3) {
                $errors[] = "Login deve ter no mínimo 3 caracteres!";
            }

            if (strlen($usuario->getSenha()) < 6) {
                $errors[] = "Senha deve ter no mínimo 6 caracteres!";
            }

            if (($usuario->getConfirmaSenha() == "")) {
                $errors[] = "Confirmação de Senha Inválida!";
            }

            if (($usuario->getSenha() != $usuario->getConfirmaSenha())) {
                $errors[] = "Senhas não conferem!";
            }

            $email = $usuario->getEmail();
            if (UsuarioManager::findByEmail($email, $usuario->getCod())){
                $errors[] = "Este Email já existe! Tente outro.";
            }
        }

        if ($usuario->getNome() == "") {
                $errors[] = "Um Nome deve ser Informado.";
        }

        if (Util::validaCpf($usuario->getCpf())) {
            $errors[] = "CPF inválido!";
```

```
        }

        if (Util::validaData($usuario->getDataNascimento())) {
            $errors[] = "Data inválida!";
        }

        if (Util::validaEmail($usuario->getEmail())) {
            $errors[] = "Email inválido!";
        }

        if (($usuario->getEndereco() == "")) {
            $errors[] = "Um Endereco deve ser Informado.";
        }

        if (($usuario->getInstituicao() == "")) {
            $errors[] = "Uma Instituição deve ser Informada.";
        }

        return $errors;
    }
}
?>
```

8.17. UsuarioValidateAction.class.php

```
<?php

   /**
    * Action responsável pela verificação do Login dos Usuários do sistema.
    * @see Action
    * @package control
    */
class UsuarioValidateAction extends Action {

   /**
    * Efetua o tratamento das requisições. Realiza a verificação do Login do Usuário solicitado.
    * Caso a validação das informações do Usuario falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      $output = array();

      session_start();
      $numVars = sizeof($_SESSION);
      $login = $_REQUEST["login"];
      $senha = md5($_REQUEST["senha"]);
      $status = TRUE;

      if(($numVars > 0) AND ($login == $_SESSION["loginUsuario"]) AND ($senha ==
$_SESSION["senha"])){
         $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
         $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      } else {
         $usuario = new Usuario();
         $usuario->setLogin($login);
         $usuario->setSenha($senha);
         $usuario->setStatus($status);

         $user = UsuarioManager::findByLoginSenha($usuario->getLogin(), $usuario->getSenha(),
$usuario->getStatus());

         $errors = $this->validate($user->getCod());

         if (count($errors) == 0) {
            $_SESSION["codUsuario"] = $user->getCod();
            $_SESSION["loginUsuario"] = $login;
            $_SESSION["senhaUsuario"] = $senha;

            if($user->getCod() != 1){
               $_SESSION["opcao"] = 2;
            }
            else{
               $_SESSION["opcao"] = 3;
            }
            $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
            $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
         } else {
            $output[PARAM_ERRORS] = $errors;
            $output[PARAM_ACTION_RESULT] = Action::INPUT;
```

```
            }
        }
        return $output;
    }

    /**
     * Efetua a validação dos campos do formulário
     * @see Action::validate()
     * @param mixed $codUsuario que será validado
     * @return array $errors
     */
    public function validate($codUsuario) {
        $errors = array();

        if ($codUsuario == "") {
            $errors[] = "Usuário ou Senha Inválidos";
        }

        return $errors;
    }

}
?>
```

9. MÓDULO USUÁRIO – CLASSES PACOTE MODEL

9.1. GetPassword.class.php

```
<?php

/**
 * Classe que define uma Senha para o Usuário no sistema.
 *
 * @package model
 */
class GetPassword extends Usuario {

    private $senhaAntiga;

    /**
     * Retorna o valor do campo <kbd>senha</kbd>.
     *
     * @return Retorna o valor do campo <kbd>senha</kbd>.
     */
    public function getSenhaAntiga() {
        return $this->senhaAntiga;
    }

    /**
     * Armazena o valor do campo <kbd>senha</kbd>.
     *
     * @param senha O valor a ser armazenado.
     */
    public function setSenhaAntiga($senhaAntiga) {
        $this->senhaAntiga = $senhaAntiga;
    }
}
?>
```

9.2. Grupo.class.php

```
<?php

/**
 * Classe que define um Grupo no sistema.
 *
 * @package model
 */
class Grupo {

    /** Código do Grupo. */
    private $cod;

    /** Nome do Grupo. */
    private $nome;

    /** Sigla do Grupo. */
    private $sigla;

    /** Descrição do Grupo. */
    private $descricao;

    /**
     * Retorna o valor do campo <kbd>cod</kbd>.
     *
     * @return Retorna o valor do campo <kbd>cod</kbd>.
     */
    public function getCod() {
        return $this->cod;
    }

    /**
     * Armazena o valor do campo <kbd>cod</kbd>.
     *
     * @param cod O valor a ser armazenado.
     */
    public function setCod($cod) {
        $this->cod = $cod;
    }

    /**
     * Retorna o valor do campo <kbd>nome</kbd>.
     *
     * @return Retorna o valor do campo <kbd>nome</kbd>.
     */
    public function getNome() {
        return $this->nome;
    }

    /**
     * Armazena o valor do campo <kbd>nome</kbd>.
     *
     * @param nome O valor a ser armazenado.
     */
    public function setNome($nome) {
        $this->nome = $nome;
    }

    /**
```

```
     * Armazena o valor do campo <kbd>sigla</kbd>.
     *
     * @param sigla O valor a ser armazenado.
     */
    public function setSigla($sigla) {
        $this->sigla = $sigla;
    }

    /**
     * Retorna o valor do campo <kbd>sigla</kbd>.
     *
     * @return Retorna o valor do campo <kbd>sigla</kbd>.
     */
    public function getSigla() {
        return $this->sigla;
    }

    /**
     * Armazena o valor do campo <kbd>descricao</kbd>.
     *
     * @param descricao O valor a ser armazenado.
     */
    public function setDescricao($descricao) {
        $this->descricao = $descricao;
    }

    /**
     * Retorna o valor do campo <kbd>descricao</kbd>.
     *
     * @return Retorna o valor do campo <kbd>descricao</kbd>.
     */
    public function getDescricao() {
        return $this->descricao;
    }

}
?>
```

9.3. Usuario.class.php

```
<?php

    /**
     * Classe que define um Usuário no sistema.
     *
     * @package model
     */

class Usuario {

    /** Codigo do Usuario. */
    private $cod;

    /** Login do Usuario. */
    private $login;

    /** Senha do Usuario. */
    private $senha;

    /** Confirma Senha do Usuario. */
    private $confirmaSenha;

    /** Nome do Usuario. */
    private $nome;

    /** RG do Usuario. */
    private $rg;

    /** CPF do Usuario. */
    private $cpf;

    /** Data de Nascimento do Usuario. */
    private $dataNascimento;

    /** Telefone do Usuario. */
    private $telefone;

    /** Email do Usuario. */
    private $email;

    /** Endereço do Usuario. */
    private $endereco;

    /** Sexo do Usuario. */
    private $sexo;

    /** Cidade do Usuario. */
    private $cidade;

    /** Unidade Federativa (UF) do Usuario. */
    private $uf;

    /** CEP do Usuario. */
    private $cep;

    /** País do Usuario. */
    private $pais;
```

```
/** Insttituição do Usuario. */
private $instituicao;

/** Status do Usuario. */
private $status;

/** Status do Usuario. */
private $dataCriacao;

/** Status do Usuario. */
private $validacaoUsuario;

/**
 * Retorna o valor do campo <kbd>cod</kbd>.
 *
 * @return Retorna o valor do campo <kbd>cod</kbd>.
 */
public function getCod() {
    return $this->cod;
}

/**
 * Armazena o valor do campo <kbd>cod</kbd>.
 *
 * @param cod O valor a ser armazenado.
 */
public function setCod($cod) {
    $this->cod = $cod;
}

/**
 * Retorna o valor do campo <kbd>login</kbd>.
 *
 * @return Retorna o valor do campo <kbd>login</kbd>.
 */
public function getLogin() {
    return $this->login;
}

/**
 * Armazena o valor do campo <kbd>login</kbd>.
 *
 * @param Login O valor a ser armazenado.
 */
public function setLogin($login) {
   $this->login = $login;
}

/**
 * Retorna o valor do campo <kbd>senha</kbd>.
 *
 * @return Retorna o valor do campo <kbd>senha</kbd>.
 */
public function getSenha() {
    return $this->senha;
}

/**
 * Armazena o valor do campo <kbd>senha</kbd>.
 *
```

```
 * @param senha O valor a ser armazenado.
 */
public function setSenha($senha) {
    $this->senha = $senha;
}

/**
 * Retorna o valor do campo <kbd>confirmaSenha</kbd>.
 *
 * @return Retorna o valor do campo <kbd>confirmaSenha</kbd>.
 */
public function getConfirmaSenha() {
    return $this->confirmaSenha;
}

/**
 * Armazena o valor do campo <kbd>confirmaSenha</kbd>.
 *
 * @param confirmaSenha O valor a ser armazenado.
 */
public function setConfirmaSenha($confirmaSenha) {
    $this->confirmaSenha = $confirmaSenha;

/**
 * Retorna o valor do campo <kbd>nome</kbd>.
 *
 * @return Retorna o valor do campo <kbd>nome</kbd>.
 */
 }
public function getNome() {
    return $this->nome;
}

/**<td width="30%"><b>Nome</b></td>
 * Armazena o valor do campo <kbd>nome</kbd>.
 *
 * @param nome O valor a ser armazenado.
 */
public function setNome($nome) {
    $this->nome = $nome;
}

/**
 * Retorna o valor do campo <kbd>rg</kbd>.
 *
 * @return Retorna o valor do campo <kbd>rg</kbd>.
 */
public function getRg() {
    return $this->rg;
}

/**
 * Armazena o valor do campo <kbd>rg</kbd>.
 *
 * @param rg O valor a ser armazenado.
 */
public function setRg($rg) {
    $this->rg = $rg;
}
```

```
/**
 * Retorna o valor do campo <kbd>cpf</kbd>.
 *
 * @return Retorna o valor do campo <kbd>cpf</kbd>.
 */
public function getCpf() {
    return $this->cpf;
}

/**
 * Armazena o valor do campo <kbd>cpf</kbd>.
 *
 * @param cpf O valor a ser armazenado.
 */
public function setCpf($cpf) {
    $this->cpf = $cpf;
}

/**
 * Retorna o valor do campo <kbd>dataNascimento</kbd>.
 *
 * @return Retorna o valor do campo <kbd>dataNascimento</kbd>.
 */
public function getDataNascimento() {
    return $this->dataNascimento;
}

/**
 * Armazena o valor do campo <kbd>dataNascimento</kbd>.
 *
 * @param dataNascimento O valor a ser armazenado.
 */
public function setDataNascimento($dataNascimento) {
    $this->dataNascimento = $dataNascimento;
}

/**
 * Retorna o valor do campo <kbd>telefone</kbd>.
 *
 * @return Retorna o valor do campo <kbd>telefone</kbd>.
 */
public function getTelefone() {
    return $this->telefone;
 }

/**
 * Armazena o valor do campo <kbd>telefone</kbd>.
 *
 * @param telefone O valor a ser armazenado.
 */
public function setTelefone($telefone) {
    $this->telefone = $telefone;
}


/**
 * Retorna o valor do campo <kbd>email</kbd>.
 *
 * @return Retorna o valor do campo <kbd>email</kbd>.
 */
```

```
public function getEmail () {
    return $this->email;
}

/**
 * Armazena o valor do campo <kbd>email</kbd>.
 *
 * @param email O valor a ser armazenado.
 */
public function setEmail($email) {
    $this->email = $email;
}

/**
 * Retorna o valor do campo <kbd>endereco</kbd>.
 *
 * @return Retorna o valor do campo <kbd>endereco</kbd>.
 */
public function getEndereco () {
    return $this->endereco;
}

/**
 * Armazena o valor do campo <kbd>endereco</kbd>.
 *
 * @param endereco O valor a ser armazenado.
 */
public function setEndereco($endereco) {
    $this->endereco = $endereco;
}

/**
 * Retorna o valor do campo <kbd>sexo</kbd>.
 *
 * @return Retorna o valor do campo <kbd>sexo</kbd>.
 */
public function getSexo () {
    return $this->sexo;
   }

/**
 * Armazena o valor do campo <kbd>sexo</kbd>.
 *
 * @param sexo O valor a ser armazenado.
 */
public function setSexo($sexo) {
    $this->sexo = $sexo;
}

/**
 * Retorna o valor do campo <kbd>cidade</kbd>.
 *
 * @return Retorna o valor do campo <kbd>cidade</kbd>.
 */
public function getCidade () {
    return $this->cidade;
   }

/**
 * Armazena o valor do campo <kbd>cidade</kbd>.
```

```
 *
 * @param cidade valor a ser armazenado.
 */
public function setCidade($cidade) {
    $this->cidade = $cidade;
}

/**
 * Retorna o valor do campo <kbd>uf</kbd>.
 *
 * @return Retorna o valor do campo <kbd>uf</kbd>.
 */
public function getUf () {
    return $this->uf;
   }

/**
 * Armazena o valor do campo <kbd>uf</kbd>.
 *
 * @param uf valor a ser armazenado.
 */
public function setUf($uf) {
    $this->uf = $uf;
}

 /**
 * Retorna o valor do campo <kbd>cep</kbd>.
 *
 * @return Retorna o valor do campo <kbd>cep</kbd>.
 */
public function getCep () {
    return $this->cep;
   }

/**
 * Armazena o valor do campo <kbd>cep</kbd>.
 *
 * @param cep valor a ser armazenado.
 */
public function setCep($cep) {
    $this->cep = $cep;
}

/**
 * Retorna o valor do campo <kbd>pais</kbd>.
 *
 * @return Retorna o valor do campo <kbd>pais</kbd>.
 */
public function getPais () {
    return $this->pais;
   }

/**
 * Armazena o valor do campo <kbd>pais</kbd>.
 *
 * @param pais valor a ser armazenado.
 */
public function setPais($pais) {
    $this->pais = $pais;
}
```

```
/**
 * Retorna o valor do campo <kbd>instituicao</kbd>.
 *
 * @return Retorna o valor do campo <kbd>instituicao</kbd>.
 */
public function getInstituicao () {
    return $this->instituicao;
   }

/**
 * Armazena o valor do campo <kbd>instituicao</kbd>.
 *
 * @param instituicao valor a ser armazenado.
 */
public function setInstituicao($instituicao) {
    $this->instituicao = $instituicao;
}

/**
 * Retorna o valor do campo <kbd>status</kbd>.
 *
 * @return Retorna o valor do campo <kbd>status</kbd>.
 */
public function getStatus () {
    return $this->status;
   }

/**
 * Armazena o valor do campo <kbd>status</kbd>.
 *
 * @param status valor a ser armazenado.
 */
public function setStatus($status) {
    $this->status = $status;
}


/**
 * Retorna o valor do campo <kbd>dataCriacao</kbd>.
 *
 * @return Retorna o valor do campo <kbd>dataCriacao</kbd>.
 */
public function getDataCriacao () {
    return $this->dataCriacao;
   }

/**
 * Armazena o valor do campo <kbd>dataCriacao</kbd>.
 *
 * @param dataCriacao valor a ser armazenado.
 */
public function setDataCriacao($dataCriacao) {
    $this->dataCriacao = $dataCriacao;
}

/**
 * Retorna o valor do campo <kbd>validacaoUsuario</kbd>.
 *
 * @return Retorna o valor do campo <kbd>validacaoUsuario</kbd>.
```

```
     */
    public function getValidacaoUsuario () {
        return $this->validacaoUsuario;
      }

    /**
     * Armazena o valor do campo <kbd>validacaoUsuario</kbd>.
     *
     * @param validacaoUsuario valor a ser armazenado.
     */
    public function setValidacaoUsuario($validacaoUsuario) {
        $this->validacaoUsuario = $validacaoUsuario;
    }
}
?>
```

10. MÓDULO USUÁRIO – CLASSES MODEL / PERSISTENCE

10.1. AlteraStatusManager.class.php

```
<?php


   /**
    * Classe que define o gerenciador de persistência para a classe <kbd>AlteraStatus</kbd>.
    *
    * @static
    * @see AlteraStatusPersistenceManager
    * @package model
    */

class AlteraStatusManager implements AlteraStatusPersistenceManager {

   /**
    * @see AlteraStatusPersistenceManager::save()
    */
   public static function save($usuario) {
      $pdo = Util::getPDOHandler();

      $sql = "UPDATE usuario SET status_usuario = :status WHERE cod_usuario = :cod";

      $stmt = $pdo->prepare($sql);
      /* A pesquisa é realizada pelo codUsuario e o status e a validacao são atualizados.
        o status recebe ATIVO ou INATIVO e a validacao recebe a data atual.*/
      $stmt->execute(array(
         ":cod" => $usuario->getCod(),
         ":status" => $usuario->getStatus())
      );

      $infoUsuario = UsuarioManager::findByKey($usuario->getCod());


      if ($infoUsuario->getStatus() == 0 ) {
         $texto = "seu cadastro do sistema foi DESATIVADO pelo Administrador.";
         $assunto = "Desativação de Cadastro";
         $texto2 = "Para ativar seu cadastro novamente entre em contato com o Administrador\n\n".
_MAIL_FROM_;
      }

      else {
         $texto = "seu cadastro foi aceito pelo Administrador.";
         $assunto = "Ativação de Cadastro";
         $texto2 = "Parabéns, seja Bem Vindo ao WorldConf.";
      }
      $assunto = _MAIL_ASSUNTO_ ." ". $assunto;
      $corpoMsg =  "\nSr(a): ".  $infoUsuario->getNome().", ". $texto ."\nSeu login é: ".$infoUsuario-
>getLogin()
      ."\n\n". $texto2;

      Util::mailSender($infoUsuario->getEmail(), $assunto, $corpoMsg);

      return $usuario;
    }
}
?>
```

10.2. AlteraStatusPersistenceManager.class.php

```
<?php

   /**
    * Interface que define um Manager, classe responsável pelo gerenciamento
    * de persistência de alguma classe do Model. Todos os Managers deste módulo
    * devem implementar no mínimo os métodos definidos por este contrato.
    *
    *
    * @static
    * @package model
    */
interface AlteraStatusPersistenceManager {

   /**
    * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica
da operação,
    * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser
especificada na
    * implementação do método pela classe concreta. O exemplo mais comum para chave primária
inteira é assumir
    * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser
inserida)
    * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas
atualizada).
    *
    * @param mixed $model Objeto a ser gravado.
    */
   public static function save($model);
}
?>
```

10.3. EsqueceSenhaManager.class.php

```
<?php

    /**
     * Classe que define o gerenciador de persistência para a classe <kbd>EsqueceSenha</kbd>.
     *
     * @static
     * @see EsqueceSenhaPersistenceManager
     * @package model
     */

class EsqueceSenhaManager implements EsqueceSenhaPersistenceManager {

    /**
     * @see EsqueceSenhaPersistenceManager::findByField($campo, $dado)
     */
    public static function findByField($dado) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT * FROM usuario WHERE email_usuario = :dado");
        $stmt->bindParam(":dado", $dado);
        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {
            $c = new Usuario();
            $c->setCod($row["cod_usuario"]);
            $c->setLogin($row["login_usuario"]);
            $c->setNome($row["nome_usuario"]);
            $c->setSenha($row["senha_usuario"]);
            $c->setConfirmaSenha($row["senha_usuario"]);
            $c->setRg($row["rg_usuario"]);
            $c->setCpf($row["cpf_usuario"]);
            $c->setDataNascimento($row["dt_nascimento_usuario"]);
            $c->setTelefone($row["telefone_usuario"]);
            $c->setEndereco($row["endereco_usuario"]);
            $c->setEmail($row["email_usuario"]);
            $c->setSexo($row["sexo_usuario"]);
            $c->setCidade($row["cidade_usuario"]);
            $c->setUf($row["uf_usuario"]);
            $c->setCep($row["cep_usuario"]);
            $c->setPais($row["pais_usuario"]);
            $c->setInstituicao($row["inst_usuario"]);
            $l[] = $c;
        }
        return $l;
    }

    /**
     * @see EsqueceSenhaPersistenceManager::save()
     */
    public static function save($objUsuario) {

        $pdo = Util::getPDOHandler();

        $sql = "UPDATE usuario SET senha_usuario = :senha WHERE email_usuario = :email";
        $stmt = $pdo->prepare($sql);
```

```
        $senha = $objUsuario->getSenha();
        $email = $objUsuario->getEmail();
        $stmt->bindParam(":senha" ,$senha);
        $stmt->bindParam(":email", $email);
        $stmt->execute();

        return $objUsuario;
    }
}
?>
```

10.4. EsqueceSenhaPersistenceManager.class.php

```
<?php

    /**
     * Interface que define um Manager, classe responsável pelo gerenciamento
     * de persistência de alguma classe do Model. Todos os Managers deste módulo
     * devem implementar no mínimo os métodos definidos por este contrato.
     *
     *
     * @static
     * @package model
     */
interface EsqueceSenhaPersistenceManager {

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através de sua
chave primária.
     *
     * @param mixed $key Chave da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
    public static function findByField($dado);

    /**
     * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica
da operação,
     * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser
especificada na
     * implementação do método pela classe concreta. O exemplo mais comum para chave primária
inteira é assumir
     * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser
inserida)
     * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas
atualizada).
     *
     * @param mixed $model Objeto a ser gravado.
     */
    public static function save($model);
}

?>
```

10.5. GetPasswordManager.class.php

```
<?php

    /**
     * Classe que define o gerenciador de persistência para a classe <kbd>GetPassword</kbd>.
     *
     * @static
     * @see GetPasswordPersistenceManager
     * @package model
     */

class GetPasswordManager implements GetPasswordPersistenceManager {

    /**
     * @see GetPasswordPersistenceManager::save()
     */
    public static function findByCod($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT senha_usuario FROM usuario WHERE cod_usuario = :cod");
        $stmt->bindParam(":cod", $key);
        $stmt->execute();

        if ($row = $stmt->fetch()) {
            $usuario = new Usuario();
            $usuario->setCod($key);
            $usuario->setSenha($row["senha_usuario"]);

        }

        return $usuario;
    }

    /**
     * @see GetPasswordPersistenceManager::save()
     */
    public static function save($usuario) {
        $pdo = Util::getPDOHandler();

        // Alterando senha
        $sql = "UPDATE usuario SET senha_usuario = :senha WHERE cod_usuario = :cod";

        $stmt = $pdo->prepare($sql);

        $resultado = $stmt->execute(array(
        ":cod" => $usuario->getCod(),
        ":senha" => md5($usuario->getSenha())));
    }
}
?>
```

10.6. GetPasswordPersistenceManager.class.php

```
<?php

    /**
    * Interface que define um Manager, classe responsável pelo gerenciamento
    * de persistência de alguma classe do Model. Todos os Managers deste módulo
    * devem implementar no mínimo os métodos definidos por este contrato.
    *
    *
    * @static
    * @package model
    */
interface GetPasswordPersistenceManager {

   /**
    * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
    *
    * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
    * haver nenhum objeto, <kbd>array</kbd> estará vazio.
    */
   public static function findByCod($key);

   /**
    * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica
da operação,
    * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser
especificada na
    * implementação do método pela classe concreta. O exemplo mais comum para chave primária
inteira é assumir
    * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser
inserida)
    * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas
atualizada).
    *
    * @param mixed $model Objeto a ser gravado.
    */
   public static function save($model);
}
?>
```

10.7. GrupoManager.class.php

```
<?php

    /**
     * Classe que define o gerenciador de persistência para a classe <kbd>Grupo</kbd>.
     *
     * @static
     * @see GrupoPersistenceManager
     * @package model
     */

class GrupoManager implements GrupoPersistenceManager {

    /**
     * @see GrupoPersistenceManager::fetchAll()
     */
    public static function fetchAll() {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT cod_grupo, nome_grupo, sigla_grupo, descricao_grupo FROM
grupo  WHERE nome_grupo != 'Criador' ORDER BY cod_grupo ASC");
        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {

            $c = new Grupo();
            $c->setCod($row["cod_grupo"]);
            $c->setNome($row["nome_grupo"]);
            $c->setSigla($row["sigla_grupo"]);
            $c->setDescricao($row["descricao_grupo"]);
            $l[] = $c;
        }

        return $l;

    }

    /**
     * @see GrupoPersistenceManager::fetchAll()
     */
    public static function fetchAllConfig($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT cod_grupo, nome_grupo, sigla_grupo, descricao_grupo FROM
grupo WHERE nome_grupo != 'Criador' AND cod_grupo IN (SELECT grupo_cod_grupo FROM
config_chat WHERE chat_cod_chat = :codChat) ORDER BY cod_grupo ASC");
        $stmt->bindParam(":codChat", $key);
        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {

            $c = new Grupo();
            $c->setCod($row["cod_grupo"]);
            $c->setNome($row["nome_grupo"]);
            $c->setSigla($row["sigla_grupo"]);
```

```
            $c->setDescricao($row["descricao_grupo"]);
            $l[] = $c;
        }

        return $l;

    }

    /**
     * @see GrupoPersistenceManager::findByKey()
     */
    public static function findByKey($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT * FROM grupo WHERE cod_grupo = :cod");
        $stmt->bindParam(":cod", $key);
        $stmt->execute();

        if ($row = $stmt->fetch()) {
            $grupo = new Grupo();
            $grupo->setCod($key);
            $grupo->setNome($row["nome_grupo"]);
            $grupo->setSigla($row["sigla_grupo"]);
            $grupo->setDescricao($row["descricao_grupo"]);
        }
        else {
            $grupo = new Grupo();
            $grupo->setCod(-1);
        }

        return $grupo;

    }

    /**
     * @see GrupoPersistenceManager::save()
     */
    public static function save($grupo) {

        $pdo = Util::getPDOHandler();

        if ($grupo->getCod() == 0) {
            $sql = "INSERT INTO grupo (nome_grupo, sigla_grupo, descricao_grupo) VALUES (:nome,
:sigla, :descricao)";
        } else {
            $sql = "UPDATE grupo SET nome_grupo = :nome, sigla_grupo = :sigla, descricao_grupo =
:descricao WHERE cod_grupo = :cod";
        }

        $stmt = $pdo->prepare($sql);

        if ($grupo->getCod() == 0) {
             $stmt->execute(array(":nome" => $grupo->getNome(), ":sigla" => $grupo->getSigla(),
":descricao" => $grupo->getDescricao()));
        }
        else {
            $stmt->execute(array(":cod" => $grupo->getCod(), ":nome" => $grupo->getNome(), ":sigla" =>
$grupo->getSigla(), ":descricao" => $grupo->getDescricao()));
        }
```

```
    }

    /**
     * @see GrupoPersistenceManager::delete()
     */
    public static function delete($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("DELETE FROM grupo WHERE cod_grupo = :cod");
        $stmt->bindParam(":cod", $key);
        $stmt->execute();

    }

}

?>
```

10.8. GrupoManagerPersistence.class.php

```
<?php

    /**
     * Interface que define um Manager, classe responsável pelo gerenciamento
     * de persistência de alguma classe do Model. Todos os Managers deste módulo
     * devem implementar no mínimo os métodos definidos por este contrato.
     *
     * Essa idéia segue o design pattern DAO (Data Access Object).
     *
     * @static
     * @package model
     */

interface GrupoPersistenceManager {

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchAll();

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @param mixed $key Chave da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchAllConfig($key);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através de sua
chave primária.
     *
     * @param mixed $key Chave da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
    public static function findByKey($key);

    /**
     * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica
da operação,
     * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser
especificada na
     * implementação do método pela classe concreta. O exemplo mais comum para chave primária
inteira é assumir
     * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser
inserida)
     * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas
atualizada).
     *
     * @param mixed $model Objeto a ser gravado.
     */
    public static function save($model);
```

```
    /**
     * Método utilizado para exclusão de uma entidade da classe Model correspondente através de sua chave primária.
     *
     * @param mixed $key Chave da entidade a ser excluída, segundo a definição de chave primária da classe Model.
     */
    public static function delete($key);

}

?>
```

10.9. UsuarioManager.class.php

```
<?php

   /**
    * Classe que define o gerenciador de persistência para a classe <kbd>Usuario</kbd>.
    *
    * @static
    * @see UsuarioPersistenceManager
    * @package model
    */
class UsuarioManager implements UsuarioPersistenceManager {

   /**
    * @see UsuarioPersistenceManager::fetchAll()
    */
   public static function fetchAll() {

      $pdo = Util::getPDOHandler();

      $stmt = $pdo->prepare("SELECT * FROM usuario WHERE cod_usuario > 1 ORDER BY
status_usuario, nome_usuario ASC");
      $stmt->execute();

      $arrayObjetos = array();
      while ($row = $stmt->fetch()) {
         $usuario = new Usuario();
         $usuario->setCod($row["cod_usuario"]);
         $usuario->setLogin($row["login_usuario"]);
         $usuario->setNome($row["nome_usuario"]);
         $usuario->setSenha($row["senha_usuario"]);
         $usuario->setConfirmaSenha($row["senha_usuario"]);
         $usuario->setRg($row["rg_usuario"]);
         $usuario->setCpf($row["cpf_usuario"]);
         $usuario->setDataNascimento($row["dt_nascimento_usuario"]);
         $usuario->setTelefone($row["telefone_usuario"]);
         $usuario->setEndereco($row["endereco_usuario"]);
         $usuario->setEmail($row["email_usuario"]);
         $usuario->setSexo($row["sexo_usuario"]);
         $usuario->setCidade($row["cidade_usuario"]);
         $usuario->setUf($row["uf_usuario"]);
         $usuario->setCep($row["cep_usuario"]);
         $usuario->setPais($row["pais_usuario"]);
         $usuario->setInstituicao($row["inst_usuario"]);
         $usuario->setStatus($row["status_usuario"]);
         $arrayObjetos[] = $usuario;
      }
      return $arrayObjetos;

   }

   /**
    * @see UsuarioPersistenceManager::fetchAll()
    */
   public static function fetchUsuarioAcesso($key) {

      $pdo = Util::getPDOHandler();
```

```
        $stmt = $pdo->prepare("SELECT * FROM usuario WHERE status_usuario = 't' AND cod_usuario
NOT IN (SELECT usuario_cod_usuario FROM acesso_chat WHERE chat_cod_chat = :codChat)
ORDER BY cod_usuario ASC");
        $stmt->bindParam(":codChat", $key);
        $stmt->execute();

        $arrayObjetos = array();
        while ($row = $stmt->fetch()) {
            $usuario = new Usuario();
            $usuario->setCod($row["cod_usuario"]);
            $usuario->setLogin($row["login_usuario"]);
            $usuario->setNome($row["nome_usuario"]);
            $usuario->setSenha($row["senha_usuario"]);
            $usuario->setConfirmaSenha($row["senha_usuario"]);
            $usuario->setRg($row["rg_usuario"]);
            $usuario->setCpf($row["cpf_usuario"]);
            $usuario->setDataNascimento($row["dt_nascimento_usuario"]);
            $usuario->setTelefone($row["telefone_usuario"]);
            $usuario->setEndereco($row["endereco_usuario"]);
            $usuario->setEmail($row["email_usuario"]);
            $usuario->setSexo($row["sexo_usuario"]);
            $usuario->setCidade($row["cidade_usuario"]);
            $usuario->setUf($row["uf_usuario"]);
            $usuario->setCep($row["cep_usuario"]);
            $usuario->setPais($row["pais_usuario"]);
            $usuario->setInstituicao($row["inst_usuario"]);
            $usuario->setStatus($row["status_usuario"]);
            $arrayObjetos[] = $usuario;
        }
        return $arrayObjetos;

    }

    /**
     * @see UsuarioPersistenceManager::findByKey()
     */
    public static function findByKey($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT * FROM usuario WHERE cod_usuario = :cod");
        $stmt->bindParam(":cod", $key);
        $stmt->execute();

        if ($row = $stmt->fetch()) {
            $usuario = new Usuario();
            $usuario->setCod($key);
            $usuario->setNome($row["nome_usuario"]);
            $usuario->setLogin($row["login_usuario"]);
            $usuario->setSenha($row["senha_usuario"]);
            $usuario->setConfirmaSenha($row["senha_usuario"]);
            $usuario->setRg($row["rg_usuario"]);
            $usuario->setCpf($row["cpf_usuario"]);
            $usuario->setDataNascimento($row["dt_nascimento_usuario"]);
            $usuario->setTelefone($row["telefone_usuario"]);
            $usuario->setEmail($row["email_usuario"]);
            $usuario->setEndereco($row["endereco_usuario"]);
            $usuario->setSexo($row["sexo_usuario"]);
            $usuario->setCidade($row["cidade_usuario"]);
            $usuario->setUf($row["uf_usuario"]);
```

```
            $usuario->setCep($row["cep_usuario"]);
            $usuario->setPais($row["pais_usuario"]);
            $usuario->setInstituicao($row["inst_usuario"]);
            $usuario->setStatus($row["status_usuario"]);

        } else {
            $usuario = new Usuario();
            $usuario->setCod(-1);
        }

        return $usuario;

    }

    /**
     * @see UsuarioPersistenceManager::findByLoginSenha()
     */
    public static function findByLoginSenha($login, $senha, $status) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT cod_usuario FROM usuario WHERE login_usuario = :login
AND senha_usuario = :senha AND status_usuario = :status");

        $stmt->bindParam(":login", $login);

        $stmt->bindParam(":senha", $senha);

        $stmt->bindParam(":status", $status);

        $stmt->execute();

        $cont = $stmt->rowCount();

        $usuario = new Usuario();

        $row = $stmt->fetch();

        if ($cont > 0) {
            $usuario->setCod($row["cod_usuario"]);
        }

        return $usuario;

    }

    /**
     * @see UsuarioPersistenceManager::save()
     */
    public static function save($usuario) {

        $pdo = Util::getPDOHandler();

        if ($usuario->getCod() == 0) {
            $sql = "INSERT INTO usuario (nome_usuario, login_usuario, senha_usuario, rg_usuario,
cpf_usuario, dt_nascimento_usuario, telefone_usuario, email_usuario, endereco_usuario,
sexo_usuario, cidade_usuario, uf_usuario, cep_usuario, pais_usuario, inst_usuario) VALUES (:nome,
:login, :senha, :rg, :cpf, :dt_nascimento, :telefone, :email, :endereco, :sexo, :cidade, :uf, :cep, :pais,
:instituicao)";
        }
```

```
        else {
            $sql = "UPDATE usuario SET nome_usuario = :nome, rg_usuario = :rg, cpf_usuario = :cpf,
dt_nascimento_usuario = :dt_nascimento, telefone_usuario = :telefone, email_usuario = :email,
endereco_usuario = :endereco, sexo_usuario = :sexo, cidade_usuario = :cidade, uf_usuario = :uf,
cep_usuario = :cep, pais_usuario = :pais, inst_usuario = :instituicao WHERE cod_usuario = :cod";
        }

        $stmt = $pdo->prepare($sql);

        if ($usuario->getCod() == 0) {
            $resultado = $stmt->execute(array(
            ":nome" => $usuario->getNome(),
            ":login" => $usuario->getLogin(),
            ":senha" => md5($usuario->getSenha()),
            ":rg" => $usuario->getRg(),
            ":cpf" => $usuario->getCpf(),
            ":dt_nascimento" => $usuario->getDataNascimento(),
            ":telefone" => $usuario->getTelefone(),
            ":email" => $usuario->getEmail(),
            ":endereco" => $usuario->getEndereco(),
            ":sexo" => $usuario->getSexo(),
            ":cidade" => $usuario->getCidade(),
            ":uf" => $usuario->getUf(),
            ":cep" => $usuario->getCep(),
            ":pais" => $usuario->getPais(),
            ":instituicao" => $usuario->getInstituicao()));


            ////////////////////////////////////////////////////////////////////////////
            ///////////////// CRIANDO E ENVIANDO EMAIL COM O LINK DE ATIVAÇÂO ///////////////////

            if ($resultado){

                $usuario->setCod($row["cod_usuario"]);

                //Pegando o ultimo codigo do Banco de dados
                $stmt = $pdo->prepare("SELECT max(cod_usuario) FROM usuario");
                $stmt->execute();
                $cod = $stmt->fetch();

                $stmt = $pdo->prepare("SELECT dt_criacao_usuario FROM usuario WHERE cod_usuario =
:cod");
                $stmt->bindParam(":cod", $cod[0]);
                $stmt->execute();
                $dtCriacao = $stmt->fetch();

            }
              // Utiliza  _HTTP_NAME_ constante definida na Controller, traz o caminho do servidor.
              $assunto = _MAIL_ASSUNTO_ . " Requisição de Ativação de Cadastro";
              $corpoMsg = "Requisição de cadastro no Sistema.". "\n"
              . "\n"
              . "Nome do Usuário: ".$usuario->getNome() . "\n"
              . "Login do Usuário: " .$usuario->getLogin()."\n"
              . "Instituição do Usuário: ".$usuario->getInstituicao()."\n"
              . "Email do Usuário: ".$usuario->getEmail();

              Util::mailSender(_MAIL_FROM_, $assunto, $corpoMsg);

              ////////////////////// FINAL DA CRIAÇÂO E ENVIO DE EMAIL ///////////////////////
              ////////////////////////////////////////////////////////////////////////////
```

```
        }

        else {
            $stmt->execute(array(
            ":cod" => $usuario->getCod(),
            ":nome" => $usuario->getNome(),
            ":login" => $usuario->getLogin(),
            ":rg" => $usuario->getRg(),
            ":cpf" => $usuario->getCpf(),
            ":dt_nascimento" => $usuario->getDataNascimento(),
            ":telefone" => $usuario->getTelefone(),
            ":email" => $usuario->getEmail(),
            ":endereco" => $usuario->getEndereco(),
            ":sexo" => $usuario->getSexo(),
            ":cidade" => $usuario->getCidade(),
            ":uf" => $usuario->getUf(),
            ":cep" => $usuario->getCep(),
            ":pais" => $usuario->getPais(),
            ":instituicao" => $usuario->getInstituicao()));
        }
    }

    /**
     * @see UsuarioPersistenceManager::delete()
     */
    public static function delete($key) {

        $pdo = Util::getPDOHandler();

        try {

            $pdo->beginTransaction();

            $stmt = $pdo->prepare("SELECT cod_chat FROM chat WHERE cod_usuario_chat =
:codUsuario");
            $stmt->bindParam(":codUsuario", $key);
            $stmt->execute();

            while ($row = $stmt->fetch()) {

                $codChat = $row[0];

                $stmt1 = $pdo->prepare("DELETE FROM acesso_chat WHERE chat_cod_chat =
:codChat");
                $stmt1->bindParam(":codChat", $codChat);
                $stmt1->execute();

                $stmt6 = $pdo->prepare("DELETE FROM requisicao_chat WHERE cod_chat = :codChat");
                $stmt6->bindParam(":codChat", $codChat);
                $stmt6->execute();

                $stmt2 = $pdo->prepare("DELETE FROM config_chat WHERE chat_cod_chat = :codChat");
                $stmt2->bindParam(":codChat", $codChat);
                $stmt2->execute();

                $stmt3 = $pdo->prepare("DELETE FROM chat WHERE cod_chat = :codChat");
                $stmt3->bindParam(":codChat", $codChat);
                $stmt3->execute();

            }
```

```
            $stmt4 = $pdo->prepare("DELETE FROM acesso_chat WHERE usuario_cod_usuario =
:codUsuario");
            $stmt4->bindParam(":codUsuario", $key);
            $stmt4->execute();

            $stmt7 = $pdo->prepare("DELETE FROM requisicao_chat WHERE cod_usuario =
:codUsuario");
            $stmt7->bindParam(":codUsuario", $key);
            $stmt7->execute();


            $stmt5 = $pdo->prepare("DELETE FROM usuario WHERE cod_usuario = :codUsuario");
            $stmt5->bindParam(":codUsuario", $key);
            $stmt5->execute();

            $pdo->commit();

        } catch (Exception $e) {
            $pdo->rollBack();
        }

    }

    /**
     * @see UsuarioPersistenceManager::findByLogin()
     */
    public static function findByLogin($login) {
        $pdo = Util::getPDOHandler();
        $stmt = $pdo->prepare("SELECT * FROM usuario WHERE login_usuario = :login");
        $stmt->bindParam(":login", $login);
        $stmt->execute();
        $res = $stmt->fetch();

        return $res;
    }

    /**
     * @see UsuarioPersistenceManager::findByEmail()
     */
    public static function findByEmail($email, $codUsuario) {

        $pdo = Util::getPDOHandler();

        // Quando for inserção
        if ($codUsuario == 0) {
            $stmt = $pdo->prepare("SELECT * FROM usuario WHERE email_usuario = :email");
        }
        // Quando for Atualização
        else {
            $stmt = $pdo->prepare("SELECT * FROM usuario WHERE (email_usuario = :email) AND
(cod_usuario != :cod)");
            $stmt->bindParam(":cod", $codUsuario);
        }

        $stmt->bindParam(":email", $email);
        $stmt->execute();

        $arrayObjetos = array();
        while ($row = $stmt->fetch()) {
```

```
            $usuario = new Usuario();
            $usuario->setCod($row["cod_usuario"]);
            $usuario->setLogin($row["login_usuario"]);
            $usuario->setNome($row["nome_usuario"]);
            $usuario->setSenha($row["senha_usuario"]);
            $usuario->setConfirmaSenha($row["senha_usuario"]);
            $usuario->setRg($row["rg_usuario"]);
            $usuario->setCpf($row["cpf_usuario"]);
            $usuario->setDataNascimento($row["dt_nascimento_usuario"]);
            $usuario->setTelefone($row["telefone_usuario"]);
            $usuario->setEndereco($row["endereco_usuario"]);
            $usuario->setEmail($row["email_usuario"]);
            $usuario->setSexo($row["sexo_usuario"]);
            $usuario->setCidade($row["cidade_usuario"]);
            $usuario->setUf($row["uf_usuario"]);
            $usuario->setCep($row["cep_usuario"]);
            $usuario->setPais($row["pais_usuario"]);
            $usuario->setInstituicao($row["inst_usuario"]);
            $usuario->setstatus($row["status_usuario"]);
            $arrayObjetos[] = $usuario;
        }

        return $arrayObjetos;
    }
}
?>
```

10.10. UsuarioPersistenceManager

```
<?php

    /**
     * Interface que define um Manager, classe responsável pelo gerenciamento
     * de persistência de alguma classe do Model. Todos os Managers deste módulo
     * devem implementar no mínimo os métodos definidos por este contrato.
     *
     * Essa idéia segue o design pattern DAO (Data Access Object).
     *
     * @static
     * @package model
     */
interface UsuarioPersistenceManager {

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchAll();

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @param mixed $key Chave da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchUsuarioAcesso($key);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através de sua
chave primária.
     *
     * @param mixed $key Chave da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
    public static function findByKey($key);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através de sua
chave primária.
     *
     * @param mixed $login Login da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @param mixed $senha Senha da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @param mixed $status Status da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
    public static function findByLoginSenha($login, $senha, $status);

    /**
```

```
    * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica da operação,
    * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser especificada na
    * implementação do método pela classe concreta. O exemplo mais comum para chave primária inteira é assumir
    * que o COD 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser inserida)
    * e que COD diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas atualizada).
    *
    * @param mixed $model Objeto a ser gravado.
    */
   public static function save($model);

   /**
    * Método utilizado para exclusão de uma entidade da classe Model correspondente através de sua chave primária.
    *
    * @param mixed $key Chave da entidade a ser excluída, segundo a definição de chave primária da classe Model.
    */
   public static function delete($key);

    /**
     * Método utilizado para obtenção de um login para que não haja mais de um usuário com o mesmo login no Banco de dados.
     *
     * @param mixed $login Login da entidade desejada, segundo a definição é o login do usuário.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
   public static function findByLogin($login);

      /**
     * Método utilizado para obtenção de um email para que não haja mais de um usuário com o mesmo email no Banco de dados.
     *
     * @param mixed $email Email da entidade desejada, segundo a definição é o email do usuário.
     * @param mixed $codUsuario Código do Usuário da entidade desejada, segundo a definição é o email do usuário.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
   public static function findByEmail($email, $codUsuario);
}

?>
```

11. MÓDULO USUÁRIO – TEMPLATES

11.1. alteraStatus/load.tpl

```
<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_29.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
   <td colspan="8">
   <br>
   <br>
   <form name="form" method="post" action="{php}print(Util::getURL("alterastatus.save"));{/php}">
   <input type="hidden" name="cod" value="{$usuario->getCod()}">
   <table class="form_peq" align="center">
   <tr>
      <td width="20%" align="right" class="form_label">Nome do Usuario:</td>
      <td width="80%" class="form_field"><input type="text" name="nome" disabled="true"
value="{$usuario->getNome()}" size="60" maxlength="50"></td>
   </tr>

   <tr>
      <td width="20%" align="right" class="form_label">Situação do Usuário:</td>
      <td width="80%" class="form_field">
      {if $usuario->getStatus() == 1}
         <input type="radio" name="status" value="t" checked="true"> Ativo <br>
         <input type="radio" name="status" value="f"> Inativo
      {elseif $usuario->getStatus() != 1}
         <input type="radio" name="status" value="t"> Ativo <br>
         <input type="radio" name="status" value="f" checked="true"> Inativo
      {/if}
      </td>
   </tr>

   <tr>
      <td colspan="2" class="form_field" align="center"><input type="submit" value="Salvar"
class="button"></td>
   </tr>

   {if $errors|@count > 0}

   <tr><td colspan="2" class="error"><div class="error">

      {foreach from=$errors item=error}
            {$error}<br>
      {/foreach}

   </div></td></tr>

   {/if}

   </table>
   </form>
   </td>
</tr>
```

11.2. esqueceSenha/load.tpl

```
<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_30.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
   <td colspan="8">
   <br>
   <form name="form" method="post" action="{php}print(Util::getURL("esquecesenha.save"));{/php}">
   <table class="form_peq" align="center">
   <tr>
      <td width="20%" align="right" class="form_label">Email:</td>
      <td width="80%" class="form_field"><input type="text" name="email" size="60"
maxlength="50"></td>
   </tr>

   <tr>
      <td colspan="2" align="center" class="form_field"><input type="submit" value="Enviar"
class="button"></td>
   </tr>

   {if $errors|@count > 0}

   <tr>
      <td colspan="2" class="error"><div class="error">
         {foreach from=$errors item=error}
               {$error}<br>
         {/foreach}
      </div></td>
   </tr>

   {/if}

   </table>
   </form>
   </td>
</tr>
```

11.3. esqueceSenha/mensagem.tpl

```
<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_30.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
   <td align="center" colspan="8">
   <br>
   <br>
   {$msgEmailEnviado}
   <br>
   <br>
   </td>
</tr>
```

11.4. getPassword/load.tpl

```
<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_25.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
   <td colspan="8">
   <br>
   <form name="form" method="post" action="{php}print(Util::getURL("getpassword.save"));{/php}">
   <table class="form_peq" align="center">

   <tr>
      <td align="right" valign="top" class="form_label">Senha Antiga:</td>
      <td class="form_field"><input type="password" name="senhaAntiga" size="15" maxlength="12">
<i> A senha deve ter entre 6 e 12 caracteres.</i></td>
   </tr>
   <tr>
      <td width="20%" align="right" class="form_label">Nova Senha:</td>
      <td width="80%" class="form_field"><input type="password" name="senha" size="15"
maxlength="12"></td>
   </tr>
   <tr>
      <td width="20%" align="right" class="form_label">Confirma Nova Senha:</td>
      <td width="80%" class="form_field"><input type="password" name="confirma_senha" size="15"
maxlength="12"></td>
   </tr>
   <tr>
      <td colspan="2" class="form_field" align="center"><input type="submit" value="Salvar"
class="button"></td>
   </tr>

   {if $errors|@count > 0}

   <tr>
      <td colspan="2" class="error"><div class="error">
      {foreach from=$errors item=error}
      {$error}<br>
      {/foreach}
   </tr>

   {/if}
   </table>
   </form>
   </td>
</tr>
```

11.5. getPassword/save.tpl

```
<tr>
   <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_25.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
   <td colspan="8">
   <br>
   <br>
   <table align="center">
      <tr>
         <td>Senha Alterada com Sucesso!</td>
      </tr>
   </table>
   <br>
   <br>
   </td>
</tr>
```

11.6. grupo/list.tpl

```
<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_34.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
        <br>
  {assign var="chave" value=0}
  {foreach from=$lista item=c name=loop}
     {if  $chave == 0 }
        <table class="list_peq" align="center">
        <tr>
           <td width="40%" class="list_header">Nome</td>
           <td width="10%" class="list_header">Sigla</td>
           <td width="50%" class="list_header">Descrição</td>
        </tr>
        {assign var="chave" value=1}
     {/if}

     <tr>
        <td class="list_body{$smarty.foreach.loop.iteration%2+1}"><a
href="{php}print(Util::getURL("grupo.load"));{/php}&cod={$c->getCod()}">{$c->getNome()}</a></td>
        <td class="list_body{$smarty.foreach.loop.iteration%2+1}">{$c->getSigla()}</td>
        <td class="list_body{$smarty.foreach.loop.iteration%2+1}">{$c->getDescricao()}</td>
     </tr>

  {foreachelse}

     <table align="center" class="empty">
     <tr>
        <td align="center" colspan="3"
class="message"><br>{$smarty.const.MESSAGE_EMPTY_LIST}<br><br></td>
     </tr>
     </table>

  {/foreach}
  {if $chave == 1}
     </table>
  {/if}
  </td>
</tr>
```

11.7. grupo/load.tpl

```
{if $grupo->getCod() >= 0}

   {if $grupo->getCod() == 0}
      <tr>
         <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_33.jpg" width="780"
height="85"></td>
      </tr>
   {else}
      <tr>
         <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_64.jpg" width="780"
height="85"></td>
      </tr>
   {/if}

<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
         <td colspan="8">
         <br>
   <form name="form" method="post" action="{php}print(Util::getURL("grupo.save"));{/php}">
   <input type="hidden" name="cod" value="{$grupo->getCod()}">
   <table class="form_peq" align="center">
   <tr>
      <td width="20%" align="right" class="form_label">Nome:</td>
      <td width="80%" class="form_field"><input type="text" name="nome" value="{$grupo-
>getNome()}" size="60" maxlength="30"></td>
   </tr>

   <tr>
      <td width="20%" align="right" class="form_label">Sigla:</td>
      <td width="80%" class="form_field"><input type="text" name="sigla" value="{$grupo->getSigla()}"
size="60" maxlength="10"></td>
   </tr>

   <tr>
      <td width="20%" align="right" class="form_label">Descrição:</td>
      <td width="80%" class="form_field"><input type="text" name="descricao" value="{$grupo-
>getDescricao()}" size="60" maxlength="255"></td>
   </tr>

   <tr>
      <td colspan="2" class="form_field" align="center"><input type="submit" value="Salvar"
class="button">

      {if $grupo->getCod() != 0}
            <input type="button" value="Excluir" class="button"
onclick="javascript:confirmarExclusao('{php}print(Util::getURL("grupo.delete"));{/php}&cod={$grupo-
>getCod()}');">
      {/if}

      </td>
   </tr>

   {if $errors|@count > 0}

   <tr><td colspan="2" class="error"><div class="error">

   {foreach from=$errors item=error}
      {$error}<br>
   {/foreach}
```

```
    </td></tr>

    {/if}

    </table>
    </form>

    {literal}
    <script language="javascript">
    function confirmarExclusao(url) {
      if (confirm('Este Grupo será excluído.\nVocê confirma esta ação?')) {
         window.location = url;
      }
    }
    </script>
    {/literal}
{else}
    </td>

    <tr>
            <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_33.jpg" width="780"
height="85"></td>
    </tr>

    <tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
                <td align="center" colspan="8">
            <br>
            <br>
            Grupo Inexistente!
            <br>
            <br>
        </td>
    </tr>
{/if}
    </td>
</tr>
```

11.8. usuario/close.tpl

```
<html>
<meta HTTP-EQUIV="Refresh" CONTENT="4;url=/workspace/base/public/">
<head>
<link rel="stylesheet" type="text/css" href="style.css"/>
<title>WorldConf - O Mundo fala aqui!</title>
</head>
<body background="{$smarty.const._IMAGEM_DIR_}bg.gif">
<br>
<br>
<br>
<table align="center" width="726" height="80" border="0" cellpadding="0" cellspacing="0">
	<tr>
		<td align="center">Você está saindo do sistema, aguarde enquanto transferimos
você!</td>
	</tr>
	<tr>
		<td align="center"><a href="{php}print(Util::getURL("usuario.index"));{/php}">Ou clique
aqui se não quiser esperar</a></td>
	</tr>
</table>
</body>
</html>
```

11.9. usuario/index.tpl

```
<html>
<head>
<title>WorldConf - O Mundo fala aqui!</title>
<script language="JavaScript" type="text/JavaScript">
{literal}
<!--
function MM_preloadImages() { //v3.0
  var d=document; if(d.images){ if(!d.MM_p) d.MM_p=new Array();
    var i,j=d.MM_p.length,a=MM_preloadImages.arguments; for(i=0; i<a.length; i++)
    if (a[i].indexOf("#")!=0){ d.MM_p[j]=new Image; d.MM_p[j++].src=a[i];}}
}

function MM_swapImgRestore() { //v3.0
  var i,x,a=document.MM_sr; for(i=0;a&&i<a.length&&(x=a[i])&&x.oSrc;i++) x.src=x.oSrc;
}

function MM_findObj(n, d) { //v4.01
  var p,i,x;  if(!d) d=document; if((p=n.indexOf("?"))>0&&parent.frames.length) {
    d=parent.frames[n.substring(p+1)].document; n=n.substring(0,p);}
  if(!(x=d[n])&&d.all) x=d.all[n]; for (i=0;!x&&i<d.forms.length;i++) x=d.forms[i][n];
  for(i=0;!x&&d.layers&&i<d.layers.length;i++) x=MM_findObj(n,d.layers[i].document);
  if(!x && d.getElementById) x=d.getElementById(n); return x;
}

function MM_swapImage() { //v3.0
  var i,j=0,x,a=MM_swapImage.arguments; document.MM_sr=new Array; for(i=0;i<(a.length-2);i+=3)
   if ((x=MM_findObj(a[i]))!=null){document.MM_sr[j++]=x; if(!x.oSrc) x.oSrc=x.src; x.src=a[i+2];}
}

//-->
{/literal}
</script>
</head>
<body background="{$smarty.const._IMAGEM_DIR_}bg.gif" leftmargin="0" topmargin="0"
marginwidth="0" marginheight="0" link="#000000" alink="#000000" vlink="#000000"
onload="{literal}document.form.login.focus(){/literal}">
<br>
<br>
<br>
<table align="center" width="726" height="291" border="0" cellpadding="0" cellspacing="0">
          <tr>
                    <td rowspan="8"><img src="{$smarty.const._IMAGEM_DIR_}usuario_login_01.jpg"
width="169" height="291"></td>
                    <td rowspan="8"><img src="{$smarty.const._IMAGEM_DIR_}usuario_login_02.jpg"
width="195" height="291"></td>
                    <td rowspan="8"><img src="{$smarty.const._IMAGEM_DIR_}usuario_login_03.jpg"
width="191" height="291"></td>
                    <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}usuario_login_04.jpg"
width="153" height="102"></td>
                    <td rowspan="8"><img src="{$smarty.const._IMAGEM_DIR_}usuario_login_05.jpg"
width="18" height="291"></td>
          </tr>
   <form name="form" method="post" action="{php}print(Util::getURL("usuario.validate"));{/php}">
        <input name="cod" type="hidden" value="0">
        <tr>
                    <td width="153" height="28" colspan="2" bgcolor="#BCC8CF"><input type="text"
name="login" value="" maxlength="12"></td>
        </tr>
```

```
        <tr>
          <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}usuario_login_15.jpg"
width="153" height="18"></td>
        </tr>
              <tr>
          <td width="118" height="27" bgcolor="#BCC8CF"><input name="senha" type="password"
size="12" onkeypress="if (event.keyCode == 13) document.form.submit();" maxlength="12"></td>
                    <td><a href="javascript: document.form.submit();"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('entrar','','{$smarty.const._IMAGEM_DIR_}usuario_login_13.jpg',0)">
<img src="{$smarty.const._IMAGEM_DIR_}usuario_login_08.jpg" name="entrar" width="55"
height="27" border="0"></a></td>
              </tr>
    </form>
    {if $errors|@count > 0}
    {foreach from=$errors item=error}
        <tr>
                    <td align="center" width="153" height="16" colspan="2"
bgcolor="#BCC8CF"><font face="Tahoma" size="1" color="#FF0000">Usuário ou Senha
Inválidos!</font></td>
              </tr>
    {/foreach}
    {else}
        <tr>
                    <td width="153" height="16" colspan="2" bgcolor="#BCC8CF"></td>
              </tr>
    {/if}
          <tr>
                <td align="center" width="153" height="21" colspan="2" bgcolor="#BCC8CF"><a
href="{php}print(Util::getURL("esquecesenha.load"));{/php}"><font face="Tahoma" size="1">Esqueci
minha senha!</font></a></td>
          </tr>
          <tr>
                <td colspan="2"><a href="{php}print(Util::getURL("usuario.load"));{/php}"
onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('cadastro','','{$smarty.const._IMAGEM_DIR_}usuario_login_14.jpg',0
)"><img src="{$smarty.const._IMAGEM_DIR_}usuario_login_11.jpg" name="cadastro" width="153"
height="44" border="0"></a></td>
          </tr>
          <tr>
                <td colspan="2"><img src="{$smarty.const._IMAGEM_DIR_}usuario_login_12.jpg"
width="153" height="35"></td>
          </tr>
</table>
</body>
</html>
```

11.10. usuario/list.tpl

```
<tr>
          <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_27.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
   <td colspan="8">
   <br>
   {foreach from=$lista item=c name=loop}
     {if $c->getLogin() != "" AND $chave == 0}
     <table class="list_peq" align="center">
        <tr>
           <td width="10%" class="list_header">Login</td>
           <td width="30%" class="list_header">Email</td>
           <td width="20%" class="list_header">Nome</td>
           <td width="10%" class="list_header">Instituição</td>
           <td width="5%" class="list_header">Status</td>
        </tr>
     {/if}
   <tr>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}">{$c->getLogin()}</td>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}">{$c->getEmail()}</td>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}"><a
href="{php}print(Util::getURL("usuario.load"));{/php}&cod={$c->getCod()}">{$c->getNome()}</a></td>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}">{$c->getInstituicao()}</td>
      {if $c->getStatus() == 1 }
         <td class="list_body{$smarty.foreach.loop.iteration%2+1}">Ativo</td>
      {else}
         <td class="list_body{$smarty.foreach.loop.iteration%2+1}">Inativo</td>
      {/if}
   </tr>
   {assign var="chave" value="1"}
   {foreachelse}

   <table align="center" class="empty">
   <tr>
      <td align="center" colspan="2"
class="message"><br>{$smarty.const.MESSAGE_EMPTY_LIST}<br><br></td>
   </tr>

   {/foreach}
   <table align="right">
   <tr align="center">
      <td align="center">
        <br><input type="button" value="Alterar Senha Admin" class="button"
onclick="javascript:location='{php}print(Util::getURL("getpassword.load"));{/php}';">  &nbs
p;       
      </td>
   </tr>
   </table>
   </td>
</tr>
```

11.11. usuario/load.tpl

```
{if $usuario->getCod() >= 0}
<tr>
   {if $opcao == 1}
      <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_28.jpg" width="780"
height="85"></td>
   {else}
      <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_26.jpg" width="780"
height="85"></td>
   {/if}
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
   <td colspan="8">
   <br>
   <form name="form" method="post" action="{php}print(Util::getURL("usuario.save"));{/php}">
   <input type="hidden" name="cod" value="{$usuario->getCod()}">
   <table class="form" align="center">

   <tr>
      <td align="left" class="list_body1" colspan="2">Os campos com * são obrigatórios.</td>
   </tr>

   <tr>
      <td width="20%" align="right" class="form_label">Nome: *</td>
      <td width="80%" class="form_field"><input type="text" name="nome" value="{$usuario-
>getNome()}" size="50" maxlength="50"></td>
   </tr>
   <tr>
      <td align="right" valign="top" class="form_label">Login: *</td>
      <td class="form_field">
      {if ($usuario->getLogin()) == ""}
         <input type="text" name="login" value="{$usuario->getLogin()}" size="20" maxlength="12"><i>
O login deve ter entre 3 e 12 caracteres.</i></td>
      {elseif ($usuario->getCod() == 0)}
         <input type="text" name="login" value="{$usuario->getLogin()}" size="20" maxlength="12"><i>
O login deve ter entre 3 e 12 caracteres.</i></td>
      {else}
         <input type="text" name="login" disabled="true" value="{$usuario->getLogin()}" size="20"
maxlength="12"></td>
      {/if}
      </td>
   </tr>

   {if ($usuario->getSenha()) == ""}
   <tr>
      <td width="20%" align="right" class="form_label">Senha: *</td>
      <td width="80%" class="form_field"><input type="password" name="senha" value="{$usuario-
>getSenha()}" size="20" maxlength="12"><i> A senha deve ter entre 6 e 12 caracteres.</i></td>
   {elseif ($usuario->getCod() == 0)}
      <td width="20%" align="right" class="form_label">Senha: *</td>
      <td width="80%" class="form_field"><input type="password" name="senha" value="{$usuario-
>getSenha()}" size="20" maxlength="12"></td>
   </tr>
   {/if}

   {if ($usuario->getSenha()) == ""}
   <tr>
      <td width="20%" align="right" class="form_label">Confirma Senha: *</td>
```

```
        <td width="80%" class="form_field"><input type="password" name="confirma_senha"
value="{$usuario->getConfirmaSenha()}" size="20" maxlength="12"></td>
    {elseif ($usuario->getCod()==0)}
        <td width="20%" align="right" class="form_label">Confirma Senha: *</td>
        <td width="80%" class="form_field"><input type="password" name="confirma_senha"
value="{$usuario->getConfirmaSenha()}" size="20" maxlength="12"></td>
    </tr>
    {/if}

    <tr>
        <td width="20%" align="right" class="form_label">RG:  </td>
        <td width="80%" class="form_field"><input type="text" name="rg" value="{$usuario->getRg()}"
size="15" maxlength="15"><i>Ex.: 99999999 (Sem separadores)</i></td>
    </tr>

    <tr>
        <td width="20%" align="right" class="form_label">CPF: *</td>
        <td width="80%" class="form_field"><input type="text" name="cpf" value="{$usuario->getCpf()}"
size="11" maxlength="11"><i>Ex.: 99999999999 (Sem separadores)</i></td>
    </tr>

    <tr>
        <td width="20%" align="right" class="form_label">Data Nascimento: *</td>
        <td width="80%" class="form_field"><input type="text" name="data_nascimento"
value="{$usuario->getDataNascimento()}" size="10" maxlength="10"><i>Ex.: 01/01/1901</i></td>
    </tr>

    <tr>
        <td width="20%" align="right" class="form_label">Telefone:  </td>
        <td width="80%" class="form_field"><input type="text" name="telefone" value="{$usuario-
>getTelefone()}" size="25" maxlength="25"></td>
    </tr>

    <tr>
        <td align="right" valign="top" class="form_label">Email: *</td>
        <td class="form_field"><input type="text" name="email" value="{$usuario->getEmail()}" size="50"
maxlength="50"></td>
    </tr>

    <tr>
        <td width="20%" align="right" class="form_label">Endereço: *</td>
        <td width="80%" class="form_field"><input type="text" name="endereco" value="{$usuario-
>getEndereco()}" size="50" maxlength="40"></td>
    </tr>

    <tr>
        <td width="20%" align="right" class="form_label">Sexo:  </td>
        <td width="80%" class="form_field">
        <select name="sexo">
            <option>{$usuario->getSexo()}</option>
            <option>M</option>
            <option>F</option>
        </select>
        </td>
    </tr>

    <tr>
        <td width="20%" align="right" class="form_label">Cidade:  </td>
        <td width="80%" class="form_field"><input type="text" name="cidade" value="{$usuario-
>getCidade()}" size="50" maxlength="40"></td>
```

```
    </tr>

    <tr>
       <td width="20%" align="right" class="form_label">CEP:  </td>
       <td width="80%" class="form_field"><input type="text" name="cep" value="{$usuario->getCep()}"
size="10" maxlength="10"><i>Ex.: 99999999 (Sem separadores)</i></td>
    </tr>

    <tr>
       <td width="20%" align="right" class="form_label">UF:  </td>
       <td width="80%" class="form_field">
       <select name="uf">
             <option>{$usuario->getUf()}</option>
             <option>AC</option>
             <option>AL</option>
             <option>AP</option>
             <option>AM</option>
             <option>BA</option>
             <option>CE</option>
             <option>DF</option>
             <option>ES</option>
             <option>GO</option>
             <option>MA</option>
             <option>MT</option>
             <option>MS</option>
             <option>MG</option>
             <option>PA</option>
             <option>PB</option>
             <option>PR</option>
             <option>PE</option>
             <option>PI</option>
             <option>RJ</option>
             <option>RN</option>
             <option>RS</option>
             <option>RO</option>
             <option>RR</option>
             <option>SC</option>
             <option>SP</option>
             <option>SE</option>
             <option>TO</option>
       </td>
    </tr>

    <tr>
       <td width="20%" align="right" class="form_label">País:  </td>
       <td width="80%" class="form_field">
       <select name="pais" value="{$usuario->getPais()}">
          <option selected> {$usuario->getPais()}</option>
          <option value=" "></option>
          <option value="AFEGANISTAO">  AFEGANISTAO </option>
          <option value="AFRICA DO SUL">   AFRICA DO SUL   </option>
          <option value="ALBANIA">  ALBANIA  </option>
          <option value="ALEMANHA">   ALEMANHA    </option>
          <option value="ANDORRA">   ANDORRA </option>
          <option value="ANGOLA">   ANGOLA  </option>
          <option value="ANTIGUA E BARBUDA">   ANTIGUA E BARBUDA   </option>
          <option value="ARABIA SAUDITA">   ARABIA SAUDITA  </option>
          <option value="ARGELIA">   ARGELIA </option>
          <option value="ARGENTINA">   ARGENTINA   </option>
          <option value="ARMENIA">   ARMENIA   </option>
```

```
<option value="ARUBA">  ARUBA  </option>
<option value="AUSTRALIA">  AUSTRALIA  </option>
<option value="AUSTRIA">  AUSTRIA </option>
<option value="AZERBAIJAO">  AZERBAIJAO  </option>
<option value="BAHAMAS">  BAHAMAS  </option>
<option value="BANGLADESH">  BANGLADESH  </option>
<option value="BARBADOS">  BARBADOS  </option>
<option value="BELGICA">  BELGICA </option>
<option value="BERMUDAS">  BERMUDAS  </option>
<option value="BOLIVIA">  BOLIVIA </option>
<option value="BRASIL">  BRASIL </option>
<option value="BOTSUANA">  BOTSUANA  </option>
<option value="BULGARIA">  BULGARIA </option>
<option value="CABO VERDE">  CABO VERDE  </option>
<option value="CAMAROES">  CAMAROES  </option>
<option value="CAMBOJA">  CAMBOJA </option>
<option value="CANADA">  CANADA  </option>
<option value="CATAR">  CATAR  </option>
<option value="CAZAQUISTAO">  CAZAQUISTAO  </option>
<option value="CHILE">  CHILE  </option>
<option value="CHINA">  CHINA  </option>
<option value="CHIPRE">  CHIPRE  </option>
<option value="CINGAPURA">  CINGAPURA  </option>
<option value="COLOMBIA">  COLOMBIA  </option>
<option value="CONGO">  CONGO  </option>
<option value="COREIA DO NORTE">  COREIA DO NORTE </option>
<option value="COREIA DO SUL">  COREIA DO SUL  </option>
<option value="COSTA RICA">  COSTA RICA  </option>
<option value="CROACIA">  CROACIA  </option>
<option value="CUBA">  CUBA  </option>
<option value="DINAMARCA">  DINAMARCA  </option>
<option value="EGITO">  EGITO  </option>
<option value="EL SALVADOR">  EL SALVADOR </option>
<option value="EMIRADOS ARABES UNIDOS">  EMIRADOS ARABES UNIDOS  </option>
<option value="EQUADOR">  EQUADOR </option>
<option value="ESLOVENIA">  ESLOVENIA </option>
<option value="ESPANHA">  ESPANHA </option>
<option value="ESTADOS UNIDOS">  ESTADOS UNIDOS  </option>
<option value="ESTONIA">  ESTONIA  </option>
<option value="ETIOPIA">  ETIOPIA </option>
<option value="FILIPINAS">  FILIPINAS  </option>
<option value="FINLANDIA">  FINLANDIA  </option>
<option value="FRANCA">  FRANÇA </option>
<option value="GAMBIA">  GAMBIA  </option>
<option value="GANA">  GANA  </option>
<option value="GEORGIA">  GEORGIA  </option>
<option value="GRECIA">  GRECIA  </option>
<option value="GROENLANDIA">  GROENLANDIA </option>
<option value="GUATEMALA">  GUATEMALA  </option>
<option value="GUIANA">  GUIANA  </option>
<option value="GUIANA FRANCESA">  GUIANA FRANCESA </option>
<option value="GUINE">  GUINE  </option>
<option value="GUINE-BISSAU">  GUINE-BISSAU  </option>
<option value="HAITI">  HAITI  </option>
<option value="HOLANDA">  HOLANDA </option>
<option value="HONDURAS">  HONDURAS  </option>
<option value="HONG KONG">  HONG KONG  </option>
<option value="HUNGRIA">  HUNGRIA  </option>
<option value="INDIA">  INDIA  </option>
<option value="INDONESIA">  INDONESIA  </option>
```

```
<option value="IRA">  IRÃ  </option>
<option value="IRAQUE">  IRAQUE  </option>
<option value="IRLANDA">  IRLANDA </option>
<option value="ISLANDIA">  ISLANDIA  </option>
<option value="ISRAEL">  ISRAEL  </option>
<option value="ITALIA">  ITALIA  </option>
<option value="IUGOSLAVIA">  IUGOSLAVIA  </option>
<option value="JAMAICA">  JAMAICA </option>
<option value="JAPAO">  JAPAO  </option>
<option value="JORDANIA">  JORDANIA  </option>
<option value="LIBANO">  LIBANO  </option>
<option value="LIBIA">  LIBIA  </option>
<option value="LITUANIA">  LITUANIA  </option>
<option value="LUXEMBURGO">  LUXEMBURGO  </option>
<option value="MACAU">  MACAU  </option>
<option value="MADAGASCAR">  MADAGASCAR  </option>
<option value="MALASIA">  MALASIA </option>
<option value="MALTA">  MALTA  </option>
<option value="MARROCOS">  MARROCOS  </option>
<option value="MEXICO">  MEXICO  </option>
<option value="MOCAMBIQUE">  MOCAMBIQUE  </option>
<option value="MONACO">  MONACO  </option>
<option value="MONGOLIA">  MONGOLIA  </option>
<option value="NAMIBIA">  NAMIBIA </option>
<option value="NEPAL">  NEPAL  </option>
<option value="NICARAGUA">  NICARAGUA  </option>
<option value="NIGERIA">  NIGERIA </option>
<option value="NORUEGA">  NORUEGA </option>
<option value="NOVA ZELANDIA">  NOVA ZELANDIA  </option>
<option value="OMA">  OMÃ </option>
<option value="PANAMA">  PANAMA  </option>
<option value="PAPUA NOVA GUINE">  PAPUA NOVA GUINE  </option>
<option value="PAQUISTAO">  PAQUISTAO  </option>
<option value="PARAGUAI">  PARAGUAI  </option>
<option value="PERU">  PERU  </option>
<option value="POLONIA">  POLONIA  </option>
<option value="PORTO RICO">  PORTO RICO  </option>
<option value="PORTUGAL">  PORTUGAL  </option>
<option value="QUENIA">  QUENIA  </option>
<option value="REINO UNIDO">  REINO UNIDO </option>
<option value="REPUBLICA TCHECA">  REPUBLICA TCHECA  </option>
<option value="REPUBLICA DOMINICANA">  REPUBLICA DOMINICANA  </option>
<option value="ROMENIA">  ROMENIA </option>
<option value="RUANDA">  RUANDA  </option>
<option value="RUSSIA">  RUSSIA  </option>
<option value="SENEGAL">  SENEGAL </option>
<option value="SERRA LEOA">  SERRA LEOA  </option>
<option value="SIRIA">  SIRIA  </option>
<option value="SOMALIA">  SOMALIA </option>
<option value="SRI LANKA">  SRI LANKA  </option>
<option value="SUDAO">  SUDAO  </option>
<option value="SUECIA">  SUECIA  </option>
<option value="SUICA">  SUIÇA  </option>
<option value="SURINAME">  SURINAME  </option>
<option value="TAILANDIA">  TAILANDIA  </option>
<option value="TAIWAN">  TAIWAN  </option>
<option value="TANZANIA">  TANZANIA  </option>
<option value="TIMOR LESTE">  TIMOR LESTE </option>
<option value="TONGA">  TONGA  </option>
<option value="TRINIDAD E TOBAGO">  TRINIDAD E TOBAGO  </option>
```

```
            <option value="TUNISIA">   TUNISIA </option>
            <option value="TURQUIA">   TURQUIA </option>
            <option value="UCRANIA">   UCRANIA </option>
            <option value="UGANDA">   UGANDA  </option>
            <option value="URUGUAI">   URUGUAI </option>
            <option value="UZBEQUISTAO">   UZBEQUISTAO   </option>
            <option value="VENEZUELA">   VENEZUELA   </option>
            <option value="VIETNA">   VIETNA  </option>
            <option value="ZAIRE">   ZAIRE   </option>
            <option value="ZAMBIA">   ZAMBIA  </option>
            <option value="ZIMBABUE">   ZIMBABUE    </option>
            <option value="OUTROS">   OUTROS    </option>
        </td>
    </tr>

    <tr>
        <td width="20%" align="right" class="form_label">Instituição: *</td>
        <td width="80%" class="form_field"><input type="text" name="instituicao" value="{$usuario-
>getInstituicao()}" size="50" maxlength="50"></td>
    </tr>

    <tr>
        <td colspan="3" class="form_field" align="center"><input type="submit" value="Salvar"
class="button">

    {if $opcao == 3}

               <input type="button" value="Excluir" class="button"
onclick="javascript:confirmarExclusao('{php}print(Util::getURL("usuario.delete"));{/php}&cod={$usuario
->getCod()}');">

               <input type="button" value="Alterar Status" class="button"
onclick="javascript:location='{php}print(Util::getURL("alterastatus.load"));{/php}&cod={$usuario-
>getCod()}';">

        {/if}

        </td>
    </tr>

    {if $errors|@count > 0}

    <tr>
        <td colspan="2" class="error"><div class="error">

        {foreach from=$errors item=error}
          {$error}<br>
        {/foreach}

        </div></td>
    </tr>

    {/if}

    </table>
    </form>

    {literal}
    <script language="javascript">
    function confirmarExclusao(url) {
```

```
      if (confirm('Este Usuário será excluído.\nVocê confirma esta ação?')) {
         window.location = url;
      }
   }
   </script>
   {/literal}
{else}
   </td>

   <tr>
         <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_26.jpg" width="780"
height="85"></td>
   </tr>
   <tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
            <td align="center" colspan="8">
         <br>
         <br>
         Usuário Inexistente!
         <br>
         <br>
      </td>
   </tr>
{/if}
</td>
</tr>
```

11.12. usuario/save.tpl

```
<tr>
   {if $opcao == 1}
      <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_28.jpg" width="780"
height="85"></td>
   {else}
      <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_26.jpg" width="780"
height="85"></td>
   {/if}
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
   <td colspan="8">
   <br>
   <br>
   <table align="center">
      <tr>
         {if $opcao == 1}
            <td>Cadastro Efetuado com Sucesso! Aguarde a confirmação do mesmo por parte do
Administrador.</td>
         {else}
            <td>Cadastro Alterado com Sucesso!</td>
         {/if}

      </tr>
   </table>
   <br>
   <br>
   </td>
</tr>
```

12. MÓDULO CHAT – CONFIGURAÇÃO

12.1. config.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<config>

   <!-- CONSTANTES VALIDAS PARA ESTE MODULO -->

   <!-- constante obrigatoria: nome do modulo -->
   <constant name="MODULE_NAME" value="[Módulo Chat]" />

   <!-- ACTIONS VALIDAS PARA ESTE MODULO -->

   <!-- Chat - List -->
   <action name="chat.list" class="ChatListAction" method="execute">
      <result name="success" type="smarty">chat/list.tpl</result>
   </action>

   <!-- Chat - Detalhes -->
   <action name="chat.load" class="ChatLoadAction">
      <result name="success" type="smarty">chat/load.tpl</result>
   </action>

   <!-- Chat - Salvar -->
   <action name="chat.save" class="ChatSaveAction">
      <result name="success" type="action">configChat.load</result>
      <result name="input" type="smarty">chat/load.tpl</result>
   </action>

   <!-- Chat - Excluir -->
   <action name="chat.delete" class="ChatDeleteAction">
      <result name="success" type="action">chat.list</result>
   </action>

   <!-- Chat - Index -->
   <action name="chat.index" class="ChatIndexAction">
      <result name="success" type="smarty">chat/index.tpl</result>
   </action>

   <!-- Chat - Entrar -->
   <action name="chat.entra" class="ChatEntraAction">
      <result name="success" type="smarty">chat/chat.tpl</result>
      <result name="input" type="smarty">chat/index.tpl</result>
   </action>

   <!-- Chat - Topo -->
   <action name="chat.topo" class="ChatTopoAction">
      <result name="success" type="smarty">chat/topo.tpl</result>
   </action>

   <!-- Chat - Princ -->
   <action name="chat.princ" class="ChatPrincAction">
      <result name="success" type="smarty">chat/princ.tpl</result>
   </action>

   <!-- Chat - Imagem -->
   <action name="chat.imagem" class="ChatImagemAction">
```

```
    <result name="success" type="smarty">chat/imagem.tpl</result>
</action>

<!-- Chat - Form -->
<action name="chat.form" class="ChatFormAction">
    <result name="success" type="smarty">chat/form.tpl</result>
</action>

<!-- Chat - Ler -->
<action name="chat.ler" class="ChatLerAction">
    <result name="success" type="smarty">chat/ler.tpl</result>
    <result name="input" type="smarty">chat/kick.tpl</result>
</action>

<!-- Chat - Usuários -->
<action name="chat.usuarios" class="ChatUsuariosAction">
    <result name="success" type="smarty">chat/usuarios.tpl</result>
</action>

<!-- Chat - Trata -->
<action name="chat.trata" class="ChatTrataAction">
    <result name="success" type="smarty">chat/form.tpl</result>
</action>

<!-- Chat - Sair -->
<action name="chat.sair" class="ChatSairAction">
    <result name="success" type="smarty">chat/sair.tpl</result>
</action>

<!-- Chat - Arquivo -->
<action name="chat.arquivo" class="ChatArquivoAction">
    <result name="input" type="smarty">chat/upload.tpl</result>
    <result name="success" type="smarty">chat/arquivo.tpl</result>
</action>

<!-- Chat - UsuariosList -->
<action name="chat.usuarioslist" class="ChatUsuariosListAction">
    <result name="success" type="smarty">chat/usuarios_list.tpl</result>
</action>

<!-- Chat - Requisição -->
<action name="chat.requisicao" class="ChatRequisicaoAction">
    <result name="success" type="smarty">chat/requisicao.tpl</result>
</action>

<!-- Recurso - List -->
<action name="recurso.list" class="RecursoListAction" method="execute">
    <result name="success" type="smarty">recurso/list.tpl</result>
</action>

<!-- Recurso - Detalhes -->
<action name="recurso.load" class="RecursoLoadAction">
    <result name="success" type="smarty">recurso/load.tpl</result>
</action>

<!-- Recurso - Salvar -->
<action name="recurso.save" class="RecursoSaveAction">
    <result name="success" type="action">recurso.list</result>
    <result name="input" type="smarty">recurso/load.tpl</result>
</action>
```

```
<!-- Recurso - Excluir -->
<action name="recurso.delete" class="RecursoDeleteAction">
   <result name="success" type="action">recurso.list</result>
</action>

<!-- Configuracao Chat - Listar -->
<action name="configChat.list" class="ConfigChatListAction">
   <result name="success" type="smarty">configChat/list.tpl</result>
</action>

<!-- Configuracao Chat - Detalhes -->
<action name="configChat.load" class="ConfigChatLoadAction">
   <result name="success" type="smarty">configChat/load.tpl</result>
</action>

<!-- Configuracao Chat - Salvar -->
<action name="configChat.save" class="ConfigChatSaveAction">
   <result name="success" type="smarty">configChat/load.tpl</result>
   <result name="input" type="smarty">configChat/load.tpl</result>
</action>

<!-- Configuracao Chat - Excluir -->
<action name="configChat.delete" class="ConfigChatDeleteAction">
   <result name="success" type="smarty">configChat/load.tpl</result>
</action>

<!-- Configuracao Chat Usuário - Listar -->
<action name="chatUsuario.list" class="ChatUsuarioListAction">
   <result name="success" type="smarty">chatUsuario/list.tpl</result>
</action>

<!-- Configuracao Chat Usuário - Detalhes -->
<action name="chatUsuario.load" class="ChatUsuarioLoadAction">
   <result name="success" type="smarty">chatUsuario/load.tpl</result>
   <result name="input" type="smarty">chatUsuario/list.tpl</result>
</action>

<!-- Configuracao Chat Usuário - Salvar -->
<action name="chatUsuario.save" class="ChatUsuarioSaveAction">
   <result name="success" type="smarty">chatUsuario/load.tpl</result>
   <result name="input" type="smarty">chatUsuario/load.tpl</result>
</action>

<!-- Configuracao Chat Usuário - Excluir -->
<action name="chatUsuario.delete" class="ChatUsuarioDeleteAction">
   <result name="success" type="smarty">chatUsuario/load.tpl</result>
   <result name="input" type="smarty">chatUsuario/load.tpl</result>
</action>

<!-- Requisicao Chat - Detalhes -->
<action name="requisicaoChat.load" class="RequisicaoChatLoadAction">
   <result name="success" type="smarty">requisicaoChat/load.tpl</result>
</action>

<!-- Requisicao Chat - Salvar -->
<action name="requisicaoChat.save" class="RequisicaoChatSaveAction">
   <result name="input" type="action">chat.list</result>
   <result name="success" type="smarty">requisicaoChat/load.tpl</result>
</action>
```

```
    <!-- Requisicao Chat - Recusar -->
    <action name="requisicaoChat.delete" class="RequisicaoChatDeleteAction">
        <result name="input" type="smarty">requisicaoChat/load.tpl</result>
        <result name="success" type="smarty">requisicaoChat/load.tpl</result>
    </action>

</config>
```

13. MÓDULO CHAT – CLASSES PACOTE ACTION

13.1. ChatArquivoAction.class.php

```
<?php

   /**
    * Action responsável por armazenar o Arquivo enviado e gravar a mensagem no banco.
    *
    * @see Action
    * @package control
    */
class ChatArquivoAction extends Action {

   /**
    * Efetua o tratamento das requisições. Monta a tela para Upload e armazena o mesmo no
Servidor.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] == ""){
        $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
        <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
        <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
              <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
        </tr></table></body></html>";
        echo $teste;
        exit();
      }

      $cod = $_REQUEST["cod"];

      if ($cod == 0) {

         $codChat = $_REQUEST["cod_chat"];
         $codUsuario = $_REQUEST["cod_usuario"];
         $output = array();
         $status = 1;
         $output["usuarios"] = ChatUsuarioManager::imprimir($codChat, $status);
         $output["cod_usuario"] = $codUsuario;
         $output["cod_chat"] = $codChat;
         $output[PARAM_ACTION_RESULT] = Action::INPUT;
      }
      else {

         $destinatario = $_REQUEST["destinatario"];
         $arquivo = $_FILES["arquivo"];
         $nome = $_REQUEST["nome"];
         $codChat = $_REQUEST["cod_chat"];
         $codUsuario = $_REQUEST["cod_usuario"];
         $status_msg = "ON";
```

```
        if ($_FILES["arquivo"]["name"] != "" ) {

            if ($_FILES['arquivo']['size'] < 2000) {
                $wMenor = "";
                $hMenor = "";
                $wMaior = "";
                $hMaior = "";
            }
            else {
                $wMenor = "0.10";
                $hMenor = "0.10";
                $wMaior = "0.50";
                $hMaior = "0.50";
            }

            $usuario = ChatUsuarioManager::findByCodUsuario($codUsuario, $codChat);

            $mensagem = new ChatMensagem();

            $perfila = ChatUsuarioManager::nick($usuario->getNome(), $codChat);
            $perfilb = ChatUsuarioManager::nick($destinatario, $codChat);

            $hora = date("(H:i:s)");

            $caminho = "../modulos/chat/arquivos/".$_FILES["arquivo"]["name"];

            $gravar = _ROOT_PATH_."modulos/chat/arquivos/".$_FILES["arquivo"]["name"];

            move_uploaded_file($_FILES['arquivo']['tmp_name'], $gravar);

            $imagemsize = getimagesize($gravar);

            if ($wMenor != "") {
                $wi = $imagemsize[0];
                $he = $imagemsize[1];

                $wMenor = ($wi * $wMenor);
                $hMenor = ($he * $hMenor);
                $wMaior = ($wi * $wMaior);
                $hMaior = ($he * $hMaior);

            }

            if (($_FILES['arquivo']['type'] == 'image/gif') or ($_FILES['arquivo']['type'] == 'image/pjpeg') or
($_FILES['arquivo']['type'] == 'image/png') or ($_FILES['arquivo']['type'] == 'image/bmp') or
($_FILES['arquivo']['type'] == 'image/jpeg')) {
                $mes.="$hora $perfila <i>indica para</i> $perfilb: "."Clique para Visualizar a Imagem: <a
href=".$caminho." target='blank'><img src=".$caminho." width='".$wMenor."' height='".$hMenor."'
border='0'></a>";
                $mes.="*<a href=".$caminho." target='blank'><img src=".$caminho." width='".$wMaior."'
height='".$hMaior."' border='0'></a>*";
            }
            else {
                $mes="$hora $perfila <i>indica para</i> $perfilb: "."Clique no link para Abrir o Arquivo:
"."<a href=".$caminho."
target='javascript:parent.imagem.location()'>".$_FILES['arquivo']['name']."<a>";
            }

            $time = time();
```

```
            $mensagem->setTime($time);
            $mensagem->setRemetente($usuario->getNome());
            $mensagem->setDestinatario($destinatario);
            $mensagem->setMensagem($mes);
            $mensagem->setStatus($status_msg);
            $mensagem->setCodUsuario($codUsuario);
            $mensagem->setCodChat($codChat);

            ChatMensagemManager::save($mensagem);
            ChatMensagemManager::delete($codChat);
        }

        $output = array();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      }
      return $output;
   }

}

?>
```

13.2. ChatDeleteAction.class.php

```
<?php

/**
 * Action responsável pela exclusão de Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ChatDeleteAction extends Action {

    /**
     * Efetua o tratamento das requisições. Exclui os dados do Chat solicitado através do Cod.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $cod = $_REQUEST["cod"];

        ChatManager::delete($cod);

        $output = array();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }

}

?>
```

13.3. ChatEntraAction.class.php

```
<?php

/**
 * Action responsável pela entrada de um usuário em um Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ChatEntraAction extends Action {

    /**
     * Efetua o tratamento das requisições. Realiza a entrada do usuário no Chat solicitado.
     * Caso a validação das informações do usuário no Chat falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $codUsuario = $_REQUEST["cod_usuario"];
        $codChat = $_REQUEST["cod_chat"];
        $apelido = $_REQUEST["apelido"];
        $cor = $_REQUEST["cor"];
        $status = 1;

        $usuario = ChatUsuarioManager::findByCodUsuario($codUsuario, $codChat);
        $usuario->setNome($apelido);
        $usuario->setCor($cor);

        $errors = $this->validate($usuario, $codChat);

        if (count($errors) > 0) {
            $output = array();
            $output[PARAM_ERRORS] = $errors;
            $output["codChat"] = $codChat;
            $output["codUsuario"] = $codUsuario;
            $output[PARAM_ACTION_RESULT] = Action::INPUT;
            $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        }
        elseif (count($errors) == 0){

            $usuario->setStatus($status);
```

```
        ChatUsuarioManager::save($usuario);
        $recurso = $_SESSION["configSalaUsuario"][$codChat];

        $aux = explode("-",$recurso);

        $recursoSala = array();
        foreach ($aux as $r) {
           $recursoSala[$r] = $r;
        }

        $time = time();
        $destinatario = "TODOS";
        $nick = ChatUsuarioManager::nick($apelido,$codChat);
        $msg = "$nick entra no Chat.";
        $status = "";

        $mensagem = new ChatMensagem();
        $mensagem->setTime($time);
        $mensagem->setRemetente($apelido);
        $mensagem->setDestinatario($destinatario);
        $mensagem->setMensagem($msg);
        $mensagem->setStatus($status);
        $mensagem->setCodChat($codChat);
        $mensagem->setCodUsuario($codUsuario);

        ChatMensagemManager::save($mensagem);
        ChatMensagemManager::delete($codChat);

        $chat = ChatManager::findByKey($codChat);

        $output = array();
        $output["usuario"] = $usuario;
        $output["codUsuario"] = $codUsuario;
        $output["codChat"] = $codChat;
        $output["nomeChat"] = $chat->getNome();
        $output["recursoSala"] = $recursoSala;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
    }

    return $output;
}

/**
 * Efetua a validação dos campos do formulário
 * @see Action::validate()
 * @param object $usuário que será validado
 * @param mixed $codChat que será validado
 * @return array $errors
 */
public function validate($usuario, $codChat) {
    $errors = array();

    $apelido = $usuario->getNome();

    if ($usuario->getCodUsuario() == ""){
        $errors[] = "Você não tem permissão para entrar neste Chat.";
    }
    else{

        if ($usuario->getNome() == "") {
```

```
            $errors[] = "O apelido deve ser informado.";
          }
          if (strtoupper($usuario->getNome()) == "TODOS") {
            $errors[] = "Apelido inválido.";
          }
          $nick_cont = ChatUsuarioManager::findByNickCont($apelido, $codChat);
          if ($nick_cont == 1){
            $errors[] = "Já existe um usuário com este nick no chat.";
          }
        }
        return $errors;
    }

}
?>
```

13.4. ChatFormAction.class.php

```
<?php

/**
 * Action responsável pela criação do frame responsável pelo envio de mensagens de texto.
 *
 * @see Action
 * @package control
 */
class ChatFormAction extends Action {

    /**
     * Efetua o tratamento das requisições. Monta o frame responsável pelo envio de mensagens de
texto.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
           echo "É NECESSÁRIO EFETUAR O LOGIN";
           exit();
        }

        $codUsuario = $_REQUEST["cod_usuario"];
        $codChat = $_REQUEST["cod_chat"];

        $usuario = ChatUsuarioManager::findByCodUsuario($codUsuario, $codChat);
        $perfila = ChatUsuarioManager::nick($usuario->getNome(),$codChat);

        $chat = ChatManager::findByKey($codChat);

        $codUsuarioChat = $chat->getCodUsuarioChat();

        $req = RequisicaoChatManager::fetchCont($codChat);

        if (($codUsuario == $codUsuarioChat) AND ($req > 0)){
            $requisicao = 1;
        }
        else {
            $requisicao = 0;
        }

        $recurso = $_SESSION["configSalaUsuario"][$codChat];

        $aux = explode("-",$recurso);

        $recursoSala = array();
        foreach ($aux as $r) {
            $recursoSala[$r] = $r;
        }

        $output = array();

        $output["apelido"] = $perfila;
        $output["nome"] = $usuario->getNome();
        $output["codUsuario"] = $usuario->getCodUsuario();
```

```
        $output["codChat"] = $chat->getCod();
        $output["nomeChat"] = $chat->getNome();
        $output["destinatario"] = "TODOS";
        $output["recursoSala"] = $recursoSala;
        $output["req_ant"] = $req;
        $output["requisicao"] = $requisicao;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

        return $output;
    }

}

?>
```

13.5. ChatImagemAction.class.php

```
<?php

/**
 * Action responsável pela montagem do frame destinado as imagens.
 *
 * @see Action
 * @package control
 */
class ChatImagemAction extends Action {

    /**
     * Efetua o tratamento das requisições. Monta o frame destinado as Imagens enviadas a Sala de
Conferência.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
           echo "É NECESSÁRIO EFETUAR O LOGIN";
           exit();
        }

        $output = array();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

        return $output;
    }

}

?>
```

13.6. ChatIndexAction.class.php

```
<?php

/**
 * Action responsável pela exibição da tela de entrada na Sala de Conferência.
 *
 * @see Action
 * @package control
 */
class ChatIndexAction extends Action {

    /**
     * Efetua o tratamento das requisições. Monta a tela para a entrada no Sala de Conferência.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
           echo "É NECESSÁRIO EFETUAR O LOGIN";
           exit();
        }

        $codUsuario = $_SESSION["codUsuario"];
        $codChat = $_REQUEST["cod_chat"];

        $output = array();
        $output["codUsuario"] = $codUsuario;
        $output["codChat"] = $codChat;
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }

}

?>
```

13.7. ChatLerAction.class.php

```
<?php

/**
 * Action responsável pela impressão das mensagens no Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ChatLerAction extends Action {

    /**
     * Efetua o tratamento das requisições. Imprime as mensagens no Chat.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
           echo "É NECESSÁRIO EFETUAR O LOGIN";
           exit();
        }

        $codUsuario = $_REQUEST["cod_usuario"];
        $codChat = $_REQUEST["cod_chat"];

        $usuario = ChatUsuarioManager::findByCodUsuario($codUsuario, $codChat);

        $nome = $usuario->getNome();
        $last = $usuario->getLast();
        $status = $usuario->getStatus();

        if($status == 0){
             $output[PARAM_ACTION_RESULT] = Action::INPUT;

             return $output;
        }

        $msg = ChatMensagemManager::imprimir($nome, $last, $codChat);

        if (!empty($msg[0]) OR !empty($msg[1])){

            $usuario->setLast($last);
            ChatUsuarioManager::save($usuario);
            ChatUsuarioManager::atualiza($codChat);

            $img = strstr($msg[0], "<a");
            $img1 = strstr($msg[0], "<img");

            if (($img != FALSE) AND ($img1 != FALSE)){
               $img = explode('*',$msg[0]);
               $cont = strrpos($img[1], "a");
               $tagImg = substr($img[1], 0, $cont);
               $tagImg = "<html><head><link rel='stylesheet' type='text/css'
href='style.css'/></head><body><table align='center' class='form'><tr><td class='list_header'
align='center'>IMAGEM</td></tr></table><table height='75%' width='75%'
align='center'><tbody><tr><td align='center'>".$img[1]."</td></tr></tbody></table></body><html>";
```

```
            $tipoArquivo = "img";
            $msg[0] = $img[0].$img[2];
        }

    }

    $output = array();
    $output["msg"] = $msg[0];
    $output["som"] = $msg[1];
    $output["cod_chat"] = $codChat;
    $output["cod_usuario"] = $codUsuario;
    $output["tipoArquivo"] = $tipoArquivo;
    $output["tagImg"] = $tagImg;
    $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

    return $output;
  }

}

?>
```

13.8. ChatListAction.class.php

```
<?php

/**
 * Action responsável pela obtenção da lista de Chats do sistema.
 *
 * @see Action
 * @package control
 */
class ChatListAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa a lista de todos os Chats do sistema.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $codUsuario = $_SESSION["codUsuario"];
        $loginUsuario = $_SESSION["loginUsuario"];

        $array1 = array();
        $array2 = array();
        $array3 = array();
        $array4 = array();
        $array5 = array();
        $array6 = array();
        $array7 = array();
        $ckChat = array();

        if ($_SESSION["codUsuario"] == 1){

            $edit = ChatManager::fetch();
            foreach ($edit as $aux) {
                $ckChat[] = $aux->getCod();
            }
        }
        else {

            $array1 = ChatManager::fetchAll($codUsuario);

            foreach ($array1 as $aux) {
                $ckChat[] = $aux->getCod();
            }
```

```
            $edit = ChatManager::fetchAllConfig($codUsuario);

            foreach ($edit as $aux) {
                $ckChat[] = $aux->getCod();
            }

            $array4 = ChatManager::fetchList($codUsuario);
            $array6 = ChatManager::fetchListMod($codUsuario);

        }

        $recursoSala = array();
        foreach ($ckChat as $codChat) {
            $recurso = ChatUsuarioManager::findByRecursoUsuario($codUsuario, $codChat);
            $recursoSala[$codChat] = implode("-",$recurso);
        }

        $_SESSION["configSalaUsuario"] = $recursoSala;

        for($i = 0; $i < sizeof($array1); $i++){
                $chat = $array1[$i];
                $chat->setDescricao(wordwrap($chat->getDescricao(),100,"<br>",1));
                $array2[] = $chat;
        }

        for($i = 0; $i < sizeof($edit); $i++){
                $chat = $edit[$i];
                $chat->setDescricao(wordwrap($chat->getDescricao(),87,"<br>",1));
                $array3[] = $chat;
        }

        for($i = 0; $i < sizeof($array4); $i++){
                $chat = $array4[$i];
                $chat->setDescricao(wordwrap($chat->getDescricao(),87,"<br>",1));
                $array5[] = $chat;
        }

        for($i = 0; $i < sizeof($array6); $i++){
                $chat = $array6[$i];
                $chat->setDescricao(wordwrap($chat->getDescricao(),87,"<br>",1));
                $array7[] = $chat;
        }

        $output = array();
        $output["list"] = $array5;
        $output["listMod"] = $array7;
        $output["edit"] = $array3;
        $output["lista"] = $array2;
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        return $output;
    }

}

?>
```

13.9. ChatLoadAction.class.php

```
<?php

/**
 * Action responsável pela exibição dos detalhes de Chat do sistema.
* @see Action
 * @package control
 */
class ChatLoadAction extends Action {

   /**
    * Efetua o tratamento das requisições. Pesquisa os dados do Chat solicitado ou então
    * cria um novo Chat, de acordo com a solicitação. Estes dados podem ser alterados e
    * posteriormente salvos com a action de inserção/atualização.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] == ""){
        $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
        <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
        <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
              <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
        </tr></table></body></html>";
        echo $teste;
        exit();
      }

      $cod = $_REQUEST["cod"];
      if ((is_numeric($cod)) && ($cod > 0)) {
         $chat = ChatManager::findByKey($cod);
      }
      else {
         $chat = new Chat();
         $chat->setCod($cod);
         $data = getdate(time());
         $data = $data[mday]."/".$data[mon]."/".$data[year];
         $chat->setDtInicio($data);
         $codUsuario = $_SESSION["codUsuario"];
         $chat->setCodUsuarioChat($codUsuario);
      }
      $output = array();
      $output["chat"] = $chat;
      $loginUsuario = $_SESSION["loginUsuario"];
      $output["login"] = $loginUsuario;
      $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      return $output;
   }

}
?>
```

13.10. ChatPermissaoAction.class.php

```
<?php

/**
 * Action responsável pela impressão das mensagens no Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ChatPermissaoAction extends Action {

    /**
     * Efetua o tratamento das requisições. Imprime as mensagens no Chat.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $permissao = $_REQUEST["msg"];

        if ($permissao == "Escrever") {
            $mensagem = "Escrever no Chat.";
        }
        else {
            if ($permissao == "Ler") {
                $mensagem = "Leitura no Chat.";
            }
        }

        $output = array();
        $output["msg"] = $mensagem;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        return $output;
    }

}

?>
```

13.11. ChatPrincAction.class.php

```
<?php

/**
 * Action responsável pela criação do frame principal, local das mensagens de texto.
 *
 * @see Action
 * @package control
 */
class ChatPrincAction extends Action {

    /**
     * Efetua o tratamento das requisições. Responsável pela criação do frame principal, local das
mensagens de texto.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $output = array();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        return $output;
    }

}

?>
```

13.12. ChatSairAction.class.php

```
<?php

/**
 * Action responsável pela saída de um Usuário do Chat.
 *
 * @see Action
 * @package control
 */
class ChatSairAction extends Action {

   /**
    * Efetua o tratamento das requisições. Responsável pela saída do usuário da sala, armazenando
no banco de dados
    * uma mensagem informando da saída do usuário.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] == ""){
        $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
        <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
        <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
              <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
        </tr></table></body></html>";
        echo $teste;
        exit();
      }

      $codUsuario = $_REQUEST["cod_usuario"];
      $codChat = $_REQUEST["cod_chat"];

      $user = ChatUsuarioManager::findByCodUsuarioCont($codUsuario, $codChat);

      if ($user == 1) {

         $usuario = ChatUsuarioManager::findByCodUsuario($codUsuario, $codChat);
         $nome = $usuario->getNome();
         $sala = ChatUsuarioManager::findByNickCont($nome, $codChat);

         if ($sala == 1){

           $time = time();
           $destinatario = "TODOS";
           $nick = ChatUsuarioManager::nick($usuario->getNome(),$codChat);
           $msg = "$nick sai do Chat.";
           $status = "";

           $mensagem = new ChatMensagem();
           $mensagem->setTime($time);
           $mensagem->setRemetente($usuario->getNome());
           $mensagem->setDestinatario($destinatario);
           $mensagem->setMensagem($msg);
```

```
            $mensagem->setStatus($status);
            $mensagem->setCodChat($codChat);
            $mensagem->setCodUsuario($codUsuario);

            ChatMensagemManager::save($mensagem);
            ChatMensagemManager::delete($codChat);

            $empty = "";
            $last = 0;
            $status_user = 0;

            $usuario->setNome($empty);
            $usuario->setCor($empty);
            $usuario->setLast($last);
            $usuario->setStatus($status_user);

            ChatUsuarioManager::save($usuario);
          }
        }
        $output = array();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

        return $output;
    }

}

?>
```

13.12. ChatSaveAction.class.php

```
<?php

/**
 * Action responsável pela inserção e atualização de Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ChatSaveAction extends Action {

    /**
     * Efetua o tratamento das requisições. Realiza a inserção/atualização do Chat solicitado.
     * Caso a validação das informações do Chat falhe, retorna o resultado <kbd>Action::INPUT</kbd>.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $chat = new Chat();
        $chat->setCod($_REQUEST["cod"]);
        $chat->setNome(ltrim($_REQUEST["nome"]));
        $chat->setDescricao(ltrim($_REQUEST["descricao"]));
        $chat->setDtInicio($_REQUEST["dtInicio"]);
        $chat->setDtFim($_REQUEST["dtFim"]);
        $chat->setCodUsuarioChat($_REQUEST["codUsuarioChat"]);

        $errors = $this->validate($chat);

        $output = array();

        if (count($errors) > 0) {
            $output["chat"] =  $chat;
            $output[PARAM_ERRORS] = $errors;
            $output[PARAM_ACTION_RESULT] = Action::INPUT;
        }
        else {
          if($_FILES["banner"]["name"] != ""){
              $caminho = "../modulos/chat/banner/".$_FILES["banner"]["name"];
              $gravar = _ROOT_PATH_."modulos/chat/banner/".$_FILES["banner"]["name"];
              move_uploaded_file($_FILES['banner']['tmp_name'], $gravar);
              $chat->setBanner($caminho);
           }
           $cod = ChatManager::save($chat);
```

```
            if ($cod != 0) {
                mkdir("../modulos/chat/jnlp/$cod");
                copy("../modulos/chat/jnlp/modelo/cliente.jar", "../modulos/chat/jnlp/$cod/cliente.jar");
                copy("../modulos/chat/jnlp/modelo/audio.jnlp", "../modulos/chat/jnlp/$cod/audio.jnlp");

                $arquivo = "../modulos/chat/jnlp/$cod/audio.jnlp";

                $fp = fopen($arquivo,"r");
                $texto = fread($fp, 4000);
                $textoMod = str_replace("codChat",$cod,$texto);
                fclose($fp);

                $fp = fopen($arquivo,"w+");
                fwrite($fp, $textoMod);
                fclose($fp);

                $output[FORWARD_PARAM] = "codChat=".$cod;
            }
            else {
                $output[FORWARD_PARAM] = "cod=".$chat->getCod();
            }
            $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        }
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }

     /**
     * Efetua a validação dos campos do formulário
     * @see Action::validate()
     * @param object $chat que será validado
     * @return array $errors
     */
    public function validate($chat) {
        $errors = array();

        if ($chat->getNome() == "") {
            $errors[] = "O nome deve ser informado.";
        }
        if (($chat->getDescricao() == "")) {
            $errors[] = "A descrição deve ser informada.";
        }
        if($_FILES["banner"]["name"] != ""){
            if (($_FILES['banner']['type'] != 'image/gif') AND ($_FILES['banner']['type'] != 'image/pjpeg')
AND ($_FILES['banner']['type'] != 'image/png') AND ($_FILES['banner']['type'] != 'image/bmp') AND
($_FILES['banner']['type'] != 'image/jpeg')) {
                $errors[] = "Tipo de arquivo inválido.";
                if ($_FILES['banner']['size'] > 2000000){
                    $errors[] = "O banner deverá ter tamanho máximo de 2MB.";
                }
            }
        }
        if (Util::validaData($chat->getDtInicio())) {
            $errors[] = "Data Início Inválida!";
        }
        if (Util::validaData($chat->getDtFim())) {
            $errors[] = "Data Fim Inválida!";
        }
```

```
        return $errors;
    }

}
?>
```

13.13. ChatTopoAction.class.php

```
<?php

/**
 * Action responsável pela montagem do frame com as informações do topo da página do Chat.
 *
 * @see Action
 * @package control
 */
class ChatTopoAction extends Action {

    /**
     * Efetua o tratamento das requisições. Monta o frame superior da página com as informações
necessárias.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $codChat = $_REQUEST["cod_chat"];
        $chat = ChatManager::findByKey($codChat);

        $output = array();
        $output["banner"] = $chat->getBanner();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        return $output;
    }

}

?>
```

13.14. ChatTrataAction.class.php

```
<?php

/**
 * Action responsável pela envio das mensagens de texto.
 *
 * @see Action
 * @package control
 */
class ChatTrataAction extends Action {

    /**
     * Efetua o tratamento das requisições. Monta a mensagem de texto e armazena no banco de
dados.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
            <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
            <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                    <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
            </tr></table></body></html>";
            echo $teste;
            exit();
        }

        $reqAnt = $_REQUEST["req_ant"];

        $apelido = $_REQUEST["nome"];
        $destinatario = $_REQUEST["destinatario"];
        $status = $_REQUEST["status"][0];
        $modofala = $_REQUEST["modofala"];
        $imagem = $_REQUEST["imagem"];
        $message = $_REQUEST["message"];
        $codChat = $_REQUEST["cod_chat"];
        $codUsuario = $_REQUEST["cod_usuario"];
        $nomeChat = $_REQUEST["nomeChat"];
        $sound = $_REQUEST["sound"];
        $imagem_c = $_REQUEST["imagem_c"];
        $sound_c = $_REQUEST["sound_c"];

        $perfila = ChatUsuarioManager::nick($apelido, $codChat);
        $perfilb = ChatUsuarioManager::nick($destinatario, $codChat);

        $hora=date("(H:i:s)");
        $res="";
        $checked="";
        if($status == "ON"){
            $res="reservadamente";
            $checked="ON";
        }
```

```
        if(!empty($sound) AND $imagem != "Imagem") {
            $mes="$hora $perfila <i>$res ".$modofala."</i> $perfilb: $imagem_c ".$message." "."<img
src=$imagem>";
        }
        elseif(!empty($sound)) {
            $mes="$hora $perfila <i>$res ".$modofala."</i> $perfilb: $imagem_c ".$message;
        }
        elseif($imagem != "Imagem") {
            $mes="$hora $perfila <i>$res ".$modofala."</i> $perfilb: ".$message." "."<img src=$imagem>";
        }
        else{
            $mes="$hora $perfila <i>$res ".$modofala."</i> $perfilb: ".$message;
        }

        $usuario = ChatUsuarioManager::findByCodUsuario($codUsuario, $codChat);

        $mensagem = new ChatMensagem();

        $time = time();
        $mensagem->setTime($time);
        $mensagem->setRemetente($apelido);
        $mensagem->setDestinatario($destinatario);
        $mensagem->setMensagem($mes);
        $mensagem->setStatus($status);
        $mensagem->setSom($sound);
        $mensagem->setCodChat($codChat);
        $mensagem->setCodUsuario($codUsuario);

        $tags = strip_tags($message);

        if($tags == $message){
            if((trim($message) != "") OR ($imagem != "Imagem") OR !empty($sound)){
                ChatMensagemManager::save($mensagem);
                ChatMensagemManager::delete($codChat);
            }
        }

        $chat = ChatManager::findByKey($codChat);

        $codUsuarioChat = $chat->getCodUsuarioChat();

        $req = RequisicaoChatManager::fetchCont($codChat);

        if (($codUsuario == $codUsuarioChat) AND ($req > $reqAnt) AND ($req > 0)){
            $requisicao = 1;
            $reqAnt = $req;
        }
        else {
            $reqAnt = $req;
            $requisicao = 0;
        }

        $recurso = $_SESSION["configSalaUsuario"][$codChat];

        $aux = explode("-",$recurso);

        $recursoSala = array();
        foreach ($aux as $r) {
            $recursoSala[$r] = $r;
```

```
        }

        $output = array();

        $output["apelido"] = $perfila;
        $output["nome"] = $usuario->getNome();
        $output["checked"] = $checked;
        $output["destinatario"] = $destinatario;
        $output["codChat"] = $codChat;
        $output["codUsuario"] = $codUsuario;
        $output["nomeChat"] = $nomeChat;
        $output["recursoSala"] = $recursoSala;
        $output["requisicao"] = $requisicao;
        $output["req_ant"] = $reqAnt;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

        return $output;
    }

}

?>
```

13.16. ChatUsuariosAction.class.php

```
<?php

/**
 * Action responsável pela apresentação dos usuários on-line no Chat.
 *
 * @see Action
 * @package control
 */
class ChatUsuariosAction extends Action {

    /**
     * Efetua o tratamento das requisições. Apresenta os usuários on-line no Chat.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        $codChat = $_REQUEST["cod_chat"];
        $status = 1;

        $output = array();
        $output["usuario"] = ChatUsuarioManager::imprimir($codChat, $status);
        $output["cod_chat"] = $codChat;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        return $output;
    }

}

?>
```

13.17. ChatUsuariosListAction.class.php

```
<?php

/**
 * Action responsável pela apresentação do frame destinado a lista de usuários.
 *
 * @see Action
 * @package control
 */
class ChatUsuariosListAction extends Action {

    /**
     * Efetua o tratamento das requisições. Apresentação do frame destinado a lista de usuários.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        $codChat = $_REQUEST["cod_chat"];

        $output = array();
        $output["codChat"] = $codChat;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        return $output;
    }

}

?>
```

13.18. ChatUsuarioDeleteAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela exclusão de Configurações de Acesso aos Chats do sistema.
 *
 * @see Action
 * @package control
 */
class ChatUsuarioDeleteAction extends Action {

    /**
     * Efetua o tratamento das requisições. Exclui os dados da Configurações de Acessos ao Chat
     * solicitada através do Código do Chat, Código do Usuário e Código do Grupo.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
            $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
            <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
            <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                    <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
            </tr></table></body></html>";
            echo $teste;
            exit();
        }

        $listaCod = $_REQUEST["listaChat"];

        $cod = $_REQUEST["cod"];

        if ($cod == 0) {

            $listaAcesso = array();

            $arrChat = array();

            $arrChat = explode("-", $listaCod);

            array_pop($arrChat);

            foreach($arrChat as $codChat) {

                $lista = array();
                $chat = ChatManager::findByKey($codChat);
                $lista["codChat"] = $codChat;
                $lista["nomeChat"] = $chat->getNome();
```

```
            $acesso = ChatUsuarioManager::findByKey($codChat);
            $usuarioGrupo = array();
            foreach($acesso as $a) {
                $codGrupo = $a->getCodGrupo();
                $grupo = GrupoManager::findByKey($codGrupo);
                $codUsuario = $a->getCodUsuario();
                $usuario = UsuarioManager::findByKey($codUsuario);
                $usuarioGrupo["nomeUsuario"] = $usuario->getNome();
                $usuarioGrupo["codUsuario"] = $codUsuario;
                $usuarioGrupo["codGrupo"] = $codGrupo;
                $usuarioGrupo["nomeGrupo"] = $grupo->getNome();
                $lista["usuarioGrupo"][] = $usuarioGrupo;
            }
            sort($lista["usuarioGrupo"]);
            $listaAcesso[] = $lista;
        }

        $output = array();

        $output["cod"] = 0;
        $output["listaAcesso"] = $listaAcesso;
        $output["listaChat"] = $listaCod;
        $output[PARAM_ACTION_RESULT] = Action::INPUT;

    }
    else {

        $chatUsuario = new ChatUsuario();

        $arrListaChat = $_REQUEST["arrCodChat"];

        $ckChat = $_REQUEST["codChat"];
        $ckUsuario = $_REQUEST["codUsuario"];
        $ckGrupo = $_REQUEST["codGrupo"];

        if (! empty($ckChat)) {
            $arrObjChat = array();
            foreach ($ckChat as $codChat) {
                $chat = new Chat();
                $chat->setCod($codChat);
                $arrObjChat[] = $chat;
            }
            $chatUsuario->setCodChat($arrObjChat);
        }

        if (! empty($ckUsuario)) {
            $arrObjUsuario = array();
            foreach ($ckUsuario as $codUsuario) {
                $usuario = new Usuario();
                $usuario->setCod($codUsuario);
                $arrObjUsuario[] = $usuario;
            }
            $chatUsuario->setCodUsuario($arrObjUsuario);
        }

        ChatUsuarioManager::delete($chatUsuario);

        $listaAcesso = array();

        $arrChat = array();
```

```
            $arrChat = explode("-", $listaCod);

            array_pop($arrChat);

            foreach($arrChat as $codChat) {

                $lista = array();
                $chat = ChatManager::findByKey($codChat);
                $lista["codChat"] = $codChat;
                $lista["nomeChat"] = $chat->getNome();

                $acesso = ChatUsuarioManager::findByKey($codChat);
                $usuarioGrupo = array();
                foreach($acesso as $a) {
                    $codGrupo = $a->getCodGrupo();
                    $grupo = GrupoManager::findByKey($codGrupo);
                    $codUsuario = $a->getCodUsuario();
                    $usuario = UsuarioManager::findByKey($codUsuario);
                    $usuarioGrupo["nomeUsuario"] = $usuario->getNome();
                    $usuarioGrupo["codUsuario"] = $codUsuario;
                    $usuarioGrupo["codGrupo"] = $codGrupo;
                    $usuarioGrupo["nomeGrupo"] = $grupo->getNome();
                    $lista["usuarioGrupo"][] = $usuarioGrupo;
                }
                sort($lista["usuarioGrupo"]);
                $listaAcesso[] = $lista;
            }

            $output = array();

            $output["cod"] = 0;
            $output["listaAcesso"] = $listaAcesso;
            $output["listaChat"] = $listaCod;
            $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

        }
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;

    }

}

?>
```

13.19. ChatUsuarioListAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela obtenção da lista das Configurações de Acessos dos Chats do sistema.
 *
 * @see Action
 * @package control
 */
class ChatUsuarioListAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa a lista de todos as Configurações de
     * Acessos aos Chat do sistema.
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
            $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
            <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
            <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                    <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
            </tr></table></body></html>";
            echo $teste;
            exit();
        }

        $output = array();
        $acesso = ChatUsuarioManager::fetchDistinct();
        $lista = array();
        foreach ($acesso as $chatUsuario) {
            $lista[] = ChatManager::findByKey($chatUsuario->getCodChat());
        }
        $output["arrObjChat"] = $lista;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        return $output;
    }

}

?>
```

13.20. ChatUsuarioLoadAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela exibição dos detalhes das Configurações dos Acessos aos Chats do
sistema.
 *
 * @see Action
 * @package control
 */
class ChatUsuarioLoadAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa os dados das Configurações dos Acessos aos
Chats solicitado ou então
     * cria uma nova Configuração de Acesso ao Chat, de acordo com a solicitação. Estes dados
podem ser alterados e
     * posteriormente salvos com a action de inserção/atualização.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
            $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
            <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
            <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                    <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
            </tr></table></body></html>";
            echo $teste;
            exit();
        }

        $chatUsuario = new ChatUsuario();

        $cod = $_REQUEST["cod"];
        $codChat = $_REQUEST["codChat"];

        if ($codChat != 0) {

            $chat = ChatManager::findByKey($codChat);

            $arrGrupo = GrupoManager::fetchAllConfig($codChat);
            $arrUsuario = UsuarioManager::fetchUsuarioAcesso($codChat);

            $output = array();

            $output["codChat"] = $codChat;
            $output["nomeChat"] = $chat->getNome();
            $output["arrGrupo"] = $arrGrupo;
```

```
    $output["arrUsuario"] = $arrUsuario;
    $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

}
else {

    $ckChat = $_REQUEST["chat"];
    $cod = 1;

    $chatUsuario->setCodChat($ckChat);

    $errors = $this->validate($chatUsuario);

    $output = array();

    if (count($errors) > 0) {

        $acesso = ChatUsuarioManager::fetchDistinct();
        $lista = array();
        foreach ($acesso as $chatUsuario) {
            $lista[] = ChatManager::findByKey($chatUsuario->getCodChat());
        }
        $output["arrObjChat"] = $lista;
        $output[PARAM_ERRORS] = $errors;
        $output[PARAM_ACTION_RESULT] = Action::INPUT;

    }
    else {

        $listaAcesso = array();

        $codChat = $chatUsuario->getCodChat();
        $lista = array();
        $chat = ChatManager::findByKey($codChat);
        $lista["codChat"] = $codChat;
        $listaCodChat .= $codChat."-";
        $lista["nomeChat"] = $chat->getNome();
        $lista["arrGrupo"] = GrupoManager::fetchAllConfig($codChat);

        $acesso = ChatUsuarioManager::findByKey($codChat);
        $usuarioGrupo = array();
        $todos = "";
        foreach($acesso as $a) {
            $codGrupo = $a->getCodGrupo();
            $grupo = GrupoManager::findByKey($codGrupo);
            $codUsuario = $a->getCodUsuario();
            $usuario = UsuarioManager::findByKey($codUsuario);
            $usuarioGrupo["nomeUsuario"] = $usuario->getNome();
            $usuarioGrupo["codUsuario"] = $codUsuario;
            $usuarioGrupo["codGrupo"] = $codGrupo;
            $usuarioGrupo["nomeGrupo"] = $grupo->getNome();
            $lista["usuarioGrupo"][] = $usuarioGrupo;
        }
        sort($lista["usuarioGrupo"]);
        $listaAcesso[] = $lista;

        $output["cod"] = $cod;
        $output["chat"] = $ckChat;
        $output["listaAcesso"] = $listaAcesso;
        $output["listaChat"] = $listaCodChat;
```

```
            $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        }

      }
      $loginUsuario = $_SESSION["loginUsuario"];
      $output["login"] = $loginUsuario;
      $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      return $output;

  }

   /**
   * Efetua a validação dos campos do formulário
   * @see Action::validate()
   * @param object $chatUsuario que será validado
   * @return array $errors
   */
  public function validate($chatUsuario) {
      $errors = array();

      if ($chatUsuario->getCodChat() == "") {
          $errors[] = "É necessário escolher pelo menos uma Sala.";
      }

      return $errors;
  }
}

?>
```

13.21. ChatUsuarioSaveAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela inserção e atualização de Configurações de Acesso ao Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ChatUsuarioSaveAction extends Action {

    /**
     * Efetua o tratamento das requisições. Realiza a inserção/atualização de Configurações de Acesso
ao Chat solicitado.
     * Caso a validação das informações do Plataforma falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $output = array();

        $chatUsuario = new ChatUsuario();

        $cod = $_REQUEST["cod"];
        $codChat = $_REQUEST["codChat"];
        $nomeChat = $_REQUEST["nomeChat"];

        if ($codChat != "") {

            $listaArr = $_REQUEST["listaArr"];

            if (!empty($listaArr)) {
                $arrObjLista = array();
                foreach ($listaArr as $listAcesso) {
                    $acesso = new ChatUsuario();
                    $acesso->setCodChat($listAcesso["codChat"]);
                    $acesso->setCodUsuario($listAcesso["codUsuario"]);
                    $acesso->setCodGrupo($listAcesso["codGrupo"]);
```

```
            $arrObjLista[] = $acesso;
        }
    }

    $ckChat[] = $codChat;

    $ckUsuario = $_REQUEST["ckUsuario"];
    $ckGrupo = $_REQUEST["ckGrupo"];

    if (! empty($ckChat)) {
        $arrObjChat = array();
        foreach ($ckChat as $cod) {
            $chat = new Chat();
            $chat->setCod($cod);
            $arrObjChat[] = $chat;
        }
        $chatUsuario->setCodChat($arrObjChat);
    }

    if (! empty($ckUsuario)) {
        $arrObjUsuario = array();
        foreach ($ckUsuario as $codUsuario) {
            $usuario = new Usuario();
            $usuario->setCod($codUsuario);
            $arrObjUsuario[] = $usuario;
        }
        $chatUsuario->setCodUsuario($arrObjUsuario);
    }

    if (! empty($ckGrupo)) {
        $arrObjGrupo = array();
        foreach ($ckGrupo as $codGrupo) {
            $grupo = new Grupo();
            if ($codGrupo != "") {
                $grupo->setCod($codGrupo);
                $arrObjGrupo[] = $grupo;
            }
        }
        $chatUsuario->setCodGrupo($arrObjGrupo);
    }

    $errors = $this->validate($chatUsuario);

    if (count($errors) > 0) {

        $arrGrupo = GrupoManager::fetchAllConfig($codChat);
        $arrUsuario = UsuarioManager::fetchUsuarioAcesso($codChat);

        $output = array();

        if (!empty ($arrObjLista)) {
            $output["lista"] = $arrObjLista;
            $output["chave"] = 1;
        }

        $output["codChat"] = $codChat;
        $output["nomeChat"] = $nomeChat;
        $output["arrGrupo"] = $arrGrupo;
        $output["arrUsuario"] = $arrUsuario;
        $output[PARAM_ERRORS] = $errors;
```

```
        $output[PARAM_ACTION_RESULT] = Action::INPUT;

    }
    else {

            $output = array();

            $lista = array();

        $error = ChatUsuarioManager::saveAcesso($chatUsuario, $codChat);

            if ($error == 23505) {
                $msgError = "Você não pode Inserir Registros Duplicados!!!";
                $output[PARAM_ERRORS] = $msgError;
            }
            else {
          foreach ($ckChat as $cod) {
              foreach ($ckUsuario as $codUsuario) {
                  foreach ($ckGrupo as $codGrupo) {
                      $lista[] = ChatUsuarioManager::findByCod($cod, $codUsuario, $codGrupo);
                              }
              }
                  }
             }

        if (! empty($arrObjLista)) {
            foreach ($arrObjLista as $acesso) {
                $lista[] = $acesso;
            }
        }

        $arrGrupo = GrupoManager::fetchAllConfig($codChat);
        $arrUsuario = UsuarioManager::fetchUsuarioAcesso($codChat);

        $output["codChat"] = $codChat;
        $output["nomeChat"] = $nomeChat;
        $output["arrGrupo"] = $arrGrupo;
        $output["arrUsuario"] = $arrUsuario;
        $output["lista"] = $lista;
        $output["chave"] = 1;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
    }
}
else {

    $arrListaChat = $_REQUEST["arrCodChat"];

    $ckChat = $_REQUEST["chat"];
    $ckUsuario = $_REQUEST["codUsuario"];
    $ckGrupo = $_REQUEST["codGrupo"];

    if (! empty($ckChat)) {
        $arrObjChat = array();
        foreach ($ckChat as $codC) {
            $chat = new Chat();
            $chat->setCod($codC);
            $arrObjChat[] = $chat;
        }
        $chatUsuario->setCodChat($arrObjChat);
    }
```

```
if (! empty($ckUsuario)) {
    $arrObjUsuario = array();
    foreach ($ckUsuario as $codUsuario) {
        $usuario = new Usuario();
        $usuario->setCod($codUsuario);
        $arrObjUsuario[] = $usuario;
    }
    $chatUsuario->setCodUsuario($arrObjUsuario);
}

if (! empty($ckGrupo)) {
    $arrObjGrupo = array();
    foreach ($ckGrupo as $codGrupo) {
        $grupo = new Grupo();
        if ($codGrupo != "") {
            $grupo->setCod($codGrupo);
            $arrObjGrupo[] = $grupo;
        }
    }
    $chatUsuario->setCodGrupo($arrObjGrupo);
}

$arrChat = array();

foreach ($arrListaChat as $chat) {
    $arrChat[] = $chat["codChat"];
}

$errors = $this->validate($chatUsuario);

if (count($errors) > 0) {

    foreach($arrChat as $codChat) {

        $lista = array();

        $chat = ChatManager::findByKey($codChat);
        $lista["codChat"] = $codChat;
        $listaCodChat .= $codChat."-";
        $lista["nomeChat"] = $chat->getNome();
        $lista["arrGrupo"] = GrupoManager::fetchAll();

        $acesso = ChatUsuarioManager::findByKey($codChat);
        $usuarioGrupo = array();

        foreach($acesso as $a) {
            $codGrupo = $a->getCodGrupo();
            $grupo = GrupoManager::findByKey($codGrupo);
            $codUsuario = $a->getCodUsuario();
            $usuario = UsuarioManager::findByKey($codUsuario);
            $usuarioGrupo["nomeUsuario"] = $usuario->getNome();
            $usuarioGrupo["codUsuario"] = $codUsuario;
            $usuarioGrupo["codGrupo"] = $codGrupo;
            $usuarioGrupo["nomeGrupo"] = $grupo->getNome();
            $lista["usuarioGrupo"][] = $usuarioGrupo;
        }
        sort($lista["usuarioGrupo"]);
        $listaAcesso[] = $lista;
    }
```

```
            $output["cod"] = $cod;
            $output["chat"] = $codChat;
            $output["listaAcesso"] = $listaAcesso;
            $output["listaChat"] = $listaCodChat;
            $output[PARAM_ERRORS] = $errors;
            $output[PARAM_ACTION_RESULT] = Action::INPUT;
        }
        else {

            $codChat = 0;

            ChatUsuarioManager::saveAcesso($chatUsuario, $codChat);

            foreach($arrChat as $codChat) {

                $lista = array();

                $chat = ChatManager::findByKey($codChat);
                $lista["codChat"] = $codChat;
                $listaCodChat .= $codChat."-";
                $lista["nomeChat"] = $chat->getNome();
                $lista["arrGrupo"] = GrupoManager::fetchAllConfig($codChat);

                $acesso = ChatUsuarioManager::findByKey($codChat);
                $usuarioGrupo = array();

                foreach($acesso as $a) {
                    $codGrupo = $a->getCodGrupo();
                    $grupo = GrupoManager::findByKey($codGrupo);
                    $codUsuario = $a->getCodUsuario();
                    $usuario = UsuarioManager::findByKey($codUsuario);
                    $usuarioGrupo["nomeUsuario"] = $usuario->getNome();
                    $usuarioGrupo["codUsuario"] = $codUsuario;
                    $usuarioGrupo["codGrupo"] = $codGrupo;
                    $usuarioGrupo["nomeGrupo"] = $grupo->getNome();
                    $lista["usuarioGrupo"][] = $usuarioGrupo;
                }
                sort($lista["usuarioGrupo"]);
                $listaAcesso[] = $lista;
            }
            $output["cod"] = $cod;
            $output["chat"] = $codChat;
            $output["listaChat"] = $listaCodChat;
            $output["listaAcesso"] = $listaAcesso;
            $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        }
    }
    $loginUsuario = $_SESSION["loginUsuario"];
    $output["login"] = $loginUsuario;
    $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
    return $output;
  }


/**
* Efetua a validação dos campos do formulário
   * @see Action::validate()
   * @param object $chatUsuario que será validado
   * @return array $errors
```

```
     */
    public function validate($chatUsuario) {
        $errors = array();

        if (($chatUsuario->getCodChat() == "")) {
            $errors[] = "Deve ser Escolhida uma Sala.";
        }

        if (($chatUsuario->getCodUsuario() == "")) {
            $errors[] = "Deve ser Escolhido um Usuário.";
        }
        $grupo = $chatUsuario->getCodGrupo();
        if (empty($grupo)) {
            $errors[] = "Deve ser Escolhido um Grupo.";
        }

        return $errors;
    }
}
?>
```

13.22. ConfigChatDeleteAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela exclusão de Configurações do Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ConfigChatDeleteAction extends Action {

   /**
    * Efetua o tratamento das requisições. Exclui os dados da Configuração do Chat solicitada através
do ID.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] == ""){
         $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
         <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
         <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
               <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
         </tr></table></body></html>";
         echo $teste;
         exit();
      }

      $ckConfigChat = $_REQUEST["ckConfigChat"];

      foreach ($ckConfigChat as $config) {

         $arrCod = explode("-",$config);

         $codChat = $arrCod[0];
         $codRecurso = $arrCod[1];
         $codGrupo = trim($arrCod[2]);

         ConfigChatManager::delete($codChat, $codRecurso, $codGrupo);
      }

      $configChat = ConfigChatManager::fetchAll($codChat);

      $arrObj = array();
      foreach ($configChat as $config) {
         $arrObj[] = ConfigChatManager::findByKey($config->getCodChat(), $config-
>getCodRecurso(), $config->getCodGrupo());
      }
```

```
        $output = array();
        $output["configChat"] = $configChat;
        $output["arrObj"] = $arrObj;
        $output["chat"] = $codChat;
        $output["cod"] = 1;
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        return $output;
    }

}

?>
```

13.23. ConfigChatListAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela obtenção da lista de Configurações de Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ConfigChatListAction extends Action {

   /**
    * Efetua o tratamento das requisições. Pesquisa a lista de todos as Configurações de um Chat no
Sistema
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] == ""){
         $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
         <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
         <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
               <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
         </tr></table></body></html>";
         echo $teste;
         exit();
      }

      $output = array();
      $lista = ConfigChatManager::fetchAll();
      $arrObj = array();
          foreach ($lista as $config) {
              $arrObj[] = ConfigChatManager::findByKey($config->getCodChat(), $config-
>getCodRecurso(), $config->getCodGrupo() );
          }
      $output["lista"] = $arrObj;
      $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
      $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
      return $output;
   }

}

?>
```

13.24. ConfigChatLoadAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela exibição dos detalhes das Configurações dos Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ConfigChatLoadAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa os dados das Configurações dos Chats solicitado
ou então
     * cria uma nova Configuração para o Chat, de acordo com a solicitação. Estes dados podem ser
alterados e
     * posteriormente salvos com a action de inserção/atualização.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
          $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $output = array();

        $cod = $_REQUEST["cod"];

        $codChat = $_REQUEST["codChat"];

        if ($codChat != 0) {

            $configChat = new ConfigChat();

            $chat = ChatManager::findByKey($codChat);

            $arrObjRecurso = RecursoManager::fetchAll();
            $arrObjGrupo = GrupoManager::fetchAll();

            $output["nomeChat"] = $chat->getNome();
            $output["codChat"] = $codChat;
```

```
            $output["arrObjRecurso"] = $arrObjRecurso;
            $output["arrObjGrupo"] = $arrObjGrupo;
            $output["configChat"] =  $configChat;
            $output["cod"] = $cod;

        }

        else {

            if ((is_numeric($cod)) && ($cod > 0)) {

                $configChat = ConfigChatManager::fetchAll($cod);

                $arrObj = array();
                foreach ($configChat as $config) {
                    $arrObj[] = ConfigChatManager::findByKey($config->getCodChat(), $config-
>getCodRecurso(), $config->getCodGrupo());
                }
                $output["configChat"] = $configChat;
                $output["arrObj"] = $arrObj;
                $output["cod"] = 1;
                $output["chat"] = $cod;
            }
            else {
                $configChat = new ConfigChat();

                $arrObjChat = ChatManager::findByKey($codChat);

                $arrObjRecurso = RecursoManager::fetchAll();
                $arrObjGrupo = GrupoManager::fetchAll();

                $output["arrObjChat"] = $arrObjChat;
                $output["arrObjRecurso"] = $arrObjRecurso;
                $output["arrObjGrupo"] = $arrObjGrupo;
                $output["configChat"] =  $configChat;
                $output["codChat"] = $codChat;

            }

        }
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

        return $output;
    }

}

?>
```

13.25. ConfigChatSaveAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela inserção e atualização de Configurações de Chat do sistema.
 *
 * @see Action
 * @package control
 */
class ConfigChatSaveAction extends Action {

   /**
    * Efetua o tratamento das requisições. Realiza a inserção/atualização de Configurações de Chat
solicitado.
    * Caso a validação das informações do Plataforma falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] == ""){
         $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
         <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
         <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN !</td></tr>"."
               <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
         </tr></table></body></html>";
         echo $teste;
         exit();
      }

      $output = array();

      $cod = $_REQUEST["cod"];

      $listaArr = $_REQUEST["listaArr"];

      if (!empty($listaArr)) {
          $arrObjLista = array();
          foreach ($listaArr as $listConfig) {
             $config = new ConfigChat();
             $config->setCodChat($listConfig["codChat"]);
             $config->setCodRecurso($listConfig["codRecurso"]);
             $config->setCodGrupo($listConfig["codGrupo"]);
             $arrObjLista[] = $config;
          }
      }

      $configChat = new ConfigChat();
```

```
$codChat = $_POST["codChat"];
$nomeChat = $_POST["nomeChat"];

$ckRecurso = $_POST["ckRecurso"];
$ckGrupo = $_POST["ckGrupo"];

if ($codChat != 0) {
    $ckChat[] = $codChat;
}
else {
    $ckChat = $_POST["chat"];
}

if (!empty($ckChat)) {
    $arrObjChat = array();
    foreach ($ckChat as $codC) {
        $chat = new Chat();
        $chat->setCod($codC);
        $arrObjChat[] = $chat;
    }
    $configChat->setCodChat($arrObjChat);
}

if (! empty($ckRecurso)) {
    $arrObjRecurso = array();
    foreach ($ckRecurso as $codRecurso) {
        $recurso = new Recurso();
        $recurso->setCod($codRecurso);
        $arrObjRecurso[] = $recurso;
    }
    $configChat->setCodRecurso($arrObjRecurso);
}

if (! empty($ckGrupo)) {
    $arrObjGrupo = array();
    foreach ($ckGrupo as $codGrupo) {
        if ($codGrupo != "") {
            $grupo = new Grupo();
            $grupo->setCod($codGrupo);
            $arrObjGrupo[] = $grupo;
        }
    }
    $configChat->setCodGrupo($arrObjGrupo);
}

if ($cod != 0){
    $output["cod"] = $cod;
}

$errors = $this->validate($configChat);

if (count($errors) > 0) {

    if ($codChat != 0) {
        $output["codChat"] = $codChat;
        $output["nomeChat"] = $nomeChat;
    }
    else {
        $arrObjChat = ChatManager::fetchAll();
```

```
        $output["arrObjChat"] = $arrObjChat;
    }

    $arrObjRecurso = RecursoManager::fetchAll();
    $arrObjGrupo = GrupoManager::fetchAll();

    if (!empty ($arrObjLista)) {
        $output["lista"] = $arrObjLista;
        $output["chave"] = 1;
    }

    $output["arrObjRecurso"] = $arrObjRecurso;
    $output["arrObjGrupo"] = $arrObjGrupo;
    $output["configChat"] =  $configChat;
    $output[PARAM_ERRORS] = $errors;
    $output[PARAM_ACTION_RESULT] = Action::INPUT;

}
else {

    $error = ConfigChatManager::save($configChat);

    $lista = array();

    if (! empty($arrObjLista)) {
        foreach ($arrObjLista as $config) {
            $lista[] = $config;
        }
    }

    if ($error == 23505) {
                    $msgError = "Você não pode Inserir Registros Duplicados!!!";
                    $output[PARAM_ERRORS] = $msgError;
                }
                else {
        foreach ($ckChat as $cod) {
            foreach ($ckRecurso as $codRecurso) {
                foreach ($ckGrupo as $codGrupo) {
                    $lista[] = ConfigChatManager::findByKey($cod, $codRecurso, $codGrupo);
                }
            }
        }
                }



    if ($codChat != 0) {
        $output["codChat"] = $codChat;
        $output["nomeChat"] = $nomeChat;
    }
    else {
        $arrObjChat = ChatManager::fetchAll();
        $output["arrObjChat"] = $arrObjChat;
    }

    $arrObjRecurso = RecursoManager::fetchAll();
    $arrObjGrupo = GrupoManager::fetchAll();

    $output["arrObjRecurso"] = $arrObjRecurso;
    $output["arrObjGrupo"] = $arrObjGrupo;
```

```
            $output["configChat"] =  $configChat;

            $output["lista"] = $lista;
            $output["chave"] = 1;
            $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        }
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        return $output;
    }


     /**
     * Efetua a validação dos campos do formulário
     * @see Action::validate()
     * @param object $configChat que será validado
     * @return array $errors
     */
    public function validate($configChat) {
        $errors = array();

        if ($configChat->getCodChat() == "") {
            $errors[] = "É necessário escolher pelo menos um Chat.";
        }
        if ($configChat->getCodRecurso() == "") {
            $errors[] = "É necessário escolher pelo menos um Recurso.";
        }

        $grupo = $configChat->getCodGrupo();
        if (empty($grupo)) {
            $errors[] = "É necessário escolher pelo menos um Grupo.";
        }

        return $errors;
    }
}
?>
```

13.26. RecursoDeleteAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela exclusão de Recurso do sistema.
 *
 * @see Action
 * @package control
 */
class RecursoDeleteAction extends Action {

    /**
     * Efetua o tratamento das requisições. Exclui os dados do Recurso solicitado através do Cod.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){

        $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf - O
Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
                  <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $cod = $_REQUEST["cod"];

        RecursoManager::delete($cod);

        $output = array();
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;
    }

}

?>
```

13.27. RecursoListAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela obtenção da lista de Recursos do sistema.
 *
 * @see Action
 * @package control
 */
class RecursoListAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa a lista de todos os Recursos do sistema.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
                $teste = "<html><head><link rel='stylesheet' type='text/css'
href='style.css'/><title>WorldConf - O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
                 <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $output = array();
        $output["lista"] = RecursoManager::fetchAll();
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
        return $output;
    }

}

?>
```

13.28. RecursoLoadAction.class.php

```
<?php

/**
 * Action responsável pela exibição dos detalhes do Recurso do sistema.
 *
 * @see Action
 * @package control
 */
class RecursoLoadAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa os dados do Recurso solicitado ou então
     * cria um novo Recurso, de acordo com a solicitação. Estes dados podem ser alterados e
     * posteriormente salvos com a action de inserção/atualização.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
                $teste = "<html><head><link rel='stylesheet' type='text/css'
href='style.css'/><title>WorldConf - O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
                 <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();
        }

        $cod = $_REQUEST["cod"];
        $body = "Recurso";

        if ((is_numeric($cod)) && ($cod > 0)) {
            $recurso = RecursoManager::findByKey($cod);
        }
        else {
            $recurso = new Recurso();
            $recurso->setCod($cod);
        }

        $output = array();
        $output["recurso"] = $recurso;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

        return $output;
    }

}

?>
```

13.29. RecursoSaveAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela inserção e atualização de Recurso do sistema.
 *
 * @see Action
 * @package control
 */
class RecursoSaveAction extends Action {

   /**
    * Efetua o tratamento das requisições. Realiza a inserção/atualização do Recurso solicitado.
    * Caso a validação das informações do Recurso falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if (($_SESSION["opcao"] != 3) AND ($_SESSION["codUsuario"] != 1)){
            $teste = "<html><head><link rel='stylesheet' type='text/css'
href='style.css'/><title>WorldConf - O Mundo fala aqui!</title></head>"."
        <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
        <tr><td align='center'>VOCÊ NÃO TEM PERMISSÃO PARA ENTRAR NESSA
ÁREA!</td></tr>"."
            <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
        </tr></table></body></html>";
        echo $teste;
        exit();
      }

      $output = array();

      $recurso = new Recurso();
      $recurso->setCod($_REQUEST["cod"]);
      $recurso->setNome(ltrim($_REQUEST["nome"]));

      $errors = $this->validate($recurso);

      if (count($errors) > 0) {

         $output["recurso"] =  $recurso;
         $output[PARAM_ERRORS] = $errors;
         $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
         $output[PARAM_ACTION_RESULT] = Action::INPUT;
      }
      else {
         RecursoManager::save($recurso);
         $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
         $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
```

```
        }

        return $output;
    }

     /**
     * Efetua a validação dos campos do formulário
     * @see Action::validate()
     * @param object $recurso que será validado
     * @return array $errors
     */
    public function validate($recurso) {
        $errors = array();

        if ($recurso->getNome() == "") {
            $errors[] = "O nome deve ser informado.";
        }

        return $errors;
    }

}
?>
```

13.30. RequisicaoChatDeleteAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela exclusão de Requisições para um Chat do sistema.
 *
 * @see Action
 * @package control
 */
class RequisicaoChatDeleteAction extends Action {

    /**
     * Efetua o tratamento das requisições. Exclui os dados das Requisições para um Chat solicitado
     * através do Código do Chat e do Código do Usuário.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
                $teste = "<html><head><link rel='stylesheet' type='text/css'
href='style.css'/><title>WorldConf - O Mundo fala aqui!</title></head>"."
            <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
            <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN!</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
            </tr></table></body></html>";
            echo $teste;
            exit();
        }

        $indice = $_REQUEST["indice"];

        $output = array();

        $cod = $_REQUEST["cod"];

        if ($cod == 1) {

            $codChat = $_REQUEST["codChat"];
            $codUsuario = $_SESSION["codUsuario"];

            $usuarios = RequisicaoChatManager::fetchAll($codChat);

            $arrUsuario = array();

            foreach($usuarios as $usuario){
                $cod = array();
                $aux = explode("-", $usuario->getCodUsuario());
                $cod[] = $aux[0];
                $cod[] = $aux[1];
                $arrUsuario[] = $cod;
```

```
    }

    $output["codChat"] = $codChat;
    $output["arrUsuario"] = $arrUsuario;
    $output[PARAM_ACTION_RESULT] = Action::INPUT;
}
else {

    $codChat = $_REQUEST["codChat"];
    $ckUsuario = $_REQUEST["ckUsuario"];

    $errors = $this->validate($ckUsuario);

    if (count($errors) > 0) {

        $usuarios = RequisicaoChatManager::fetchAll($codChat);

        $arrUsuario = array();

        foreach($usuarios as $usuario){
            $cod = array();
            $aux = explode("-", $usuario->getCodUsuario());
            $cod[] = $aux[0];
            $cod[] = $aux[1];
            $arrUsuario[] = $cod;
        }

        $output["codChat"] = $codChat;
        $output["arrUsuario"] = $arrUsuario;
        $output[PARAM_ERRORS] = $errors;
        $output[PARAM_ACTION_RESULT] = Action::INPUT;
    }
    else {

        $requisicao = new RequisicaoChat();
        $requisicao->setCodChat($codChat);

        foreach ($ckUsuario as $codUsuario) {
            $requisicao->setCodUsuario($codUsuario);
            RequisicaoChatManager::delete($requisicao);
        }

        $usuarios = RequisicaoChatManager::fetchAll($codChat);

        $arrUsuario = array();

        foreach($usuarios as $usuario){
            $cod = array();
            $aux = explode("-", $usuario->getCodUsuario());
            $cod[] = $aux[0];
            $cod[] = $aux[1];
            $arrUsuario[] = $cod;
        }

        $output["codChat"] = $codChat;
        $output["arrUsuario"] = $arrUsuario;
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
    }
}
```

```
        $output["indice"] = 1;
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        return $output;

    }

    /**
     * Efetua a validação dos campos do formulário
     * @see Action::validate()
     * @param mixed $ckUsuario que será validado
     * @return array $errors
     */
    public function validate($ckUsuario) {
        $errors = array();

        if (($ckUsuario == "")) {
            $errors[] = "Deve ser Escolhido um Usuário.";
        }

        return $errors;
    }


}

?>
```

13.31. RequisicaoChatLoadAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela exibição dos detalhes das Requisições de um Chat do sistema.
 *
 * @see Action
 * @package control
 */
class RequisicaoChatLoadAction extends Action {

    /**
     * Efetua o tratamento das requisições. Pesquisa os dados das Requisições de um Chat solicitado
     * de acordo com a solicitação. Estes dados podem ser alterados e
     * posteriormente salvos com a action de inserção/atualização.
     *
     * @return array Retorna os dados resultantes do processamento.
     * @see Action::execute()
     */
    public function execute() {

        session_start();
        if ($_SESSION["loginUsuario"] == ""){
         $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
          <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
          <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN!</td></tr>"."
                <tr><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
          </tr></table></body></html>";
          echo $teste;
          exit();       }

        $codChat = $_REQUEST["cod"];
        $usuarios = RequisicaoChatManager::fetchAll($codChat);

        $arrUsuario = array();

        foreach($usuarios as $usuario){
            $cod = array();
            $aux = explode("-", $usuario->getCodUsuario());
            $cod[] = $aux[0];
            $cod[] = $aux[1];
            $arrUsuario[] = $cod;
        }

        $arrGrupo = GrupoManager::fetchAllConfig($codChat);

        $output = array();
        $output["arrUsuario"] = $arrUsuario;
        $output["arrGrupo"] = $arrGrupo;
        $output["codChat"] = $codChat;
        $output["indice"] = 0;
        $loginUsuario = $_SESSION["loginUsuario"];
```

```
        $output["login"] = $loginUsuario;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];
        $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

        return $output;
    }

}

?>
```

13.32. RequisicaoChatSaveAction.class.php

```
<?php

/**
 * @package control
 */

/**
 * Action responsável pela inserção e atualização de Requisição para um Chat do sistema.
 *
 * @see Action
 * @package control
 */
class RequisicaoChatSaveAction extends Action {

   /**
    * Efetua o tratamento das requisições. Realiza a inserção/atualização de uma Requisição para um
Chat solicitado.
    * Caso a validação das informações do Recurso falhe, retorna o resultado
<kbd>Action::INPUT</kbd>.
    *
    * @return array Retorna os dados resultantes do processamento.
    * @see Action::execute()
    */
   public function execute() {

      session_start();
      if ($_SESSION["loginUsuario"] == ""){

         $teste = "<html><head><link rel='stylesheet' type='text/css' href='style.css'/><title>WorldConf -
O Mundo fala aqui!</title></head>"."
         <body background='/workspace/base/public/imagem/bg.gif'><br><br><table align='center'
border='0' cellpadding='0' cellspacing='0'>"."
         <tr><td align='center'>É NECESSÁRIO EFETUAR O LOGIN!</td></tr>"."
               <tr><br><br><br></tr><tr><td align='center'><a
href='/workspace/base/public/index.html'><br>Clique aqui para Efetuar o Login</a></td>"."
         </tr></table></body></html>";
         echo $teste;
         exit();
      }

      $indice = $_REQUEST["indice"];

      $op = $_REQUEST["op"];

      if ($op == 1) {
         $output = array();

         $requisicao = new RequisicaoChat();
         $requisicao->setCodChat($_REQUEST["cod"]);
         $requisicao->setCodUsuario($_SESSION["codUsuario"]);

         RequisicaoChatManager::save($requisicao);
         $output[PARAM_ACTION_RESULT] = Action::INPUT;
      }
      else{

         $codChat = $_REQUEST["codChat"];
```

```
$ckChat[] = $codChat;
$ckUsuario = $_REQUEST["ckUsuario"];
$ckGrupo = $_REQUEST["ckGrupo"];

$listaArr = $_REQUEST["listaArr"];

if (!empty($listaArr)) {
    $arrObjLista = array();
    foreach ($listaArr as $listAcesso) {
        $acesso = new ChatUsuario();
        $acesso->setCodChat($listAcesso["codChat"]);
        $acesso->setCodUsuario($listAcesso["codUsuario"]);
        $acesso->setCodGrupo($listAcesso["codGrupo"]);
        $arrObjLista[] = $acesso;
    }
}

$chatUsuario = new ChatUsuario();

if (! empty($ckChat)) {
    $arrObjChat = array();
    foreach ($ckChat as $cod) {
        $chat = new Chat();
        $chat->setCod($cod);
        $arrObjChat[] = $chat;
    }
    $chatUsuario->setCodChat($arrObjChat);
}

if (! empty($ckUsuario)) {
    $arrObjUsuario = array();
    foreach ($ckUsuario as $codUsuario) {
        $usuario = new Usuario();
        $usuario->setCod($codUsuario);
        $arrObjUsuario[] = $usuario;
    }
    $chatUsuario->setCodUsuario($arrObjUsuario);
}

if (! empty($ckGrupo)) {
    $arrObjGrupo = array();
    foreach ($ckGrupo as $codGrupo) {
        $grupo = new Grupo();
        if ($codGrupo != "") {
            $grupo->setCod($codGrupo);
            $arrObjGrupo[] = $grupo;
        }
    }
    $chatUsuario->setCodGrupo($arrObjGrupo);
}

$errors = $this->validate($chatUsuario);

$output = array();

if (count($errors) > 0) {

    $usuarios = RequisicaoChatManager::fetchAll($codChat);

    $arrUsuario = array();
```

```
    foreach($usuarios as $usuario){
        $cod = array();
        $aux = explode("-", $usuario->getCodUsuario());
        $cod[] = $aux[0];
        $cod[] = $aux[1];
        $arrUsuario[] = $cod;
    }

    if (!empty ($arrObjLista)) {
        $output["lista"] = $arrObjLista;
        $output["chave"] = 1;
    }

    $arrGrupo = GrupoManager::fetchAllConfig($codChat);

    $output["codChat"] = $codChat;
    $output["arrUsuario"] = $arrUsuario;
    $output["arrGrupo"] = $arrGrupo;
    $output["indice"] = 0;
    $output[PARAM_ERRORS] = $errors;
    $output[PARAM_ACTION_RESULT] = Action::SUCCESS;
}

else {

    ChatUsuarioManager::saveAcesso($chatUsuario, $codChat);

    foreach ($ckChat as $cod) {
        foreach ($ckUsuario as $codUsuario) {
                foreach ($ckGrupo as $codGrupo) {
                    $lista[] = ChatUsuarioManager::findByCod($cod, $codUsuario, $codGrupo);
                    $requisicao = new RequisicaoChat();
                    $requisicao->setCodChat($cod);
                    $requisicao->setCodUsuario($codUsuario);
                    RequisicaoChatManager::delete($requisicao);
                            }
        }
                }

    if (! empty($arrObjLista)) {
        foreach ($arrObjLista as $acesso) {
            $lista[] = $acesso;
        }
    }

    $usuarios = RequisicaoChatManager::fetchAll($codChat);

    $arrUsuario = array();

    foreach($usuarios as $usuario){
        $cod = array();
        $aux = explode("-", $usuario->getCodUsuario());
        $cod[] = $aux[0];
        $cod[] = $aux[1];
        $arrUsuario[] = $cod;
    }

    $arrGrupo = GrupoManager::fetchAllConfig($codChat);
```

```
                $output["codChat"] = $codChat;
                $output["arrUsuario"] = $arrUsuario;
                $output["arrGrupo"] = $arrGrupo;
                $output["lista"] = $lista;
                $output["chave"] = 1;
                $output["indice"] = 0;
                $output[PARAM_ACTION_RESULT] = Action::SUCCESS;

            }

        }
        $loginUsuario = $_SESSION["loginUsuario"];
        $output["login"] = $loginUsuario;
        $output[REQUEST_HEADER_FOOTER] = $_SESSION["opcao"];

        return $output;
    }

    /**
     * Efetua a validação dos campos do formulário
     * @see Action::validate()
     * @param mixed $chatUsuario que será validado
     * @return array $errors
     */
    public function validate($chatUsuario) {
        $errors = array();

        if (($chatUsuario->getCodChat() == "")) {
            $errors[] = "Deve ser Escolhida uma Sala.";
        }

        if (($chatUsuario->getCodUsuario() == "")) {
            $errors[] = "Deve ser Escolhido um Usuário.";
        }
        $grupo = $chatUsuario->getCodGrupo();
        if (empty($grupo)) {
            $errors[] = "Deve ser Escolhido um Grupo.";
        }

        return $errors;
    }

}
?>
```

14. MÓDULO CHAT – CLASSES PACOTE MODEL

14.1. Chat.class.php

```
<?php

/**
 * @package model
 */

/**
 * Classe que define um Chat no sistema.
 *
 * @package model
 */
class Chat {

    /** Código do Chat. */
    private $cod;

    /** Nome do Chat. */
    private $nome;

    /** Descrição do Chat. */
    private $descricao;

    /** Banner do Chat. */
    private $banner;

    /** Data do Início do Chat. */
    private $dtInicio;

    /** Data do Fim do Chat. */
    private $dtFim;

    /** Data de Criação do Chat. */
    private $dtCriacao;

    /** Código do Usuário Criador da Sala. */
    private $codUsuarioChat;

    /** Número de pessoas que estão participando do chat. */
    private $numPessoas;

    /** Número de Requisições para o chat. */
    private $numRequisicoes;

    /**
     * Retorna o valor do campo <kbd>cod_chat</kbd> da tabela chat.
     *
     * @return Retorna o valor do campo <kbd>cod_chat</kbd> da tabela chat.
     */
    public function getCod() {
        return $this->cod;
    }

    /**
     * Armazena o valor do campo <kbd>cod_chat</kbd>.
```

```
     *
     * @param cod O valor a ser armazenado.
     */
    public function setCod($cod) {
        $this->cod = $cod;
    }

    /**
     * Retorna o valor do campo <kbd>nome_chat</kbd>.
     *
     * @return Retorna o valor do campo <kbd>nome_chat</kbd>.
     */
    public function getNome() {
        return $this->nome;
    }

    /**
     * Armazena o valor do campo <kbd>nome_chat</kbd>.
     *
     * @param nome O valor a ser armazenado.
     */
    public function setNome($nome) {
        $this->nome = $nome;
    }

    /**
     * Retorna o valor do campo <kbd>descricao_chat</kbd>.
     *
     * @return Retorna o valor do campo <kbd>descricao_chat</kbd>.
     */
    public function getDescricao() {
        return $this->descricao;
    }

    /**
     * Armazena o valor do campo <kbd>descricao_chat</kbd>.
     *
     * @param descricao O valor a ser armazenado.
     */
    public function setDescricao($descricao) {
        $this->descricao = $descricao;
    }

    /**
     * Retorna o valor do campo <kbd>banner</kbd>.
     *
     * @return Retorna o valor do campo <kbd>banner</kbd>.
     */
    public function getBanner() {
        return $this->banner;
    }

    /**
     * Armazena o valor do campo <kbd>banner</kbd>.
     *
     * @param banner O valor a ser armazenado.
     */
    public function setBanner($banner) {
        $this->banner = $banner;
    }
```

```
    /**
     * Retorna o valor do campo <kbd>dt_inicio_chat</kbd>.
     *
     * @return Retorna o valor do campo <kbd>dt_inicio_chat</kbd>.
     */
    public function getDtInicio() {
        return $this->dtInicio;
    }

    /**
     * Armazena o valor do campo <kbd>dt_inicio_chat</kbd>.
     *
     * @param dt_criacao O valor a ser armazenado.
     */
    public function setDtInicio($dtInicio) {
        $this->dtInicio = $dtInicio;
    }

/**
     * Retorna o valor do campo <kbd>dt_fim_chat</kbd>.
     *
     * @return Retorna o valor do campo <kbd>dt_fim_chat</kbd>.
     */
    public function getDtFim() {
        return $this->dtFim;
    }

    /**
     * Armazena o valor do campo <kbd>dt_fim_chat</kbd>.
     *
     * @param dt_criacao O valor a ser armazenado.
     */
    public function setDtFim($dtFim) {
        $this->dtFim = $dtFim;
    }

    /**
     * Retorna o valor do campo <kbd>dt_criacao_chat</kbd>.
     *
     * @return Retorna o valor do campo <kbd>dt_criacao_chat</kbd>.
     */
    public function getDtCriacao() {
        return $this->dtCriacao;
    }

    /**
     * Armazena o valor do campo <kbd>dt_criacao_chat</kbd>.
     *
     * @param dt_criacao O valor a ser armazenado.
     */
    public function setDtCriacao($dtCriacao) {
        $this->dtCriacao = $dtCriacao;
    }

    /**
     * Retorna o valor do campo <kbd>cod_usuario_chat</kbd>.
     *
     * @return Retorna o valor do campo <kbd>cod_usuario_chat</kbd>.
     */
```

```
    public function getCodUsuarioChat() {
        return $this->codUsuarioChat;
    }

    /**
     * Armazena o valor do campo <kbd>cod_usuario_chat</kbd>.
     *
     * @param dt_criacao O valor a ser armazenado.
     */
    public function setCodUsuarioChat($codUsuarioChat) {
        $this->codUsuarioChat = $codUsuarioChat;
    }

    /**
     * Retorna o valor do campo <kbd>numPessoas</kbd>.
     *
     * @return Retorna o valor do campo <kbd>numPessoas</kbd>.
     */
    public function getNumPessoas() {
        return $this->numPessoas;
    }

    /**
     * Armazena o valor do campo <kbd>numPessoas</kbd>.
     *
     * @param numPessoas O valor a ser armazenado.
     */
    public function setNumPessoas($numPessoas) {
        $this->numPessoas = $numPessoas;
    }

    /**
     * Retorna o valor do campo <kbd>numRequisicoes</kbd>.
     *
     * @return Retorna o valor do campo <kbd>numRequisicoes</kbd>.
     */
    public function getNumRequisicoes() {
        return $this->numRequisicoes;
    }

    /**
     * Armazena o valor do campo <kbd>numRequisicoes</kbd>.
     *
     * @param numPessoas O valor a ser armazenado.
     */
    public function setNumRequisicoes($numRequisicoes) {
        $this->numRequisicoes = $numRequisicoes;
    }

}

?>
```

14.2. ChatMensagem.class.php

```
<?php

/**
 * @package model
 */

/**
 * Classe que define uma Mensagem em um Chat no sistema.
 *
 * @package model
 */
class ChatMensagem {

    /** Código da Mensagem no Chat. */
    private $cod;

    /** Tempo da Mensagem em segundos no Chat. */
    private $time;

    /** Remetente da Mensagem. */
    private $remetente;

    /** Destinatário da Mensagem */
    private $destinatario;

    /** Conteúdo da Mensagem */
    private $mensagem;

    /** Som da Mensagem*/
    private $som;

    /** Status da mensagem (reservada ou pública)*/
    private $status;

    /** Código do Usuário*/
    private $codUsuario;

    /** Código do Chat*/
    private $codChat;

    /**
     * Retorna o valor do campo <kbd>codigo</kbd> da tabela mensagens.
     *
     * @return Retorna o valor do campo <kbd>codigo</kbd> da tabela mensagens.
     */
    public function getCod() {
        return $this->cod;
    }

    /**
     * Armazena o valor do campo <kbd>codigo</kbd>.
     *
     * @param cod O valor a ser armazenado.
     */
    public function setCod($cod) {
        $this->cod = $cod;
    }
```

```
/**
 * Retorna o valor do campo <kbd>time</kbd>.
 *
 * @return Retorna o valor do campo <kbd>time</kbd>.
 */
public function getTime() {
    return $this->time;
}

/**
 * Armazena o valor do campo <kbd>time</kbd>.
 *
 * @param time O valor a ser armazenado.
 */
public function setTime($time) {
    $this->time = $time;
}

/**
 * Retorna o valor do campo <kbd>remetente</kbd>.
 *
 * @return Retorna o valor do campo <kbd>remetente</kbd>.
 */
public function getRemetente() {
    return $this->remetente;
}

/**
 * Armazena o valor do campo <kbd>remetente</kbd>.
 *
 * @param ip O valor a ser armazenado.
 */
public function setRemetente($remetente) {
    $this->remetente = $remetente;
}

/**
 * Retorna o valor do campo <kbd>destinatário</kbd>.
 *
 * @return Retorna o valor do campo <kbd>destinatário</kbd>.
 */
public function getDestinatario() {
    return $this->destinatario;
}

/**
 * Armazena o valor do campo <kbd>destinatário</kbd>.
 *
 * @param nome O valor a ser armazenado.
 */
public function setDestinatario($destinatario) {
    $this->destinatario = $destinatario;
}

/**
 * Retorna o valor do campo <kbd>mensagem</kbd>.
 *
 * @return Retorna o valor do campo <kbd>mensagem</kbd>.
 */
public function getMensagem() {
```

```
    return $this->mensagem;
}

/**
 * Armazena o valor do campo <kbd>mensagem</kbd>.
 *
 * @param mensagem O valor a ser armazenado.
 */
public function setMensagem($mensagem) {
    $this->mensagem = $mensagem;
}

/**
 * Retorna o valor do campo <kbd>som</kbd>.
 *
 * @return Retorna o valor do campo <kbd>som</kbd>.
 */
public function getSom() {
    return $this->som;
}

/**
 * Armazena o valor do campo <kbd>som</kbd>.
 *
 * @param som O valor a ser armazenado.
 */
public function setSom($som) {
    $this->som = $som;
}

/**
 * Retorna o valor do campo <kbd>status</kbd>.
 *
 * @return Retorna o valor do campo <kbd>status</kbd>.
 */
public function getStatus() {
    return $this->status;
}

/**
 * Armazena o valor do campo <kbd>status</kbd>.
 *
 * @param status O valor a ser armazenado.
 */
public function setStatus($status) {
    $this->status = $status;
}

/**
 * Retorna o valor do campo <kbd>codUsuario</kbd>.
 *
 * @return Retorna o valor do campo <kbd>codUsuario</kbd>.
 */
public function getCodUsuario() {
    return $this->codUsuario;
}

/**
 * Armazena o valor do campo <kbd>codUsuario</kbd>.
 *
```

```
     * @param codUsuario O valor a ser armazenado.
     */
    public function setCodUsuario($codUsuario) {
        $this->codUsuario = $codUsuario;
    }

    /**
     * Retorna o valor do campo <kbd>codChat</kbd>.
     *
     * @return Retorna o valor do campo <kbd>codChat</kbd>.
     */
    public function getCodChat() {
        return $this->codChat;
    }

    /**
     * Armazena o valor do campo <kbd>codChat</kbd>.
     *
     * @param codChat O valor a ser armazenado.
     */
    public function setCodChat($codChat) {
        $this->codChat = $codChat;
    }

}

?>
```

14.3. ChatUsuario.class.php

```
<?php

/**
 * @package model
 */

/**
 * Classe que define um Usuário em um Chat no sistema.
 *
 * @package model
 */
class ChatUsuario {

    /** Código do Chat. */
    private $codChat;

    /** Código do Usuário. */
    private $codUsuario;

    /** Código do Grupo. */
    private $codGrupo;

    /** Apelido do Usuário no Chat. */
    private $nome;

    /** Cor do Apelido do Usuário no Chat. */
    private $cor;

    /** Código da última mensagem enviada pelo Usuário no Chat. */
    private $last;

    /** Status do Usuário no Chat (0 = Offline, 1 = Online). */
    private $status;

    /**
     * Retorna o valor do campo <kbd>chat_cod_chat</kbd> da tabela acesso_chat.
     *
     * @return Retorna o valor do campo <kbd>chat_cod_chat</kbd> da tabela acesso_cjat.
     */
    public function getCodChat() {
        return $this->codChat;
    }

    /**
     * Armazena o valor do campo <kbd>chat_cod_chat</kbd>.
     *
     * @param cod O valor a ser armazenado.
     */
    public function setCodChat($codChat) {
        $this->codChat = $codChat;
    }

    /**
     * Retorna o valor do campo <kbd>usuario_cod_usuario</kbd> da tabela acesso_chat.
     *
     * @return Retorna o valor do campo <kbd>usuario_cod_usuario</kbd> da tabela acesso_chat.
     */
    public function getCodUsuario() {
```

```
    return $this->codUsuario;
}

/**
 * Armazena o valor do campo <kbd>usuario_cod_usuario</kbd>.
 *
 * @param cod O valor a ser armazenado.
 */
public function setCodUsuario($codUsuario) {
    $this->codUsuario = $codUsuario;
}

/**
 * Retorna o valor do campo <kbd>grupo_cod_grupo</kbd> da tabela acesso_chat.
 *
 * @return Retorna o valor do campo <kbd>grupo_cod_grupo</kbd> da tabela acesso_chat.
 */
public function getCodGrupo() {
    return $this->codGrupo;
}

/**
 * Armazena o valor do campo <kbd>grupo_cod_grupo</kbd>.
 *
 * @param cod O valor a ser armazenado.
 */
public function setCodGrupo($codGrupo) {
    $this->codGrupo = $codGrupo;
}

/**
 * Retorna o valor do campo <kbd>nome</kbd>.
 *
 * @return Retorna o valor do campo <kbd>nome</kbd>.
 */
public function getNome() {
    return $this->nome;
}

/**
 * Armazena o valor do campo <kbd>nome</kbd>.
 *
 * @param nome O valor a ser armazenado.
 */
public function setNome($nome) {
    $this->nome = $nome;
}

/**
 * Retorna o valor do campo <kbd>cor</kbd>.
 *
 * @return Retorna o valor do campo <kbd>cor</kbd>.
 */
public function getCor() {
    return $this->cor;
}

/**
 * Armazena o valor do campo <kbd>cor</kbd>.
 *
```

```
     * @param cor O valor a ser armazenado.
     */
    public function setCor($cor) {
        $this->cor = $cor;
    }

    /**
     * Retorna o valor do campo <kbd>last</kbd>.
     *
     * @return Retorna o valor do campo <kbd>last</kbd>.
     */
    public function getLast() {
        return $this->last;
    }

    /**
     * Armazena o valor do campo <kbd>last</kbd>.
     *
     * @param last O valor a ser armazenado.
     */
    public function setLast($last) {
        $this->last = $last;
    }

    /**
     * Retorna o valor do campo <kbd>status</kbd>.
     *
     * @return Retorna o valor do campo <kbd>status</kbd>.
     */
    public function getStatus() {
        return $this->status;
    }

    /**
     * Armazena o valor do campo <kbd>status</kbd>.
     *
     * @param status O valor a ser armazenado.
     */
    public function setStatus($status) {
        $this->status = $status;
    }


}

?>
```

14.4. ConfigChat.class.php

```
<?php

/**
 * @package model
 */

/**
 * Classe que define uma Configuração de Chat no sistema.
 *
 * @package model
 */
class ConfigChat {

    /** Codigo do Chat. */
    private $codChat;

    /** Codigo do Recurso. */
    private $codRecurso;

    /** Codigo do Grupo. */
    private $codGrupo;


    /**
     * Retorna o valor do campo <kbd>chat_cod_chat</kbd> da tabela config_chat.
     *
     * @return Retorna o valor do campo <kbd>chat_cod_chat</kbd> da tabela config_chat.
     */
    public function getCodChat() {
        return $this->codChat;
    }

    /**
     * Armazena o valor do campo <kbd>chat_cod_chat</kbd>.
     *
     * @param cod O valor a ser armazenado.
     */
    public function setCodChat($codChat) {
        $this->codChat = $codChat;
    }

    /**
     * Retorna o valor do campo <kbd>recurso_cod_recurso</kbd> da tabela config_chat.
     *
     * @return Retorna o valor do campo <kbd>recurso_cod_recurso</kbd> da tabela config_chat.
     */
    public function getCodRecurso() {
        return $this->codRecurso;
    }

    /**
     * Armazena o valor do campo <kbd>recurso_cod_recurso</kbd>.
     *
     * @param cod O valor a ser armazenado.
     */
    public function setCodRecurso($codRecurso) {
        $this->codRecurso = $codRecurso;
    }
```

```
    /**
     * Retorna o valor do campo <kbd>grupo_cod_grupo</kbd> da tabela config_chat.
     *
     * @return Retorna o valor do campo <kbd>grupo_cod_grupo</kbd> da tabela config_chat.
     */
    public function getCodGrupo() {
        return $this->codGrupo;
    }

    /**
     * Armazena o valor do campo <kbd>grupo_cod_grupo</kbd>.
     *
     * @param cod O valor a ser armazenado.
     */
    public function setCodGrupo($codGrupo) {
        $this->codGrupo = $codGrupo;
    }

}
?>
```

14.5. Recurso.class.php

```
<?php

/**
 * Classe que define um Recurso no sistema.
 *
 * @package model
 */
class Recurso {

    /** Código do Recurso. */
    private $cod;

    /** Nome do Recurso. */
    private $nome;


    /**
     * Retorna o valor do campo <kbd>cod_recurso</kbd> da tabela recurso.
     *
     * @return Retorna o valor do campo <kbd>cod_recurso</kbd> da tabela recurso.
     */
    public function getCod() {
        return $this->cod;
    }

    /**
     * Armazena o valor do campo <kbd>cod_recurso</kbd>.
     *
     * @param cod O valor a ser armazenado.
     */
    public function setCod($cod) {
        $this->cod = $cod;
    }

    /**
     * Retorna o valor do campo <kbd>nome_recurso</kbd>.
     *
     * @return Retorna o valor do campo <kbd>nome_recurso</kbd>.
     */
    public function getNome() {
        return $this->nome;
    }

    /**
     * Armazena o valor do campo <kbd>nome_recurso</kbd>.
     *
     * @param nome O valor a ser armazenado.
     */
    public function setNome($nome) {
        $this->nome = $nome;
    }

}

?>
```

14.6. RequisicaoChat.class.php

```
<?php

/**
 * @package model
 */

/**
 * Classe que define um Requisição de Chat no sistema.
 *
 * @package model
 */
class RequisicaoChat {

    /** Código do Chat. */
    private $codChat;

    /** Nome do Usuário. */
    private $codUsuario;

    /**
     * Retorna o valor do campo <kbd>cod_chat</kbd> da tabela requisicao_chat.
     *
     * @return Retorna o valor do campo <kbd>cod_chat</kbd> da tabela requisicao_chat.
     */
    public function getCodChat() {
        return $this->codChat;
    }

    /**
     * Armazena o valor do campo <kbd>cod_chat</kbd>.
     *
     * @param cod O valor a ser armazenado.
     */
    public function setCodChat($codChat) {
        $this->codChat = $codChat;
    }

    /**
     * Retorna o valor do campo <kbd>cod_usuario</kbd>.
     *
     * @return Retorna o valor do campo <kbd>cod_usuario</kbd>.
     */
    public function getCodUsuario() {
        return $this->codUsuario;
    }

    /**
     * Armazena o valor do campo <kbd>cod_usuario</kbd>.
     *
     * @param nome O valor a ser armazenado.
     */
    public function setCodUsuario($codUsuario) {
        $this->codUsuario = $codUsuario;
    }

}

?>
```

15. MÓDULO CHAT – CLASSES PACOTE MODEL / PERSISTENCE

15.1. ChatManager.class.php

```
<?php
/**
 * @package model
 */

/**
 * Classe que define o gerenciador de persistência para a classe <kbd>Chat</kbd>.
 *
 * @static
 * @see ChatPersistenceManager
 * @package model
 */
class ChatManager implements ChatPersistenceManager {

    /**
     * @see ChatPersistenceManager::fetch()
     */
    public static function fetch() {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT a.cod_chat, a.nome_chat, a.descricao_chat, a.dt_inicio_chat,
a.dt_fim_chat, a.dt_criacao_chat, COUNT(b.status) as numpessoas FROM chat a LEFT JOIN
acesso_chat b ON a.cod_chat = b.chat_cod_chat AND b.status = 1 GROUP BY a.cod_chat,
a.nome_chat, a.descricao_chat, a.dt_inicio_chat, a.dt_fim_chat, a.dt_criacao_chat ORDER BY
a.dt_criacao_chat DESC");

        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {
            $c = new Chat();
            $c->setCod($row["cod_chat"]);
            $c->setNome($row["nome_chat"]);
            $c->setDescricao($row["descricao_chat"]);
            $c->setBanner($row["banner_chat"]);
            $c->setDtInicio($row["dt_inicio_chat"]);
            $c->setDtFim($row["dt_fim_chat"]);
            $c->setDtCriacao($row["dt_criacao_chat"]);
            $c->setNumPessoas($row["numpessoas"]);

            $stmt1 = $pdo->prepare("SELECT COUNT(b.cod_chat) as total FROM requisicao_chat b
WHERE b.cod_chat = :codChat");
            $stmt1->bindParam(":codChat", $row["cod_chat"]);
            $stmt1->execute();

            $row1 = $stmt1->fetch();

            $c->setNumRequisicoes($row1["total"]);

            $l[] = $c;
        }

        return $l;
```

```
    }


    /**
     * @see ChatPersistenceManager::fetchAll()
     */
    public static function fetchAll($codUsuario) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT a.cod_chat, a.nome_chat, a.descricao_chat, a.dt_inicio_chat,
a.dt_fim_chat, a.dt_criacao_chat, COUNT(b.status) as numpessoas FROM chat a LEFT JOIN
acesso_chat b ON a.cod_chat = b.chat_cod_chat AND b.status = 1 WHERE a.cod_chat IN (SELECT
b.chat_cod_chat FROM acesso_chat b, chat a WHERE b.usuario_cod_usuario = :codUsuario AND
a.cod_chat = b.chat_cod_chat AND a.cod_usuario_chat NOT IN(SELECT a.cod_usuario_chat FROM
chat a WHERE a.cod_usuario_chat = :codUsuario )) GROUP BY a.cod_chat, a.nome_chat,
a.descricao_chat, a.dt_inicio_chat, a.dt_fim_chat, a.dt_criacao_chat ORDER BY a.dt_criacao_chat
DESC");

        $stmt->bindParam(":codUsuario", $codUsuario);

        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {
            $c = new Chat();
            $c->setCod($row["cod_chat"]);
            $c->setNome($row["nome_chat"]);
            $c->setDescricao($row["descricao_chat"]);
            $c->setBanner($row["banner_chat"]);
            $c->setDtInicio($row["dt_inicio_chat"]);
            $c->setDtFim($row["dt_fim_chat"]);
            $c->setDtCriacao($row["dt_criacao_chat"]);
            $c->setNumPessoas($row["numpessoas"]);
            $l[] = $c;
        }

        return $l;

    }

    /**
     * @see ChatPersistenceManager::fetchAllConfig()
     */
    public static function fetchAllConfig($codUsuario) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT a.cod_chat, a.nome_chat, a.descricao_chat, a.dt_inicio_chat,
a.dt_fim_chat, a.dt_criacao_chat, COUNT(b.status) as numpessoas FROM chat a LEFT JOIN
acesso_chat b ON a.cod_chat = b.chat_cod_chat AND b.status = 1 WHERE a.cod_chat IN (SELECT
cod_chat FROM chat WHERE cod_usuario_chat = :codUsuario) GROUP BY a.cod_chat,
a.nome_chat, a.descricao_chat, a.dt_inicio_chat, a.dt_fim_chat, a.dt_criacao_chat ORDER BY
a.dt_criacao_chat DESC");

        $stmt->bindParam(":codUsuario", $codUsuario);

        $stmt->execute();
```

```
        $l = array();
        while ($row = $stmt->fetch()) {
            $c = new Chat();
            $c->setCod($row["cod_chat"]);
            $c->setNome($row["nome_chat"]);
            $c->setDescricao($row["descricao_chat"]);
            $c->setBanner($row["banner_chat"]);
            $c->setDtInicio($row["dt_inicio_chat"]);
            $c->setDtFim($row["dt_fim_chat"]);
            $c->setDtCriacao($row["dt_criacao_chat"]);
            $c->setNumPessoas($row["numpessoas"]);

            $stmt1 = $pdo->prepare("SELECT COUNT(b.cod_chat) as total FROM requisicao_chat b
WHERE b.cod_chat = :codChat");
            $stmt1->bindParam(":codChat", $row["cod_chat"]);
            $stmt1->execute();

            $row1 = $stmt1->fetch();

            $c->setNumRequisicoes($row1["total"]);
            $l[] = $c;
        }

        return $l;

    }

    /**
     * @see ChatPersistenceManager::fetchList()
     */
    public static function fetchList($codUsuario) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT a.cod_chat, a.nome_chat, a.descricao_chat, a.dt_inicio_chat,
a.dt_fim_chat, a.dt_criacao_chat, COUNT(b.status) as numpessoas FROM chat a LEFT JOIN
acesso_chat b ON a.cod_chat = b.chat_cod_chat AND b.status = 1 WHERE a.cod_chat NOT IN
(SELECT r.cod_chat FROM acesso_chat b, chat a, requisicao_chat r WHERE r.cod_usuario =
:codUsuario AND a.cod_chat = b.chat_cod_chat) AND a.cod_chat NOT IN (SELECT b.chat_cod_chat
FROM acesso_chat b, chat a WHERE b.usuario_cod_usuario = :codUsuario AND a.cod_chat =
b.chat_cod_chat) GROUP BY a.cod_chat, a.nome_chat, a.descricao_chat, a.dt_inicio_chat,
a.dt_fim_chat, a.dt_criacao_chat ORDER BY a.dt_criacao_chat DESC");

        $stmt->bindParam(":codUsuario", $codUsuario);

        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {
            $c = new Chat();
            $c->setCod($row["cod_chat"]);
            $c->setNome($row["nome_chat"]);
            $c->setDescricao($row["descricao_chat"]);
            $c->setBanner($row["banner_chat"]);
            $c->setDtInicio($row["dt_inicio_chat"]);
            $c->setDtFim($row["dt_fim_chat"]);
            $c->setDtCriacao($row["dt_criacao_chat"]);
            $c->setNumPessoas($row["numpessoas"]);
            $l[] = $c;
        }
```

```
        return $l;

    }

    /**
     * @see PersistenceManager::fetchListMod()
     */
    public static function fetchListMod($codUsuario) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT a.cod_chat, a.nome_chat, a.descricao_chat, a.dt_inicio_chat,
a.dt_fim_chat, a.dt_criacao_chat, COUNT(b.status) as numpessoas FROM chat a LEFT JOIN
acesso_chat b ON a.cod_chat = b.chat_cod_chat AND b.status = 1 WHERE a.cod_chat IN (SELECT
r.cod_chat FROM acesso_chat b, chat a, requisicao_chat r WHERE r.cod_usuario = :codUsuario AND
a.cod_chat = b.chat_cod_chat) AND a.cod_chat NOT IN (SELECT b.chat_cod_chat FROM
acesso_chat b, chat a WHERE b.usuario_cod_usuario = :codUsuario AND a.cod_chat =
b.chat_cod_chat) GROUP BY a.cod_chat, a.nome_chat, a.descricao_chat, a.dt_inicio_chat,
a.dt_fim_chat, a.dt_criacao_chat ORDER BY a.dt_criacao_chat DESC");

        $stmt->bindParam(":codUsuario", $codUsuario);

        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {
            $c = new Chat();
            $c->setCod($row["cod_chat"]);
            $c->setNome($row["nome_chat"]);
            $c->setDescricao($row["descricao_chat"]);
            $c->setBanner($row["banner_chat"]);
            $c->setDtInicio($row["dt_inicio_chat"]);
            $c->setDtFim($row["dt_fim_chat"]);
            $c->setDtCriacao($row["dt_criacao_chat"]);
            $c->setNumPessoas($row["numpessoas"]);
            $l[] = $c;
        }

        return $l;

    }


    /**
     * @see PersistenceManager::findByKey()
     */
    public static function findByKey($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT cod_chat, nome_chat, descricao_chat, banner_chat,
dt_inicio_chat, dt_fim_chat, cod_usuario_chat FROM chat WHERE cod_chat = :cod");

        $stmt->bindParam(":cod", $key);

        $stmt->execute();

        if ($row = $stmt->fetch()) {
            $chat = new Chat();
```

```
            $chat->setCod($key);
            $chat->setNome($row["nome_chat"]);
            $chat->setDescricao($row["descricao_chat"]);
            $chat->setBanner($row["banner_chat"]);
            $chat->setDtInicio($row["dt_inicio_chat"]);
            $chat->setDtFim($row["dt_fim_chat"]);
            $chat->setCodUsuarioChat($row["cod_usuario_chat"]);
        } else {
            $chat = new Chat();
            $chat->setCod(-1);
        }

        return $chat;

    }

    /**
     * @see ChatPersistenceManager::save()
     */
    public static function save($chat) {

        $pdo = Util::getPDOHandler();

        try {

            $pdo->beginTransaction();

            if ($chat->getCod() == 0) {
                $sql = "INSERT INTO chat (nome_chat, descricao_chat, banner_chat, dt_inicio_chat,
dt_fim_chat, cod_usuario_chat) VALUES (:nome, :descricao, :banner, :dtInicio, :dtFim, :codUsuario)";
            }
            else {
                $sql = "UPDATE chat SET nome_chat = :nome, descricao_chat = :descricao, banner_chat =
:banner, dt_inicio_chat = :dtInicio, dt_fim_chat = :dtFim WHERE cod_chat = :cod";
            }

            $stmt = $pdo->prepare($sql);

            if ($chat->getCod() == 0) {
                $stmt->execute(array(":nome" => $chat->getNome(), ":descricao" => $chat->getDescricao(),
":banner" => $chat->getBanner(), ":dtInicio" => $chat->getDtInicio(), ":dtFim" => $chat->getDtFim(),
":codUsuario" => $chat->getCodUsuarioChat()));
            }
            else {
                $stmt->execute(array(":cod" => $chat->getCod(), ":nome" => $chat->getNome(),
":descricao" => $chat->getDescricao(), ":banner" => $chat->getBanner(), ":dtInicio" => $chat-
>getDtInicio(), ":dtFim" => $chat->getDtFim()));
            }

            if ($chat->getCod() == 0) {
                $stmt1 = $pdo->prepare("SELECT MAX(cod_chat) FROM chat");
                $stmt1->execute();

                if ($row = $stmt1->fetch()) {
                  $codChat = $row["max"];
                }

                $stmt2 = $pdo->prepare("SELECT cod_grupo FROM grupo WHERE nome_grupo =
'Criador'");
                $stmt2->execute();
```

```
            if ($row = $stmt2->fetch()) {
               $codGrupo = $row["cod_grupo"];
            }

            $arrRecurso = RecursoManager::fetchAll();

            foreach ($arrRecurso as $recurso) {
               $codRecurso = $recurso->getCod();
               $stmt3 = $pdo->prepare("INSERT INTO config_chat (chat_cod_chat,
recurso_cod_recurso, grupo_cod_grupo) VALUES (:codChat, :codRecurso, :codGrupo)");
               $stmt3->bindParam(":codChat", $codChat);
               $stmt3->bindParam(":codRecurso", $codRecurso);
               $stmt3->bindParam(":codGrupo", $codGrupo);
               $stmt3->execute();
            }

            $codUsuario = $chat->getCodUsuarioChat();

            if ($codUsuario != 1) {
               $stmt4 = $pdo->prepare("INSERT INTO acesso_chat (chat_cod_chat,
usuario_cod_usuario, grupo_cod_grupo) VALUES (:codChat, :codUsuario, :codGrupo)");
               $stmt4->bindParam(":codChat", $codChat);
               $stmt4->bindParam(":codUsuario", $codUsuario);
               $stmt4->bindParam(":codGrupo", $codGrupo);
               $stmt4->execute();
            }

            $codUsuario = 1;
            $stmt5= $pdo->prepare("INSERT INTO acesso_chat (chat_cod_chat, usuario_cod_usuario,
grupo_cod_grupo) VALUES (:codChat, :codUsuario, :codGrupo)");
            $stmt5->bindParam(":codChat", $codChat);
            $stmt5->bindParam(":codUsuario", $codUsuario);
            $stmt5->bindParam(":codGrupo", $codGrupo);
            $stmt5->execute();
         }

         $pdo->commit();

      } catch (Exception $e) {
         $pdo->rollBack();
      }

      return $codChat;

   }

   /**
    * @see ChatPersistenceManager::delete()
    */
   public static function delete($key) {

      $pdo = Util::getPDOHandler();

      try {

         $pdo->beginTransaction();

         $stmt = $pdo->prepare("DELETE FROM requisicao_chat WHERE cod_chat = :cod");
         $stmt->bindParam(":cod", $key);
```

```
            $stmt->execute();

            $stmt3 = $pdo->prepare("DELETE FROM acesso_chat WHERE chat_cod_chat = :cod");
            $stmt3->bindParam(":cod", $key);
            $stmt3->execute();

            $stmt1 = $pdo->prepare("DELETE FROM config_chat WHERE chat_cod_chat = :cod");
            $stmt1->bindParam(":cod", $key);
            $stmt1->execute();

            $stmt2 = $pdo->prepare("DELETE FROM chat WHERE cod_chat = :cod");
            $stmt2->bindParam(":cod", $key);
            $stmt2->execute();

            if (file_exists("../modulos/chat/jnlp/$key/cliente.jar"))
                unlink("../modulos/chat/jnlp/$key/cliente.jar");
            if (file_exists("../modulos/chat/jnlp/$key/audio.jnlp"))
                unlink("../modulos/chat/jnlp/$key/audio.jnlp");
            if (file_exists("../modulos/chat/jnlp/$key"))
                rmdir("../modulos/chat/jnlp/$key");

            $pdo->commit();

        } catch (Exception $e) {
            $pdo->rollBack();
        }

    }

}

?>
```

15.2. ChatPersistenceManager.class.php

```
<?php

/**
 * @package model
 */

/**
 * Interface que define um Manager, classe responsável pelo gerenciamento
 * de persistência de alguma classe do Model. Todos os Managers deste módulo
 * devem implementar no mínimo os métodos definidos por este contrato.
 *
 * Essa idéia segue o design pattern DAO (Data Access Object).
 *
 * @static
 * @package model
 */
interface ChatPersistenceManager {

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetch();

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @param mixed $codUsuario Código do Usuário das entidades desejadas, segundo a definição
de chave primária da classe Model.
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchAll($codUsuario);

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @param mixed $codUsuario Código do Usuário das entidades desejadas, segundo a definição
de chave primária da classe Model.
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchAllConfig($codUsuario);

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @param mixed $codUsuario Código do Usuário das entidades desejadas, segundo a definição
de chave primária da classe Model.
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchList($codUsuario);

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
```

```
    *
    * @param mixed $codUsuario Código do Usuário das entidades desejadas, segundo a definição de chave primária da classe Model.
    * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
    * haver nenhum objeto, <kbd>array</kbd> estará vazio.
    */
   public static function fetchListMod($codUsuario);

   /**
    * Método utilizado para obtenção de uma entidade da classe Model correspondente através de sua chave primária.
    *
    * @param mixed $key Chave da entidade desejada, segundo a definição de chave primária da classe Model.
    * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
    */
   public static function findByKey($key);

   /**
    * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica da operação,
    * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser especificada na
    * implementação do método pela classe concreta. O exemplo mais comum para chave primária inteira é assumir
    * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser inserida)
    * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas atualizada).
    *
    * @param mixed $model Objeto a ser gravado.
    */
   public static function save($chat);

   /**
    * Método utilizado para exclusão de uma entidade da classe Model correspondente através de sua chave primária.
    *
    * @param mixed $key Chave da entidade a ser excluída, segundo a definição de chave primária da classe Model.
    */
   public static function delete($key);

}

?>
```

15.3. ChatMensagemManager.class.php

```
<?php
/**
 * @package model
 */

/**
 * Classe que define o gerenciador de persistência para a classe <kbd>ChatMensagem</kbd>.
 *
 * @static
 * @see ChatMensagemPersistenceManager
 * @package model
 */
class ChatMensagemManager implements ChatMensagemPersistenceManager {

    /**
     * @see ChatMensagemPersistenceManager::save()
     */
    public static function save($mensagem) {

        $pdo = Util::getPDOHandler();

        $sql = "INSERT INTO chat_mensagem (time, remetente, destinatario, mensagem, som, status,
cod_chat, cod_usuario) VALUES (:time, :remetente, :destinatario, :mensagem, :som, :status,
:codChat, :codUsuario)";

        $stmt = $pdo->prepare($sql);

        $stmt->execute(array(":time" => $mensagem->getTime(), ":remetente" => $mensagem-
>getRemetente(), ":destinatario" => $mensagem->getDestinatario(), ":mensagem" => $mensagem-
>getMensagem(), ":som" => $mensagem->getSom(), ":status" => $mensagem->getStatus(),
":codChat" => $mensagem->getCodChat(), ":codUsuario" => $mensagem->getCodUsuario()));
    }

    /**
     * @see ChatMensagemPersistenceManager::delete()
     */
    public static function delete($codChat) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT time FROM chat_mensagem WHERE cod_chat = :codChat");

        $stmt->bindParam(":codChat", $codChat);

        $stmt->execute();

        $timeout = time() - 60;

        while ($row = $stmt->fetch()) {

            if ($row["time"] < $timeout) {

                $stmt1 = $pdo->prepare("DELETE FROM chat_mensagem WHERE time = :time AND
cod_chat = :codChat");

                $stmt1->bindParam(":time", $row["time"]);

                $stmt1->bindParam(":codChat", $codChat);
```

```
                $stmt1->execute();
            }
        }

    }

    /**
     * @see ChatMensagemPersistenceManager::imprimir()
     */
    public static function imprimir($nome, &$last, $codChat) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT cod_mensagem, remetente, destinatario, mensagem, som,
status, cod_usuario, cod_chat FROM chat_mensagem WHERE cod_chat=:codChat ORDER BY
cod_mensagem ASC");

        $stmt->bindParam(":codChat", $codChat);

        $stmt->execute();

        $msg = array();
        $msg[0] = "";
        $msg[1] = "";

        while ($row = $stmt->fetch()) {

            $cont = "";

            if ($last < $row["cod_mensagem"]) {

                $last = $row["cod_mensagem"];

                $row["status"] = trim($row["status"]);

                if (($row["status"] != "ON" AND $row["destinatario"] != $nome) OR ($row["remetente"] ==
$nome) OR ($row["status"] == "ON" AND trim($row["destinatario"]) == "TODOS")) {
                    $msg[0].="<p><table width=100% height=25><tr><td><font face=Tahoma
size=2> ".$row["mensagem"]."</font></td></tr></table></p>";
                    $cont = 1;
                }
                elseif ($row["destinatario"] == $nome AND $row["status"] != "ON") {
                    $msg[0].="<p><table cellspacing=0 cellpadding=0 bordercolor=#CCCCCC border=1
width=100% height=25 bgcolor=#E1E3E3><tr><td><font face=Tahoma
size=2> ".$row["mensagem"]."</font></td></tr></table></p>";
                    $cont = 1;
                }
                elseif ($row["destinatario"] == $nome AND $row["status"] == "ON") {
                    $msg[0].="<p><table cellspacing=0 cellpadding=0 bordercolor=#EFB5A0 border=1
width=100% height=25 bgcolor=#ECDDDD><tr><td><font face=Tahoma
size=2> ".$row["mensagem"]."</font></td></tr></table></p>";
                    $cont = 1;
                }

                if (($row["destinatario"] == $nome) OR (trim($row["destinatario"]) == "TODOS") OR
($row["remetente"] == $nome)){
                    $msg[1] = trim($row["som"]);
                }
```

```
            }
        }
        return $msg;

    }

}

?>
```

15.4. ChatMensagemPersistenceManager.class.php

```
<?php

/**
 * @package model
 */

/**
 * Interface que define um Manager, classe responsável pelo gerenciamento
 * de persistência de alguma classe do Model. Todos os Managers deste módulo
 * devem implementar no mínimo os métodos definidos por este contrato.
 *
 * Essa idéia segue o design pattern DAO (Data Access Object).
 *
 * @static
 * @package model
 */
interface ChatMensagemPersistenceManager {

    /**
     * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica
da operação,
     * se vai ser uma inserção, é dependente de cada classe Model e deve ser especificada na
     * implementação do método pela classe concreta.
     *
     * @param mixed $mensagem Objeto a ser gravado.
     */
    public static function save($mensagem);

    /**
     * Método utilizado para exclusão de uma entidade da classe Model correspondente.
     *
     * @param mixed $codChat Código do Chat das mensagens antigas que serão excluídas.
     */
    public static function delete($codChat);

    /**
     * Método utilizado para impressão das mensagens destinadas a uma entidade.
     *
     * @param mixed $nome Passa o apelido do usuário como parâmetro para buscar mensagens
destinadas a ele.
     * @param mixed $last Passa o código da última mensagem enviada pelo usuário para buscar
mensagens posteriores a ela.
     * @param mixed $codChat Passa o código do Chat.
     * @return mixed Retorna a mensagem formatada ou <kbd>null</kbd>, caso não encontre.
     */
    public static function imprimir($nome, &$last, $codChat);

}

?>
```

15.5. ChatUsuarioManager.class.php

```
<?php
/**
 * @package model
 */

/**
 * Classe que define o gerenciador de persistência para a classe <kbd>ChatUsuario</kbd>.
 *
 * @static
 * @see ChatUsuarioPersistenceManager
 * @package model
 */
class ChatUsuarioManager implements ChatUsuarioPersistenceManager {

    /**
     * @see ChatUsuarioPersistenceManager::findByCodUsuario()
     */
    public static function findByCodUsuario($codUsuario, $codChat) {

    $pdo = Util::getPDOHandler();

    $stmt = $pdo->prepare("SELECT chat_cod_chat, usuario_cod_usuario, grupo_cod_grupo, nome,
cor, last, status FROM acesso_chat WHERE usuario_cod_usuario = :codUsuario AND chat_cod_chat
= :codChat");

    $stmt->bindParam(":codUsuario", $codUsuario);

    $stmt->bindParam(":codChat", $codChat);

    $stmt->execute();

        if ($row = $stmt->fetch()) {
            $usuario = new ChatUsuario();
            $usuario->setCodChat($row["chat_cod_chat"]);
            $usuario->setCodUsuario($row["usuario_cod_usuario"]);
            $usuario->setCodGrupo($row["grupo_cod_grupo"]);
            $usuario->setNome($row["nome"]);
            $usuario->setCor($row["cor"]);
            $usuario->setLast($row["last"]);
            $usuario->setStatus($row["status"]);
        } else {
            $usuario = new ChatUsuario();
            $usuario->setCodUsuario("");
        }

        return $usuario;

    }

    /**
     * @see ChatUsuarioPersistenceManager::findByRecursoUsuario()
     */
    public static function findByRecursoUsuario($codUsuario, $codChat) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT r.nome_recurso FROM config_chat as cc, acesso_chat as ac,
chat as c, recurso_chat as r WHERE
```

```
                        c.cod_chat = ac.chat_cod_chat AND ac.grupo_cod_grupo = cc.grupo_cod_grupo
AND c.cod_chat = cc.chat_cod_chat
                        AND r.cod_recurso = recurso_cod_recurso AND cod_chat = :codChat AND
ac.usuario_cod_usuario = :codUsuario");

        $stmt->bindParam(":codUsuario", $codUsuario);

        $stmt->bindParam(":codChat", $codChat);

        $stmt->execute();

        $l = array();

        while($row = $stmt->fetch()){

            $l[] = $row["nome_recurso"];

        }

        return $l;

    }

    /**
     * @see ChatUsuarioPersistenceManager::findByCodUsuarioCont()
     */
    public static function findByCodUsuarioCont($codUsuario, $codChat) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT usuario_cod_usuario FROM acesso_chat WHERE
usuario_cod_usuario = :codUsuario AND chat_cod_chat = :codChat");

        $stmt->bindParam(":codUsuario", $codUsuario);

        $stmt->bindParam(":codChat", $codChat);

        $stmt->execute();

        $cont = $stmt->rowCount();

        return $cont;
    }

    /**
     * @see ChatUsuarioPersistenceManager::findByNickCont()
     */
    public static function findByNickCont($nome, $codChat) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT usuario_cod_usuario FROM acesso_chat WHERE nome =
:nome AND chat_cod_chat = :codChat");

        $stmt->bindParam(":nome", $nome);

        $stmt->bindParam(":codChat", $codChat);

        $stmt->execute();
```

```
        $cont = $stmt->rowCount();

        return $cont;
    }

    /**
     * @see ChatUsuarioPersistenceManager::save()
     */
    public static function save($usuario) {

        $pdo = Util::getPDOHandler();

        if($usuario->getNome() != ""){
            $sql = "UPDATE acesso_chat SET nome = :nome, cor = :cor, last = :last, status = :status
WHERE usuario_cod_usuario = :codUsuario AND chat_cod_chat = :codChat";
            $chave= 1;
        } elseif($usuario->getStatus() == 0){
            $sql = "UPDATE acesso_chat SET nome = :nome, cor = :cor, last = :last, status = :status
WHERE usuario_cod_usuario = :codUsuario AND chat_cod_chat = :codChat";
            $chave= 2;
        } else {
            $sql = "UPDATE acesso_chat SET last = :last, status = :status WHERE cod_usuario =
:codUsuario AND cod_chat = :codChat";
            $chave= 3;
        }

        $stmt = $pdo->prepare($sql);

        if($usuario->getNome() != ""){
            $stmt->execute(array(":codUsuario" => $usuario->getCodUsuario(), ":codChat" => $usuario-
>getCodChat(), ":nome" => $usuario->getNome(), ":cor" => $usuario->getCor(), ":last" => $usuario-
>getLast(), ":status" => $usuario->getStatus()));
        } elseif($usuario->getStatus() == 0){
            $stmt->execute(array(":codUsuario" => $usuario->getCodUsuario(), ":codChat" => $usuario-
>getCodChat(), ":nome" => $usuario->getNome(), ":cor" => $usuario->getCor(), ":last" => $usuario-
>getLast(), ":status" => $usuario->getStatus()));
        } else {
            $stmt->execute(array(":codUsuario" => $usuario->getCodUsuario(), ":codChat" => $usuario-
>getCodChat(), ":last" => $usuario->getLast(), ":status" => $usuario->getStatus()));
        }
    }

    /**
     * @see ChatUsuarioPersistenceManager::imprimir()
     */
    public static function imprimir($codChat, $status) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT nome FROM acesso_chat WHERE chat_cod_chat = :codChat
AND status = :status ORDER BY nome ASC");

        $stmt->bindParam(":codChat", $codChat);

        $stmt->bindParam(":status", $status);

        $stmt->execute();

        $l = array();
        $l[] = "TODOS";
```

```
        while ($row = $stmt->fetch()) {
          $c = new ChatUsuario();
          $c->setNome($row["nome"]);
          $l[] = trim($c->getNome());
        }

        return $l;
    }

    /**
     * @see ChatUsuarioPersistenceManager::nick()
     */
    public static function nick($nome, $codChat) {

        $nick = "TODOS";

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT nome, cor FROM acesso_chat WHERE nome = :nome AND
chat_cod_chat = :codChat");

        $stmt->bindParam(":nome", $nome);

        $stmt->bindParam(":codChat", $codChat);

        $stmt->execute();

        $cont = $stmt->rowCount();

        if ($cont == 1) {

          $row = $stmt->fetch();

          switch (trim($row["cor"])){
               case "Azul":
                   $cor="#0000FF";
                   break;
               case "Vermelho":
                   $cor="#FF0000";
                   break;
               case "Preto":
                   $cor="#000000";
                   break;
               case "Verde":
                   $cor="#339933";
                   break;
               case "Roxo":
                   $cor="#9900CC";
                   break;
               case "Laranja":
                   $cor="#FF9900";
                   break;
          }
          $nick = "<font color=$cor><b>".trim($row[nome])."</b></font>";
        }
        return $nick;

    }
```

```
    /**
    * @see ChatUsuarioPersistenceManager::atualiza()
    */
    public static function atualiza($codChat) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT MAX(last) as last FROM acesso_chat WHERE chat_cod_chat
= :codChat");

        $stmt->bindParam(":codChat", $codChat);

        $stmt->execute();

        $row = $stmt->fetch();

        $last = $row["last"];

        $limite = $last - 50;

        $empty = "";

        $stmt1 = $pdo->prepare("UPDATE acesso_chat SET nome = :empty, cor = :empty, last = 0,
status = 0 WHERE chat_cod_chat = :codChat AND last < :limite AND last > 0");

        $stmt1->bindParam(":codChat", $codChat);

        $stmt1->bindParam(":last", $last);

        $stmt1->bindParam(":limite", $limite);

        $stmt1->bindParam(":empty", $empty);

        $stmt1->execute();

    }

    /**
     * @see ChatUsuarioPersistenceManager::fetchAll()
     */

    public static function fetchAll() {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT chat_cod_chat, usuario_cod_usuario, grupo_cod_grupo FROM
acesso_chat ORDER BY chat_cod_chat ASC");
        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {

            $c = new ChatUsuario();
            $c->setCodChat($row["chat_cod_chat"]);
            $c->setCodUsuario($row["usuario_cod_usuario"]);
            $c->setCodGrupo($row["grupo_cod_grupo"]);
            $l[] = $c;
        }

        return $l;
```

```
    }

    /**
     * @see ChatUsuarioPersistenceManager::fetchDistinct()
     */
    public static function fetchDistinct() {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT DISTINCT(chat_cod_chat) FROM acesso_chat ORDER BY
chat_cod_chat ASC");
        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {

            $c = new ChatUsuario();
            $c->setCodChat($row["chat_cod_chat"]);
            $l[] = $c;
        }

        return $l;

    }

    /**
     * @see ChatUsuarioPersistenceManager::findByCod()
     */
    public static function findByCod($codChat, $codUsuario, $codGrupo) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT * FROM acesso_chat WHERE (chat_cod_chat = :codChat
AND usuario_cod_usuario = :codUsuario AND grupo_cod_grupo = :codGrupo)");
        $stmt->bindParam(":codChat", $codChat);
        $stmt->bindParam(":codUsuario", $codUsuario);
        $stmt->bindParam(":codGrupo", $codGrupo);
        $stmt->execute();

        if ($row = $stmt->fetch()) {

            $c = new ChatUsuario();
            $chat = ChatManager::findByKey($row["chat_cod_chat"]);
            $c->setCodChat($chat->getNome());
            $usuario = UsuarioManager::findByKey($row["usuario_cod_usuario"]);
            $c->setCodUsuario($usuario->getNome());
            $grupo = GrupoManager::findByKey($row["grupo_cod_grupo"]);
            $c->setCodGrupo($grupo->getNome());

        }

        return $c;

    }

    /**
     * @see ChatUsuarioPersistenceManager::findByKey()
     */
    public static function findByKey($key) {
```

```
        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT * FROM acesso_chat WHERE chat_cod_chat = :codChat");
        $stmt->bindParam(":codChat", $key);
        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {
            $c = new ChatUsuario();
            $c->setCodChat($row["chat_cod_chat"]);
            $c->setCodUsuario($row["usuario_cod_usuario"]);
            $c->setCodGrupo($row["grupo_cod_grupo"]);
            $l[] = $c;
        }

        return $l;

    }

    /**
     * @see ChatUsuarioPersistenceManager::saveAcesso()
     */
    public static function saveAcesso($chatUsuario, $codChat) {

    try {

        $pdo = Util::getPDOHandler();

        $pdo->beginTransaction();

        // Extraindo código de CHAT do objeto acessoChat e construindo um array no formato
array[key]=codChat
        $arrCodChat = array();
        foreach ($chatUsuario->getCodChat() as $chat) {
            $arrCodChat[] = $chat->getCod();
        }

        // Extraindo código de GRUPO do objeto acessoChat e construindo um array no formato
array[key]=codGrupo
        $arrCodGrupo = array();
        foreach ($chatUsuario->getCodGrupo() as $grupo) {
            $arrCodGrupo[] = $grupo->getCod();
        }

        if ($codChat != 0) {

            // Extraindo código de USUÁRIO do objeto acessoChat e construindo um array no formato
array[key]=codGrupo
            $arrCodUsuario = array();
            foreach ($chatUsuario->getCodUsuario() as $usuario) {
                $arrCodUsuario[] = $usuario->getCod();
            }

            // Inserindo na tabela acesso_chat
            foreach ($arrCodChat as $cod){
                foreach ($arrCodUsuario as $codUsuario) {
                    foreach ($arrCodGrupo as $codGrupo) {
                        $stmt = $pdo->prepare("INSERT INTO acesso_chat (chat_cod_chat,
usuario_cod_usuario, grupo_cod_grupo) VALUES (:codChat, :codUsuario, :codGrupo)");
```

```
                $stmt->bindParam(":codChat", $cod);
                $stmt->bindParam(":codUsuario", $codUsuario);
                $stmt->bindParam(":codGrupo", $codGrupo);
                $stmt->execute();
            }
        }
    }

}
else {

    // Extraindo código de USUÀRIO / GRUPO do objeto acessoChat e construindo um array no formato array[key]=codUsuario - codChat
    $arrCodUsuarioGrupo = array();
    foreach ($chatUsuario->getCodUsuario() as $usuario) {
        $arrCodUsuarioGrupo[] = $usuario->getCod();
    }

    $cont = count($arrCodChat);

    for ($i = 0; $i < $cont; $i++) {

        $codChat = $arrCodChat[$i];
        $codGrupo = $arrCodGrupo[$i];

        $c = count($arrCodUsuarioGrupo[$i]);

        for ($j = 0; $j < $c; $j++) {

            $acesso = explode(" - ", $arrCodUsuarioGrupo[$i][$j]);

            $codUsuario = $acesso[0];
            $codGrupoAntigo = $acesso[1];

            // Atualização na tabela acesso_chat

            $stmt = $pdo->prepare("UPDATE acesso_chat SET grupo_cod_grupo = :codGrupo WHERE chat_cod_chat = :codChat AND usuario_cod_usuario = :codUsuario AND grupo_cod_grupo = :codGrupoAntigo");
            $stmt->bindParam(":codGrupo", $codGrupo);
            $stmt->bindParam(":codChat", $codChat);
            $stmt->bindParam(":codUsuario", $codUsuario);
            $stmt->bindParam(":codGrupoAntigo", $codGrupoAntigo);
            $stmt->execute();

        }

    }

}

$pdo->commit();

    } catch (PDOException $e) {
        $info = $e->getCode();
        return $info;
    }

}
```

```
    /**
     * @see ChatUsuarioPersistenceManager::delete()
     */
    public static function delete($chatUsuario) {

        $pdo = Util::getPDOHandler();

        $pdo->beginTransaction();

        // Extraindo código de CHAT do objeto acessoChat e construindo um array no formato
array[key]=codChat
        $arrCodChat = array();
        foreach ($chatUsuario->getCodChat() as $chat) {
            $arrCodChat[] = $chat->getCod();
        }

        // Extraindo código de USUÀRIO / GRUPO do objeto acessoChat e construindo um array no
formato array[key]=codUsuario - codChat
        $arrCodUsuarioGrupo = array();
        foreach ($chatUsuario->getCodUsuario() as $usuario) {
            $arrCodUsuarioGrupo[] = $usuario->getCod();
        }

        $cont = count($arrCodChat);

        for ($i = 0; $i < $cont; $i++) {

            $codChat = $arrCodChat[$i];

            $c = count($arrCodUsuarioGrupo[$i]);

            for ($j = 0; $j < $c; $j++) {

                $acesso = explode(" - ", $arrCodUsuarioGrupo[$i][$j]);

                $codUsuario = $acesso[0];
                $codGrupo = $acesso[1];

                // Exclusão na tabela acesso_chat

                $stmt = $pdo->prepare("DELETE FROM acesso_chat WHERE chat_cod_chat = :codChat
AND usuario_cod_usuario = :codUsuario AND grupo_cod_grupo = :codGrupo");
                $stmt->bindParam(":codGrupo", $codGrupo);
                $stmt->bindParam(":codChat", $codChat);
                $stmt->bindParam(":codUsuario", $codUsuario);
                $stmt->execute();

            }

        }

        $pdo->commit();

    }

}


?>
```

15.6. ChatUsuarioPersistenceManager.class.php

```
<?php

/**
 * @package model
 */

/**
 * Interface que define um Manager, classe responsável pelo gerenciamento
 * de persistência de alguma classe do Model. Todos os Managers deste módulo
 * devem implementar no mínimo os métodos definidos por este contrato.
 *
 * Essa idéia segue o design pattern DAO (Data Access Object).
 *
 * @static
 * @package model
 */
interface ChatUsuarioPersistenceManager {

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através do
Código do Usuário e do Código do Chat.
     *
     * @param mixed $codUsuario Código do Usuário da entidade desejada.
     * @param mixed $codChat Código do Chat da entidade desejada.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
    public static function findByCodUsuario($codUsuario, $codChat);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através do
Código do Usuário e do Código do Chat.
     *
     * @param mixed $codUsuario Código do Usuário da entidade desejada.
     * @param mixed $codChat Código do Chat da entidade desejada.
     * @return mixed Retorna um array de objetos encontrado ou <kbd>null</kbd>, caso não encontre.
     */
    public static function findByRecursoUsuario($codUsuario, $codChat);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através do
Código do Usuário e do Código do Chat.
     *
     * @param mixed $codUsuario Código do Usuário da entidade desejada.
     * @param mixed $codChat Código do Chat da entidade desejada.
     * @return mixed Retorna a quantidade de objetos encontrados ou <kbd>0</kbd>, caso não
encontre.
     */
    public static function findByCodUsuarioCont($codUsuario, $codChat);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através do
nome e do Código do Chat.
     *
     * @param mixed $nome Nome da entidade desejada.
     * @param mixed $codChat Código do Chat da entidade desejada.
     * @return mixed Retorna o a quantidade de objetos encontrados ou <kbd>0</kbd>, caso não
encontre.
     */
```

```
    public static function findByNickCont($nome, $codChat);

    /**
     * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica
da operação,
     * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser
especificada na
     * implementação do método pela classe concreta. O exemplo mais comum para chave primária
inteira é assumir
     * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser
inserida)
     * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas
atualizada).
     *
     * @param mixed $model Objeto a ser gravado.
     */
    public static function save($usuario);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através do
Código do Chat e do Status.
     *
     * @param mixed $codChat Código do Chat da entidade desejada.
     * @param mixed $status Status da entidade desejada.
     * @return mixed Retorna um array de objetos encontrados ou <kbd>0</kbd>, caso não encontre.
     */
    public static function imprimir($codChat, $status);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através do
Nome e do Código do Chat.
     *
     * @param mixed $nome Nome da entidade desejada.
     * @param mixed $codChat Código do Chat da entidade desejada.
     * @return mixed Retorna uma variavel encontrada ou <kbd>0</kbd>, caso não encontre.
     */
    public static function nick($nome, $codChat);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através do
Código do Chat.
     *
     * @param mixed $codChat Código do Chat da entidade desejada.
     */
    public static function atualiza($codChat);

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchAll();

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados distintos. No caso
de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
```

```
     */
    public static function fetchDistinct();

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através do Código do Chat, Código do Usuário e Código do Grupo.
     *
     * @param mixed $codChat Código do Chat da entidade desejada.
     * @param mixed $codUsuario Código do Usuário da entidade desejada.
     * @param mixed $codGrupo Código do Grupo da entidade desejada.
     * @return mixed Retorna um objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
    public static function findByCod($codChat, $codUsuario, $codGrupo);


    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através de sua chave primária.
     *
     * @param mixed $key Chave da entidade desejada, segundo a definição de chave primária da classe Model.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
    public static function findByKey($key);

    /**
     * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica da operação,
     * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser especificada na
     * implementação do método pela classe concreta. O exemplo mais comum para chave primária inteira é assumir
     * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser inserida)
     * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas atualizada).
     *
     * @param mixed $model Objeto a ser gravado.
     * @param mixed $codChat Código do Chat a ser gravado.
     */
    public static function saveAcesso($model, $codChat);

    /**
     * Método utilizado para exclusão de uma entidade da classe Model correspondente através de um Objeto.
     *
     * @param mixed $model Objeto da entidade a ser excluída.
     */
    public static function delete($model);


}

?>
```

15.7. ConfigChatManager.class.php

```
<?php

/**
 * @package model
 */

/**
 * Classe que define o gerenciador de persistência para a classe <kbd>ConfigChat</kbd>.
 *
 * @static
 * @see PersistenceManager
 * @package model
 */
class ConfigChatManager implements ConfigChatPersistenceManager {

    /**
     * @see PersistenceManager::fetchAll()
     */
    public static function fetchAll($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT * FROM config_chat WHERE chat_cod_chat = :codChat
ORDER BY chat_cod_chat ASC");
        $stmt->bindParam(":codChat",$key);
        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {

            $c = new ConfigChat();
            $c->setCodChat($row["chat_cod_chat"]);
            $c->setCodRecurso($row["recurso_cod_recurso"]);
            $c->setCodGrupo($row["grupo_cod_grupo"]);
            $l[] = $c;
        }

        return $l;

    }

    /**
     * @see PersistenceManager::findByKey()
     */
    public static function findByKey($codChat, $codRecurso, $codGrupo) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT * FROM config_chat WHERE (chat_cod_chat = :codChat AND
recurso_cod_recurso = :codRecurso AND grupo_cod_grupo = :codGrupo)");
        $stmt->bindParam(":codChat", $codChat);
        $stmt->bindParam(":codRecurso", $codRecurso);
        $stmt->bindParam(":codGrupo", $codGrupo);
        $stmt->execute();

        if ($row = $stmt->fetch()) {
            $c = new ConfigChat();
            $chat = ChatManager::findByKey($row["chat_cod_chat"]);
```

```
            $c->setCodChat($chat->getNome());
            $recurso = RecursoManager::findByKey($row["recurso_cod_recurso"]);
            $c->setCodRecurso($recurso->getNome());
            $grupo = GrupoManager::findByKey($row["grupo_cod_grupo"]);
            $c->setCodGrupo($grupo->getNome());
        }

        return $c;

    }

    /**
     * Método responsável em incluir uma nova plataforma no sistema
     * e ao mesmo tempo inclui os módulos selecionados.
     *
     * @see PersistenceManager::save()
     */
    public static function save($configChat) {

    //p($plataforma);
        try {

            $pdo = Util::getPDOHandler();

            $pdo->beginTransaction();

            // Extraindo código de CHAT do objeto configChat e construindo um array no formato
array[key]=codChat
            $arrCodChat = array();
            foreach ($configChat->getCodChat() as $chat) {
                $arrCodChat[] = $chat->getCod();
            }

            // Extraindo código de RECURSO do objeto configChat e construindo um array no formato
array[key]=codRecurso
            $arrCodRecurso = array();
            foreach ($configChat->getCodRecurso() as $recurso) {
                $arrCodRecurso[] = $recurso->getCod();
            }

            // Extraindo código de GRUPO do objeto configChat e construindo um array no formato
array[key]=codGrupo
            $arrCodGrupo = array();
            foreach ($configChat->getCodGrupo() as $grupo) {
                $arrCodGrupo[] = $grupo->getCod();
            }

            // Inserindo na tabela config_chat
            foreach ($arrCodChat as $codChat){
                foreach ($arrCodRecurso as $codRecurso) {
                    foreach ($arrCodGrupo as $codGrupo) {
                        $stmt = $pdo->prepare("INSERT INTO config_chat (chat_cod_chat,
recurso_cod_recurso, grupo_cod_grupo) VALUES (:codChat, :codRecurso, :codGrupo)");
                        $stmt->bindParam(":codChat", $codChat);
                        $stmt->bindParam(":codRecurso", $codRecurso);
                        $stmt->bindParam(":codGrupo", $codGrupo);
                        $stmt->execute();
                    }
                }
            }
```

```
            $pdo->commit();

            } catch (PDOException $e) {
                $info = $e->getCode();
                return $info;
            }
    }

    /**
     * @see PersistenceManager::delete()
     */
    public static function delete($codChat, $codRecurso, $codGrupo) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("DELETE FROM config_chat WHERE chat_cod_chat = :codChat AND
recurso_cod_recurso = :codRecurso AND grupo_cod_grupo = :codGrupo");
        $stmt->bindParam(":codChat", $codChat);
        $stmt->bindParam(":codRecurso", $codRecurso);
        $stmt->bindParam(":codGrupo", $codGrupo);
        $stmt->execute();
    }

}
?>
```

15.8. ConfigChatPersistenceManager.class.php

```
<?php

/**
 * @package model
 */

/**
 * Interface que define um Manager, classe responsável pelo gerenciamento
 * de persistência de alguma classe do Model. Todos os Managers deste módulo
 * devem implementar no mínimo os métodos definidos por este contrato.
 *
 * Essa idéia segue o design pattern DAO (Data Access Object).
 *
 * @static
 * @package model
 */
interface ConfigChatPersistenceManager {

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchAll($key);

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através de sua
chave primária.
     *
     * @param mixed $key Chave da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
       public static function findByKey($codChat, $codRecurso, $codGrupo);

    /**
     * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica
da operação,
     * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser
especificada na
     * implementação do método pela classe concreta. O exemplo mais comum para chave primária
inteira é assumir
     * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser
inserida)
     * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas
atualizada).
     *
     * @param mixed $model Objeto a ser gravado.
     */
       public static function save($model);

    /**
     * Método utilizado para exclusão de uma entidade da classe Model correspondente através da
combinação de três códigos gerando sua chave primária.
     *
     * @param mixed $codChat Chave da entidade a ser excluída.
     * @param mixed $codRecurso Chave da entidade a ser excluída.
```

```
    * @param mixed $codGrupo Chave da entidade a ser excluída, segundo a definição de chave primária da classe Model.
    */
      public static function delete($codChat, $codRecurso, $codGrupo);

}

?>
```

15.9. RecursoManager.class.php

```
<?php
/**
 * @package model
 */

/**
 * Classe que define o gerenciador de persistência para a classe <kbd>Recurso</kbd>.
 *
 * @static
 * @see RecursoPersistenceManager
 * @package model
 */
class RecursoManager implements RecursoPersistenceManager {

    /**
     * @see RecursoPersistenceManager::fetchAll()
     */
    public static function fetchAll() {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT cod_recurso, nome_recurso FROM recurso_chat ORDER BY
nome_recurso ASC");
        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {
            $c = new Recurso();
            $c->setCod($row["cod_recurso"]);
            $c->setNome($row["nome_recurso"]);
            $l[] = $c;
        }

        return $l;

    }

    /**
     * @see RecursoPersistenceManager::findByKey()
     */
    public static function findByKey($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT cod_recurso, nome_recurso FROM recurso_chat WHERE
cod_recurso = :cod");

        $stmt->bindParam(":cod", $key);

        $stmt->execute();

        if ($row = $stmt->fetch()) {
            $recurso = new Recurso();
            $recurso->setCod($key);
            $recurso->setNome($row["nome_recurso"]);
        } else {
            $recurso = new Recurso();
            $recurso->setCod(-1);
```

```
        }

        return $recurso;

    }

    /**
     * @see RecursoPersistenceManager::save()
     */
    public static function save($recurso) {

        $pdo = Util::getPDOHandler();

        if ($recurso->getCod() == 0) {
            $sql = "INSERT INTO recurso_chat (nome_recurso) VALUES (:nome)";
        }
        else {
            $sql = "UPDATE recurso_chat SET nome_recurso = :nome WHERE cod_recurso = :cod";
        }

        $stmt = $pdo->prepare($sql);

        if ($recurso->getCod() == 0) {
            $stmt->execute(array(":nome" => $recurso->getNome()));
        }
        else {
            $stmt->execute(array(":cod" => $recurso->getCod(), ":nome" => $recurso->getNome()));
        }

    }

    /**
     * @see RecursoPersistenceManager::delete()
     */
    public static function delete($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("DELETE FROM recurso_chat WHERE cod_recurso = :cod");

        $stmt->bindParam(":cod", $key);
        $stmt->execute();

    }

}

?>
```

15.10. RecursoPersistenceManager.class.php

```
<?php

/**
 * @package model
 */

/**
 * Interface que define um Manager, classe responsável pelo gerenciamento
 * de persistência de alguma classe do Model. Todos os Managers deste módulo
 * devem implementar no mínimo os métodos definidos por este contrato.
 *
 * Essa idéia segue o design pattern DAO (Data Access Object).
 *
 * @static
 * @package model
 */
interface RecursoPersistenceManager {

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchAll();

    /**
     * Método utilizado para obtenção de uma entidade da classe Model correspondente através de sua
chave primária.
     *
     * @param mixed $key Chave da entidade desejada, segundo a definição de chave primária da
classe Model.
     * @return mixed Retorna o objeto encontrado ou <kbd>null</kbd>, caso não encontre.
     */
    public static function findByKey($key);

    /**
     * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica
da operação,
     * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser
especificada na
     * implementação do método pela classe concreta. O exemplo mais comum para chave primária
inteira é assumir
     * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser
inserida)
     * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas
atualizada).
     *
     * @param mixed $model Objeto a ser gravado.
     */
    public static function save($model);

    /**
     * Método utilizado para exclusão de uma entidade da classe Model correspondente através de sua
chave primária.
     *
     * @param mixed $key Chave da entidade a ser excluída, segundo a definição de chave primária
da classe Model.
```

```
     */
    public static function delete($key);

}

?>
```

15.11. RequisicaoChatManager.class.php

```
<?php
/**
 * @package model
 */

/**
 * Classe que define o gerenciador de persistência para a classe <kbd>Requisicao</kbd>.
 *
 * @static
 * @see RequisicaoChatPersistenceManager
 * @package model
 */
class RequisicaoChatManager implements RequisicaoChatPersistenceManager {

    /**
     * @see RequisicaoChatPersistenceManager::fetchAll()
     */
    public static function fetchAll($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT * FROM requisicao_chat WHERE cod_chat = :codChat
ORDER BY cod_usuario ASC");
        $stmt->bindParam(":codChat", $key);
        $stmt->execute();

        $l = array();
        while ($row = $stmt->fetch()) {
            $c = new RequisicaoChat();
            $c->setCodChat($row["cod_chat"]);
            $usuario = UsuarioManager::findByKey($row["cod_usuario"]);
            $c->setCodUsuario($row["cod_usuario"] . "-" . $usuario->getNome());
            $l[] = $c;
        }
        return $l;

    }

    /**
     * @see RequisicaoChatPersistenceManager::fetchAll()
     */
    public static function fetchCont($key) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("SELECT * FROM requisicao_chat WHERE cod_chat = :codChat
ORDER BY cod_usuario ASC");
        $stmt->bindParam(":codChat", $key);
        $stmt->execute();

        $cont = $stmt->rowCount();

        return $cont;

    }

     /**
     * @see RequisicaoChatPersistenceManager::save()
```

```
     */
    public static function save($requisicaoChat) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("INSERT INTO requisicao_chat (cod_chat, cod_usuario) VALUES
(:codChat, :codUsuario)");

        $stmt->execute(array(":codChat" => $requisicaoChat->getCodChat(), ":codUsuario" =>
$requisicaoChat->getCodUsuario()));

        $chat = ChatManager::findByKey($requisicaoChat->getCodChat());

        $usuario = UsuarioManager::findByKey($requisicaoChat->getCodUsuario());
        $infoUsuario = UsuarioManager::findByKey($chat->getCodUsuarioChat());

        $assunto = _MAIL_ASSUNTO_ . " Requisição de Participação no Chat";
        $corpoMsg = "Requisição de Participação no Chat - " . $chat->getNome()."\n"
        . "\n"
        . "Nome do Usuário: ".$usuario->getNome() . "\n"
        . "Login do Usuário: " .$usuario->getLogin()."\n"
        . "Instituição do Usuário: ".$usuario->getInstituicao()."\n"
        . "Email do Usuário: ".$usuario->getEmail();

        Util::mailSender($infoUsuario->getEmail(), $assunto, $corpoMsg);

    }

    /**
     * @see RequisicaoChatPersistenceManager::delete()
     */
    public static function delete($requisicaoChat) {

        $pdo = Util::getPDOHandler();

        $stmt = $pdo->prepare("DELETE FROM requisicao_chat WHERE cod_chat = :codChat AND
cod_usuario = :codUsuario");

        $stmt->execute(array(":codChat" => $requisicaoChat->getCodChat(), ":codUsuario" =>
$requisicaoChat->getCodUsuario()));

        $chatUsuario = ChatUsuarioManager::findByCodUsuario($requisicaoChat->getCodUsuario(),
$requisicaoChat->getCodChat());

        $chat = ChatManager::findByKey($requisicaoChat->getCodChat());
        $usuario = UsuarioManager::findByKey($requisicaoChat->getCodUsuario());

        if ($chatUsuario->getCodChat() == "") {

            $infoUsuario = UsuarioManager::findByKey($chat->getCodUsuarioChat());

            $assunto = _MAIL_ASSUNTO_ . " Requisição Recusada";
            $corpoMsg = "Sua requisição foi recusada pelo Administrador do Chat. \n"
            . "\n"
            . "Nome do Chat: ".$chat->getNome() . "\n"
            . "Para maiores informações entre em contato com o Administrador do Chat: ". $infoUsuario-
>getEmail();

        }
        else {
```

```
            $assunto = _MAIL_ASSUNTO_ . " Requisição Aceita";
            $corpoMsg = "Sua requisição foi aceita pelo Administrador do Chat. \n"
            . "\n"
            . "Nome do Chat: ".$chat->getNome() . "\n"
            . "\n"
            . "WorldConf";
        }

        Util::mailSender($usuario->getEmail(), $assunto, $corpoMsg);

    }

}

?>
```

15.12. RequisicaoChatPersistenceManager.class.php

```
<?php

/**
 * Interface que define um Manager, classe responsável pelo gerenciamento
 * de persistência de alguma classe do Model. Todos os Managers deste módulo
 * devem implementar no mínimo os métodos definidos por este contrato.
 *
 * Essa idéia segue o design pattern DAO (Data Access Object).
 *
 * @static
 * @package model
 */
interface RequisicaoChatPersistenceManager {

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchAll($key);

    /**
     * Método utilizado para obtenção de todas as entidades da classe Model correspondente.
     *
     * @return array Retorna um <kbd>array</kbd> contendo os objetos encontrados. No caso de não
     * haver nenhum objeto, <kbd>array</kbd> estará vazio.
     */
    public static function fetchCont($key);

     /**
     * Método utilizado para gravação de uma entidade da classe Model correspondente. A semântica
da operação,
     * se vai ser uma inserção ou atualização, é dependente de cada classe Model e deve ser
especificada na
     * implementação do método pela classe concreta. O exemplo mais comum para chave primária
inteira é assumir
     * que o ID 0 (zero) sinaliza uma entidade ainda não persistida no banco de dados (e que deve ser
inserida)
     * e que ID diferente disso mostra que a entidade já existe na base (devendo portanto ser apenas
atualizada).
     *
     * @param mixed $model Objeto a ser gravado.
     */
    public static function save($model);

    /**
     * Método utilizado para exclusão de uma entidade da classe Model correspondente através de sua
chave primária.
     *
     * @param mixed $key Chave da entidade a ser excluída, segundo a definição de chave primária
da classe Model.
     */
    public static function delete($model);

}

?>
```

16. MÓDULO CHAT – TEMPLATES

16.1. chat/arquivo.tpl

```
<html>

<head>
<link rel="stylesheet" type="text/css" href="style.css"/>
<script language="javascript">
{literal}
function Fechar(){
        window.self.close();
}
{/literal}
</script>
</head>

<body bgcolor="#006699">
<br><br><div align="center"><font face="Tahoma" color="#FFFFFF" size="3"><b>Arquivo Enviado
com Sucesso!</b></font></div>
<script language="JavaScript">
setTimeout("javascript:Fechar();", "3000");
</script>
</body>
</html>
```

16.2. chat/chat.tpl

```
<html>
<title>Sala de Conferência - {$nomeChat}</title>

<head>
<script language="javascript">
{literal}
function SairChat(){

window.open("{/literal}{php}print(Util::getURL("chat.sair"));{/php}{literal}&cod_chat={/literal}{$codChat}{
literal}&cod_usuario={/literal}{$codUsuario}{literal}", "Sair", "toolbar=no, location=no, directories=no,
status=no, menubar=no, scrollbars=no, resizable=yes, width=200, height=20, top=0, left=0");
}
{/literal}
</script>
</head>

<frameset rows="110,*,100,0" cols="*" framespacing="0" frameborder="NO" border="0"
onunLoad="javascript:
window.open('{php}print(Util::getURL('chat.sair'));{/php}&cod_chat={$codChat}&cod_usuario={$codUs
uario}', 'Sair', 'toolbar=no, location=no, directories=no, status=no, menubar=no, scrollbars=no,
resizable=yes, width=200, height=20, top=0, left=0');">
   <frame name="topo" scrolling="no" noresize
src="{php}print(Util::getURL("chat.topo"));{/php}&cod_chat={$codChat}">
   <frameset rows="*" cols="*,128" framespacing="0" frameborder="NO" border="0">
      <frameset rows="*" cols="50%,50%" framespacing="0" frameborder="NO" border="0">
           <frame name="principal" src="{php}print(Util::getURL("chat.princ"));{/php}">
           <frame name="imagem" scrolling="no" src="{php}print(Util::getURL("chat.imagem"));{/php}">
      </frameset>
   <frame src="{php}print(Util::getURL("chat.usuarioslist"));{/php}&cod_chat={$codChat}"
name="usuarios" noresize>
   </frameset>
   <frame name="inferior" scrolling="no" noresize
src="{php}print(Util::getURL("chat.form"));{/php}&cod_chat={$codChat}&cod_usuario={$codUsuario}">
   <frame name="conteudo" scrolling="no" noresize target="principal"
src="{php}print(Util::getURL("chat.ler"));{/php}&cod_chat={$codChat}&cod_usuario={$codUsuario}">
</frameset>
<noframes>
   <body>
      <p>Esta página está usando frames mas o seu navegador não suporta.</p>
   </body>
</noframes>
<html>
```

16.3. chat/form.tpl

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="style.css"/>
<script language="javascript">
{literal}
function SairChat(){
	this.top.close();
}

function TravarEnviar() {
	document.menu.enviar.disabled = true;
}
{/literal}
{if $requisicao != 0}

{literal}window.open("{/literal}{php}print(Util::getURL('chat.requisicao'));{/php}&req_ant={$req_ant}{liter
al}","Requisição","toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=no,resizabl
e=no,width=120,height=25,top=200,left=200");{/literal}
{/if}
</script>
</head>
<body background="{$smarty.const._IMAGEM_DIR_}form_fundo.gif" leftmargin="0" topmargin="0"
marginwidth="0" marginheight="0" onload="{literal}document.menu.message.focus(){/literal}">
<br>
<table border="0" cellspacing="0">
<form action="{php}print(Util::getURL("chat.trata"));{/php}" method="post" name="menu"
onSubmit="javascript:TravarEnviar();">
<input type="hidden" name="cod_chat" value="{$codChat}">
<input type="hidden" name="cod_usuario" value="{$codUsuario}">
<input type="hidden" name="nomeChat" value="{$nomeChat}">
<input type="hidden" name="nome" value="{$nome}">
<input type="hidden" name="req_ant" value="{$req_ant}">
   <tr>
		<td rowspan="2" width="25%" align="center" class="form_titulo"><b>CONFERÊNCIA
{$nomeChat}<br>USUÁRIO: </b>{$apelido}</td>

{if $recursoSala.Escrever == 'Escrever'}

	<td class="form_chat">
	  <select name="imagem" size="1">
	     <option>Imagem</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon5.gif">Assustado</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon4.gif">Careta</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon14.gif">Dúvida</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon8.gif">Eca !</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon13.gif">Exclamação</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon6.gif">Gargalhada</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon2.gif">Indeciso</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon10.gif">Na praia</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon1.gif">OK!</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon7.gif">Raiva</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon15.gif">Sorriso</option>
	     <option value="{$smarty.const._IMAGEM_DIR_}icon3.gif">Zangado</option>
	  </select>
	</td>

	<td height="40" class="form_chat">
	   <select size="1" name="modofala">
```

```
                <option value="fala para" selected>fala para</option>
                <option value="pergunta para">pergunta para</option>
                <option value="responde para">responde para</option>
                <option value="concorda com">concorda com</option>
                <option value="discorda de">discorda de</option>
                <option value="desculpa-se com">desculpa-se com</option>
                <option value="murmura para">murmura para</option>
                <option value="sorri para">sorri para</option>
                <option value="suspira por">suspira por</option>
                <option value="flerta com">flerta com</option>
                <option value="ri de">ri de</option>
                <option value="dá um fora em">dá um fora em</option>
                <option value="briga com">briga com</option>
                <option value="grita com">grita com</option>
                                    <option value=xinga>xinga</option>
                                </select>
        </td>

        <td class="form_chat">
          <select name="sound" size="1">
            <option value="" selected>enviar som:</option>
            <option value="brinde">brinde</option>
            <option value="acelera">acelera</option>
            <option value="assobio">assobio</option>
            <option value="despertador">despertador</option>
            <option value="latido">latido</option>
            <option value="miado">miado</option>
            <option value="mugido">mugido</option>
            <option value="grito">grito</option>
            <option value="vaia">vaia</option>
            <option value="gargalhada">gargalhada</option>
            <option value="ronco">ronco</option>
            <option value="beijo">beijo</option>
            <option value="aplausos">aplauso</option>
            <option value="arroto">arroto</option>
          </select>
          <input type="hidden" name="imagem_c" value="<img
src={$smarty.const._IMAGEM_DIR_}som.gif>">
          <input type="hidden" name="sound_c" value="{$smarty.const._SOUND_DIR_}">
        </td>

        <td class="form_chat">
            <b><a name="a_dest" id="a_dest">{$destinatario}</a></b>
            <input type="hidden" name="destinatario" id="destinatario" value="{$destinatario}">
        </td>

        <td align="center" class="form_titulo">
            {html_checkboxes name="status" values="ON" checked=$checked
output="Reservadamente"}
        </td>

        {if $recursoSala.Arquivo == 'Arquivo'}
            <td align="left" colspan="2" class="form_chat" width="10%">
                <input type="button" name="arquivo" value="ARQUIVO"
onClick="window.open('{php}print(Util::getURL("chat.arquivo"));{/php}&cod_chat={$codChat}&cod_usu
ario={$codUsuario}&cod=0','Upload','toolbar=no,location=no,directories=no,status=no,menubar=no,sc
rollbars=no,resizable=yes,width=350,height=140, top=200, left=200');" class="button">
            </td>
        {else}
            <td align="left" colspan="2" class="form_chat" width="10%">
```

```
            </td>
          {/if}
          {if $recursoSala.Audio == 'Audio'}
            <td align="left" rowspan="2" class="form_chat" width="18%">
               <img src="{$smarty.const._IMAGEM_DIR_}audio.png" onClick="javascript:
location.href='../modulos/chat/jnlp/{$codChat}/audio.jnlp'">
            </td>
          {else}
            <td align="left" rowspan="2" class="form_chat" width="18%">
            </td>
          {/if}

           </tr>
           <tr>
          <td colspan="5" class="form_chat" align="left" width="42%">
             <input type="text" size="90%" maxlength="500" name="message">
          </td>

          <td class="form_chat">
          </td>
          <td class="form_chat">
             <input type="submit" value="ENVIAR" class="button" name="enviar">
          </td>
    </tr>
{else}
          <td class="form_chat" width="535">
             <b>VOCÊ NÃO TEM ACESSO A ESCRITA NESSE CHAT.</b>
          </td>

           {if $recursoSala.Arquivo == 'Arquivo'}
            <td align="right" colspan="2" class="form_chat">
               <input type="button" name="arquivo" value="ARQUIVO"
onClick="window.open('{php}print(Util::getURL("chat.arquivo"));{/php}&cod_chat={$codChat}&cod_usu
ario={$codUsuario}&cod=0','Upload','toolbar=no,location=no,directories=no,status=no,menubar=no,sc
rollbars=no,resizable=no,width=320,height=150, top=200, left=200');" class="button">
            </td>
           {else}
            <td align="right" colspan="2" class="form_chat">
            </td>
           {/if}
          {if $recursoSala.Audio == 'Audio'}
            <td align="left" rowspan="2" class="form_chat" width="18%">
               <img src="{$smarty.const._IMAGEM_DIR_}audio.png" onClick="javascript:
location.href='../modulos/chat/jnlp/{$codChat}/audio.jnlp'">
            </td>
          {else}
            <td align="left" colspan="3" class="form_chat" width="28%">
            </td>
          {/if}
    </tr>
    <tr>
          <td class="form_chat" width="335">
          </td>
            </tr>
{/if}

</form>
</table>
</body>
</html>
```

16.4. chat/imagem.tpl

```
<meta HTTP-EQUIV="Refresh" CONTENT="1;url=../modulos/chat/templates/chat/arq.htm">
```

16.5. chat/index.tpl

```
<script>
{literal}
function TravarEntrar() {
        document.usuario.entrar.disabled = true;
}
{/literal}
</script>

<tr>
          <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_37.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
          <td colspan="8">
   <br>
   <table class="form_peq" align="center">
   <form name="usuario" method="post" action="{php}print(Util::getURL("chat.entra"));{/php}"
onSubmit="javascript:TravarEntrar();">
   <input type="hidden" name="cod_chat" value="{$codChat}">
   <input type="hidden" name="cod_usuario" value="{$codUsuario}">
   <tr>
      <td class="form_label">Apelido:</td>
      <td class="form_field"><input type="text" name="apelido" size="60" maxlength="12"></td>
   </tr>
   <tr>
      <td class="form_label">Cor do Apelido:</td>
      <td class="form_field"><select name="cor">
                                          <option value="Preto" selected>Preto</option>
                                          <option value="Laranja">Laranja</option>
                                          <option value="Azul">Azul</option>
                                          <option value="Roxo">Roxo</option>
                                          <option value="Verde">Verde</option>
                                          <option value="Vermelho">Vermelho</option>
                                          </select>
      </td>
   </tr>
   <tr>
      <td align="center" colspan="2" class="form_field"><input type="submit" class="button"
value="Entrar" name="entrar"></td>
   </tr>
   {if $errors|@count > 0}

   <tr>
      <td colspan="2" class="error">
         <div class="error">
         {foreach from=$errors item=error}
            {$error}<br>
         {/foreach}
         </div>
      </td>
   </tr>

   {/if}

   </form>
   </table>
</td>
</tr>
```

16.6. chat/kick.tpl

```
<script>
{literal}window.top.close();{/literal}
</script>
```

16.7. chat/ler.tpl

```
<html>
<head>
<script language="JavaScript">

{literal}
var mensagem = "{/literal}{$msg}{literal}";
var ultima = "";
if (mensagem != ultima) {
 window.parent.principal.document.writeln(mensagem);
 window.self.ultima = mensagem;
 if (navigator.userAgent.indexOf("MSIE") != "-1"){
    window.top.usuarios.history.go(0);
 } else {
    window.top.usuarios.location.reload();
 }
}
{/literal}

{if $tipoArquivo == "img"}
    {literal}parent.imagem.document.close();{/literal}
    {literal}parent.imagem.document.writeln("{/literal}{$tagImg}{literal}");{/literal}
{/if}

{literal}
function PlaySound() {
var ativar = window.parent.topo.document.form.som.checked;
 if (ativar == true) {
  var browser = (navigator.userAgent.indexOf("MSIE") != "-1") ? '<bgsound
src="{/literal}{$smarty.const._SOUND_DIR_}{$som}{literal}.wav" loop="1" hidden="true"
autostart="true">' : '<embed src="{/literal}{$smarty.const._SOUND_DIR_}{$som}{literal}.wav" loop="1"
hidden="true" autostart="true">';
 } else {
  var browser = "";
 }
 window.self.document.write(browser);
}

function RolarMensagens() {
var ativar = window.parent.topo.document.form.rolagem.checked;
 if (ativar == true) {
  window.parent.principal.scrollTo("0", "100000");
 }
}

{/literal}
{if $som != ""}
{literal}var tempo = "5000"; {/literal}
{else}
{literal}var tempo = "3000"; {/literal}
{/if}

{literal}
setTimeout("javascript:delayReload();", tempo);
function delayReload() {
 if (navigator.userAgent.indexOf("MSIE") != "-1"){
  window.self.history.go(0);
 } else {
  window.self.location.reload();
```

```
 }
}
{/literal}

</script>
</head>
<body>
<script>
{literal}RolarMensagens();{/literal}
{if $som != ""}
    {literal}PlaySound();{/literal}
{/if}
</script>
</body>
</html>
```

16.8. chat/list.tpl

```
<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_24.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
   <br>
   {assign var="chave" value=1}

   {foreach from=$edit item=e}

      <table class="list_peq" border='0' align='center' width='90%' cellpadding="5" cellspacing="0"
style="border: solid 1px #CCCCCC;">
        <tr>
          <td class="list_header" align="left" width="40%">{$e->getNome()}</td>
          <td class="list_header" align="right" width="30%">{$e->getNumPessoas()} Pessoa(s)
Participando</td>
          <td class="list_header" align="right" width="20%"><img
src="{$smarty.const._IMAGEM_DIR_}entrarChat.jpg" onClick="javascript:
window.open('{php}print(Util::getURL('chat.index'));{/php}&cod_chat={$e->getCod()}', '{$e->getCod()}',
'toolbar=no, location=no, directories=no, status=no, menubar=no, scrollbars=no, resizable=yes,
width=800, height=700, top=0, left=0');"></td>
          <td class="list_header" align="right" width="10%"><img
src="{$smarty.const._IMAGEM_DIR_}editarChat.jpg" onClick="javascript:
location.href='{php}print(Util::getURL('chat.load'));{/php}&cod={$e->getCod()}'"></td>
        </tr>
        <tr colspan=4>
          <td class="list_body2" colspan=4>{$e->getDescricao()}</td>
        </tr>
        <tr>
          <td class="list_body2" colspan=4>Criado em {$e->getDtCriacao()|truncate:19:"":true}</td>
        </tr>
        <tr>
          <td class="list_body2" colspan=4>Data Início: {$e->getDtInicio()|truncate:19:"":true} - Data
Fim: {$e->getDtFim()|truncate:19:"":true}</td>
        </tr>
        {if $e->getNumRequisicoes() != 0}
          <tr>
            <td class="list_header1" colspan=4>Requisições: <a
href="{php}print(Util::getURL('requisicaoChat.load'));{/php}&cod={$e->getCod()}">{$e-
>getNumRequisicoes()}</a></td>
          </tr>
        {/if}
      </table>
      <br>
      {assign var="chave" value=0}

   {/foreach}

   {foreach from=$lista item=c}

      <table class="list_peq" border='0' align='center' width='90%' cellpadding="5" cellspacing="0"
style="border: solid 1px #CCCCCC;">
        <tr>
          <td class="list_header" align="left" width="40%">{$c->getNome()}</td>
          <td class="list_header" align="right" width="30%">{$c->getNumPessoas()} Pessoa(s)
Participando</td>
```

```
            <td class="list_header" align="center" width="30%"><img
src="{$smarty.const._IMAGEM_DIR_}entrarChat.jpg" onClick="javascript:
window.open('{php}print(Util::getURL('chat.index'));{/php}&cod_chat={$c->getCod()}', '{$c->getCod()}',
'toolbar=no, location=no, directories=no, status=no, menubar=no, scrollbars=no, resizable=yes,
width=800, height=700, top=0, left=0');"></td>
         </tr>
         <tr colspan=4>
            <td class="list_body1" colspan=4>{$c->getDescricao()}</td>
         </tr>
         <tr>
            <td class="list_body1" colspan=4>Criado em {$c->getDtCriacao()|truncate:19:"":true}</td>
         </tr>
         <tr>
            <td class="list_body1" colspan=4>Data Início: {$c->getDtInicio()|truncate:19:"":true} - Data
Fim: {$c->getDtFim()|truncate:19:"":true}</td>
         </tr>
      </table>
      <br>
      {assign var="chave" value=0}

   {/foreach}

   {foreach from=$listMod item=z}

      <table class="list_peq" border='0' align='center' width='90%' cellpadding="5" cellspacing="0"
style="border: solid 1px #CCCCCC;">
         <tr>
            <td class="list_header" align="left" width="40%">{$z->getNome()}</td>
            <td class="list_header" align="right" width="30%">{$z->getNumPessoas()} Pessoa(s)
Participando</td>
            <td class="list_header" height="31" align="center" width="30%"><font
color="#A35124">Aguardando Autorização</font></td>
         </tr>
         <tr colspan=4>
            <td class="list_body2" colspan=4>{$z->getDescricao()}</td>
         </tr>
         <tr>
            <td class="list_body2" colspan=4>Criado em {$z->getDtCriacao()|truncate:19:"":true}</td>
         </tr>
         <tr>
            <td class="list_body2" colspan=4>Data Início: {$z->getDtInicio()|truncate:19:"":true} - Data
Fim: {$z->getDtFim()|truncate:19:"":true}</td>
         </tr>
      </table>
      <br>
      {assign var="chave" value=0}

   {/foreach}

   {foreach from=$list item=t}

      <table class="list_peq" border='0' align='center' width='90%' cellpadding="5" cellspacing="0"
style="border: solid 1px #CCCCCC;">
         <tr>
            <td class="list_header" align="left" width="40%">{$t->getNome()}</td>
            <td class="list_header" align="right" width="30%">{$t->getNumPessoas()} Pessoa(s)
Participando</td>
            <td class="list_header" align="center" width="30%"><img
src="{$smarty.const._IMAGEM_DIR_}participarChat.gif" onClick="javascript:
location.href='{php}print(Util::getURL('requisicaoChat.save'));{/php}&cod={$t->getCod()}&op=1'"></td>
```

```
            </tr>
            <tr colspan=4>
               <td class="list_body2" colspan=4>{$t->getDescricao()}</td>
            </tr>
            <tr>
               <td class="list_body2" colspan=4>Criado em {$t->getDtCriacao()|truncate:19:"":true}</td>
            </tr>
            <tr>
               <td class="list_body2" colspan=4>Data Início: {$t->getDtInicio()|truncate:19:"":true} - Data
Fim: {$t->getDtFim()|truncate:19:"":true}</td>
            </tr>
         </table>
         <br>
         {assign var="chave" value=0}

      {/foreach}

      {if $chave != 0}
         <table align="center" class="empty">
            <tr>
               <td align="center" colspan="2"
class="message"><br>{$smarty.const.MESSAGE_EMPTY_LIST}<br><br></td>
            </tr>
         </table>
      {/if}

      </td>
</tr>
```

16.9. chat/load.tpl

```
{if $chat->getCod() >= 0}

   {if $chat->getCod() == 0}
      <tr>
         <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_15.jpg" width="780"
height="85"></td>
      </tr>
   {else}
      <tr>
         <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_63.jpg" width="780"
height="85"></td>
      </tr>
   {/if}
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
         <td colspan="8">
   <br>
   <table class="form_peq" align="center">
      <form name="form" method="post" action="{php}print(Util::getURL("chat.save"));{/php}"
enctype="multipart/form-data">
      <input type="hidden" name="cod" value="{$chat->getCod()}">
      <tr>
         <td class="form_label">Nome do Chat:</td>
         <td class="form_field"><input type="text" name="nome" size="60" maxlength="30"
value="{$chat->getNome()}"></td>
      </tr>
      <tr>
         <td class="form_label">Descrição:</td>
         <td class="form_field"><textarea rows="3" cols="45" name="descricao"
maxlength="10">{$chat->getDescricao()}</textarea></td>
      </tr>
      <tr>
         <td class="form_label">Banner Publicitário:</td>
         <td class="form_field"><input type="file" name="banner" size="40" class="botao"></td>
      </tr>
      <tr>
         <td class="form_label">Data Início:</td>
         <td class="form_field"><input type="text" name="dtInicio" size="10" maxlength="10"
value="{$chat->getDtInicio()}"></td>
      </tr>
      <tr>
         <td class="form_label">Data Fim:</td>
         <td class="form_field"><input type="text" name="dtFim" size="10" maxlength="10"
value="{$chat->getDtFim()}"></td>
      </tr>
      <input type="hidden" name="codUsuarioChat" value="{$chat->getCodUsuarioChat()}">
      <tr align="center">
         <td colspan="2" class="form_field"><input type="submit" class="button" value="Salvar">

         {if $chat->getCod() != 0}

            <input type="button" class="button" value="Excluir"
onclick="javascript:confirmarExclusao('{php}print(Util::getURL("chat.delete"));{/php}&cod={$chat-
>getCod()}');">

         {/if}

            
         </td>
```

```
        </tr>
        {if $errors|@count > 0}

        <tr>
            <td colspan="2" class="error">
            <div class="error">

            {foreach from=$errors item=error}
            {$error}<br>
            {/foreach}

            </div>
            </td>
        </tr>
        {/if}
        </form>
    </table>

    {literal}
    <script language="javascript">
    function confirmarExclusao(url) {
            if (confirm('Este Chat será excluído.\nVocê confirma esta ação?')) {
                window.location = url;
            }
    }
    </script>
    {/literal}
{else}
    <tr>
            <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_15.jpg" width="780"
height="85"></td>
     </tr>
     <tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
                <td align="center" colspan="8">
            <br>
            <br>
            Chat Inexistente!
            <br>
            <br>
          </td>
    </tr>
{/if}
```

16.10. chat/princ.tpl

```
<meta HTTP-EQUIV="Refresh" CONTENT="1;url=../modulos/chat/templates/chat/form.htm">
```

16.11. chat/requisicao.tpl

```
<html>
<link rel="stylesheet" type="text/css" href="style.css"/>
<body bgcolor="#D5DDE3">
 <table>
  <tr>
   <td class="form_field">
     Requisições: {$req_ant}
   </td>
  </tr>
 </table>
</body>
</html>
```

16.12. chat/sair.tpl

```
<html>

<head>
<link rel="stylesheet" type="text/css" href="style.css"/>
<script language="javascript">
{literal}
function Fechar(){

window.open("{/literal}{php}print(Util::getURL("chat.sair"));{/php}{literal}&codChat={/literal}{$cod_chat}{
literal}", "Sair", "toolbar=no, location=no, directories=no, status=no, menubar=no, scrollbars=no,
resizable=yes, width=200, height=20, top=0, left=0");
        window.top.close();
}
{/literal}
</script>
</head>

<body bgcolor="#006699">
<br><div align="center"><font face="Tahoma" color="#FFFFFF" size="2"><b>Saindo do Chat...
<br>Obrigado pela Participação!</b></font></div>
<script language="JavaScript">
setTimeout("javascript:Fechar();", "3000");
</script>
</body>
</html>
```

16.13. chat/topo.tpl

```
{* Frame de Habilitação Recursos do Chat *}

<html>
<head>
<link rel="stylesheet" type="text/css" href="style.css"/>
<script language="javascript">
{literal}
function SairChat(){
        this.top.close();
}
{/literal}
</script>
</head>

<body topmargin="0" leftmargin="0" background="{$smarty.const._IMAGEM_DIR_}sup_fundo.gif">

<table background="{$smarty.const._IMAGEM_DIR_}sup_fundo.gif" width="100%">
<form name="form">
{if $banner != ""}
<tr>
   <td width="15%" class="topo_chat" rowspan="3" align="left"><img
src="{$smarty.const._IMAGEM_DIR_}logo.jpg">
   <td width="15%" class="topo_chat" rowspan="3" align="center"><img border="1"
bordercolor="#CCCCCC" src="{$banner}" width="460" height="90"></td>
   <td width="15%" class="topo_chat" align="right"> <img
src="{$smarty.const._IMAGEM_DIR_}sair.png" onClick="SairChat();"></td>
</tr>

<tr>
   <td width="25%" class="topo_chat" align="right" valign="bottom">Habilitar Som<input
type=checkbox name="som" checked></td>
</tr>

<tr>
   <td class="topo_chat" align="right">Rolagem Automática<input type=checkbox name="rolagem"
checked></td>
</tr>
{else}

<tr>
   <td width="15%" class="topo_chat" rowspan="3" align="left"><img
src="{$smarty.const._IMAGEM_DIR_}logo.jpg">
   <td width="45%" class="topo_chat" rowspan="3" align="center"></td>
   <td width="25%" class="topo_chat" align="right">
   <img src="{$smarty.const._IMAGEM_DIR_}sair.png" onClick="SairChat();">
   </td>
</tr>

<tr>
   <td class="topo_chat" align="right" valign="bottom">Habilitar Som<input type=checkbox
name="som" checked></td>
</tr>

<tr>
   <td class="topo_chat" align="right">Rolagem Automática<input type=checkbox name="rolagem"
checked></td>
</tr>
```

```
{/if}
</form>
</table>

</body>
</html>
```

16.14. chat/upload.tpl

```
<html>

<head>
	{literal}
<script>
<!--

var W3CDOM = (document.createElement && document.getElementsByTagName);

function init()
{
	if (!W3CDOM) return;
	var fakeFileUpload = document.createElement('div');
	fakeFileUpload.className = 'fakefile';
	fakeFileUpload.appendChild(document.createElement('input'));
	var image = document.createElement('img');
	image.src='pix/button_select.gif';
	fakeFileUpload.appendChild(image);
	var x = document.getElementsByTagName('input');
	for (var i=0;i<x.length;i++)
	{
		if (x[i].type != 'file') continue;
		if (x[i].getAttribute('noscript')) continue;
		if (x[i].parentNode.className != 'fileinputs') continue;
		x[i].className = 'file hidden';
		var clone = fakeFileUpload.cloneNode(true);
		x[i].parentNode.appendChild(clone);
		x[i].relatedElement = clone.getElementsByTagName('input')[0];
		if (x[i].value)
			x[i].onchange();
		x[i].onchange = x[i].onmouseout = function () {
			this.relatedElement.value = this.value;
		}
	}
}

// -->

</script>
	{/literal}

  <title>Envio do Arquivo</title>
  <link rel="stylesheet" type="text/css" href="style.css"/>
</head>

<body bgcolor="#D5DDE3" leftmargin="0" topmargin="0" marginwidth="0" marginheight="0">

<table >

   <form name="upload" method="post"
action="{php}print(Util::getURL("chat.arquivo"));{/php}&cod_chat={$cod_chat}&cod_usuario={$cod_us
uario}&cod=1" enctype="multipart/form-data">

	<tr>
	   <td class="list_header" align="center" colspan="2"><b>ENVIO DE ARQUIVO</b></td>
	</tr>

	<tr>
```

```
            <td class="list_upload">
                Enviar Para:  
                <select name="destinatario" title="Menu das pessoas">
                    {html_options values=$usuarios output=$usuarios}
                </select>
                <br><br>
                <input type="file" name="arquivo" size="40" class="botao">
                <br><br>

                <center><input type="submit" value="Enviar" class="button"></center>

            </td>
        </tr>

    </form>

</table>

</body>
</html>
```

16.15. chat/usuarios.tpl

```
{* Lista de Usuários do Chat *}

<html>
<head>

<link rel="stylesheet" type="text/css" href="style.css"/>

<script language="JavaScript">
{literal}
function TrocarUser(valor) {
 top.inferior.document.getElementById("a_dest").innerHTML = valor;
 top.inferior.document.getElementById("destinatario").value = valor;
}
{/literal}
</script>
</head>
<body topmargin="0" leftmargin="0"
background="{$smarty.const._IMAGEM_DIR_}usuarios_fundo.gif">
<br>
<div align="center">
<table>
{foreach from=$usuario item=c}
    <tr>
        <td align="center" colspan=4 class="usuarios_chat">
            <a href="javascript:TrocarUser('{$c}');">{$c}</a>
        </td>
    </tr>
{/foreach}
</table>
</div>
</body>
</html>
```

16.16. chat/usuários_list.tpl

```
<html>
<head>
<script language="JavaScript">
{literal}
function Redirect() {
     location =
"{/literal}{php}print(Util::getURL("chat.usuarios"));{/php}{literal}&cod_chat={/literal}{$codChat}{literal}";
}
{/literal}
</script>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#CCCCCC" onLoad="javascript: Redirect();">
</body>
</html>
```

16.17. chatUsuario/list.tpl

```
<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_35.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
        <br>
<form name="form" method="post"
action="{php}print(Util::getURL("chatUsuario.load"));{/php}&cod=1">
<table align="center" class="form">

<tr>
   <td align="center">
      <table class="list">

         <tr>
            <td class="list_header" align="center">Escolha da Sala para a Exibição dos Detalhes de
Acesso</td>
         </tr>

         <tr>
            <td class="list_body2" align="center">
               <select name="chat[]" multiple>
                  {foreach from=$arrObjChat item=chat}
                     {html_options values=$chat->getCod() output=$chat->getNome()}
                  {foreachelse}
                     Não há grupo cadastrado, por favor cadastre pelo menos um grupo
                  {/foreach}
               </select>
            </td>
         </tr>

      </table>
   </td>
</tr>

<tr>
   <td class="form_field" align="center">
      <input type="submit" value="Detalhes" class="button">
   </td>
</tr>

{if $errors|@count > 0}

<tr><td colspan="2" class="error"><div class="error">

   {foreach from=$errors item=error}
      {$error}<br>
   {/foreach}

</div></td></tr>

{/if}

</table>

</td>
</tr>
```

16.18. chatUsuario/load.tpl

```
{if $codChat != 0}

<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_32.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
   <br>

   <form name="form" method="post" action="{php}print(Util::getURL("chatUsuario.save"));{/php}">
   <table align="center" class="form">

   <tr>
      <td colspan="2">

         <table class="list">

            <tr>
               <td class="list_header" width="45%">Usuários</td>
               <td class="list_header" width="25%">Grupos</td>
            </tr>

            <tr>

               <input type="hidden" name="codChat" value="{$codChat}">
               <input type="hidden" name="nomeChat" value="{$nomeChat}">

               <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                  {foreach from=$arrUsuario item=usuario}
                     <label>
                        <input type="checkbox" name="ckUsuario[]" value="{$usuario-
>getCod()}">{$usuario->getNome()}
                     </label><br>
                  {foreachelse}
                     Nenhum Usuário Cadastrado ou Todos os Usuários já foram Cadastrados no Chat.
                  {/foreach}
               </td>

               <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                  <select name="ckGrupo[]">
                     <option value="" selected>SELECIONE</option>
                     {foreach from=$arrGrupo item=grupo}
                        {html_options values=$grupo->getCod() output=$grupo->getNome()}
                     {/foreach}
                  </select>
               </td>

            </tr>

         </table>

      </td>
   </tr>

   <tr>
      <td class="form_field" align="center">
         <input type="submit" value="Salvar" class="button">
```

```
        <input type="button" class="button" value="Finalizar"
onclick="javascript:location='{php}print(Util::getURL("chat.list"));{/php}'">
      </td>
   </tr>

   <br>

   <tr>
   {if $chave == 1 }
      <tr>
         <td>
            <table class="list">

               <tr>
                  <td class="list_header" colspan="3" align="center">Configurações Adicionadas</td>
               </tr>

               <tr>
                  <td class="list_header">Sala</td>
                  <td class="list_header">Usuário</td>
                  <td class="list_header">Grupo</td>
               </tr>

               {foreach from=$lista item=acesso name=loop key=ind}

                  <input type="hidden" name="listaArr[{$ind}][codChat]" value="{$acesso-
>getCodChat()}">
                  <input type="hidden" name="listaArr[{$ind}][codUsuario]" value="{$acesso-
>getCodUsuario()}">
                  <input type="hidden" name="listaArr[{$ind}][codGrupo]" value="{$acesso-
>getCodGrupo()}">

                  <tr>
                     <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                        {$acesso->getCodChat()}
                     </td>
                     <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                        {$acesso->getCodUsuario()}
                     </td>
                     <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                        {$acesso->getCodGrupo()}
                     </td>
                  </tr>

               {/foreach}

            </table>
         </td>
      </tr>
   {/if}


   {if $errors|@count > 0}

   <tr><td colspan="2" class="error"><div class="error">

      {foreach from=$errors item=error}
         {$error}<br>
      {/foreach}
```

```
    </div></td></tr>

    {/if}

    </table>
    </form>

{elseif $cod == 1}

<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_67.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
    <br>

    <form name="form" method="post"
action="{php}print(Util::getURL("chatUsuario.save"));{/php}&cod=1">
    <table align="center" class="form">

    <tr>
       <td colspan="2">

          <table class="list">

             <tr>
                <td class="list_header" width="17%">Sala</td>
                <td class="list_header" width="45%">Usuário / Grupo</td>
                <td class="list_header" width="25%">Grupo</td>
             </tr>

             {foreach from=$listaAcesso item=lista name=loop key=ind}
             <tr>

                <input type="hidden" name="arrCodChat[{$ind}][codChat]" value="{$lista.codChat}">

                <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                   <label>
                      <input type="checkbox" name="chat[]" value="{$lista.codChat}">{$lista.nomeChat}
                   </label>
                </td>
                <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                   {foreach from=$lista.usuarioGrupo item=usuarioGrupo}
                      {if $usuarioGrupo.nomeGrupo != "Criador"}
                         <label>
                            <input type="checkbox" name="codUsuario[{$lista.codChat}][]"
value="{$usuarioGrupo.codUsuario} - {$usuarioGrupo.codGrupo}">{$usuarioGrupo.nomeUsuario} -
{$usuarioGrupo.nomeGrupo}
                         </label>
                         <br>
                      {/if}
                   {/foreach}
                </td>
                <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                   <select name="codGrupo[]">
                      <option value="" selected>SELECIONE</option>
                      {foreach from=$lista.arrGrupo item=grupo}
                         {html_options values=$grupo->getCod() output=$grupo->getNome()}
                      {/foreach}
```

```
                    </select>
                </td>
            </tr>
            {/foreach}

        </table>

    </td>
</tr>

<tr>
    <td class="form_field" align="center">
        <input type="submit" value="Salvar" class="button">
        <input type="button" class="button" value="Novo"
onclick="javascript:location='{php}print(Util::getURL("chatUsuario.load"));{/php}&codChat={$chat}'">
        <input type="button" class="button" value="Exclusão"
onclick="javascript:location='{php}print(Util::getURL("chatUsuario.delete"));{/php}&cod=0&listaChat={$l
istaChat}'">
        <input type="button" class="button" value="Finalizar"
onclick="javascript:location='{php}print(Util::getURL("chat.list"));{/php}'">
    </td>
</tr>

{if $errors|@count > 0}

<tr><td colspan="2" class="error"><div class="error">

    {foreach from=$errors item=error}
        {$error}<br>
    {/foreach}

</div></td></tr>

{/if}

</table>
</form>

{else}

<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_38.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
<br>

<form name="form" method="post"
action="{php}print(Util::getURL("chatUsuario.delete"));{/php}&cod=1&listaChat={$listaChat}">
<table align="center" class="form">

<tr>
    <td colspan="2">

        <table class="list">

            <tr>
                <td class="list_header" width="35%">Sala</td>
                <td class="list_header" width="65%">Usuário / Grupo</td>
```

```
            </tr>

            {foreach from=$listaAcesso item=lista name=loop key=ind}
            <tr>

               <input type="hidden" name="arrCodChat[{$ind}][codChat]" value="{$lista.codChat}">

               <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                  <label>
                     <input type="checkbox" name="codChat[]"
value="{$lista.codChat}">{$lista.nomeChat}
                  </label>
               </td>
               <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                  {foreach from=$lista.usuarioGrupo item=usuarioGrupo}
                     {if $usuarioGrupo.nomeGrupo != "Criador"}
                        <label>
                           <input type="checkbox" name="codUsuario[{$lista.codChat}][]"
value="{$usuarioGrupo.codUsuario} - {$usuarioGrupo.codGrupo}">{$usuarioGrupo.nomeUsuario} -
{$usuarioGrupo.nomeGrupo}
                        </label>
                        <br>
                     {/if}
                  {/foreach}
               </td>
            </tr>
            {/foreach}

         </table>

      </td>
   </tr>

   <tr>
      <td class="form_field" align="center">
         <input type="submit" value="Excluir" class="button">
         <input type="button" class="button" value="Finalizar"
onclick="javascript:location='{php}print(Util::getURL("chat.list"));{/php}'">
      </td>
   </tr>

   {if $errors|@count > 0}

   <tr><td colspan="2" class="error"><div class="error">

      {foreach from=$errors item=error}
         {$error}<br>
      {/foreach}

   </div></td></tr>

   {/if}

   </table>
   </form>

{/if}

</td>
</tr>
```

16.19. configChat/list.tpl

```
<p class="title">Configuração Chat - Lista</p>
<table class="list">

<tr>
  <td width="33%"  class="list_header">Chat</td>
  <td width="33%" class="list_header">Recurso</td>
  <td width="33%" class="list_header">Grupo</td>

</tr>

{foreach from=$lista item=c name=loop}

   <tr>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}">{$c->getCodChat()}</td>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}">{$c->getCodRecurso()}</td>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}">{$c->getCodGrupo()}</td>
   </tr>

{foreachelse}

   <tr>
      <td colspan="3" class="message">
         <div class="message">
            {$smarty.const.MESSAGE_EMPTY_LIST}
         </div>
      </td>
   </tr>

{/foreach}

   <tr>
      <td colspan="3" class="list_header" align="center">
         <input type="button" class="button" value="Cadastrar"
onclick="javascript:location='{php}print(Util::getURL("configChat.load"));{/php}&cod=0'">
         <input type="button" class="button" value="Editar"
onclick="javascript:location='{php}print(Util::getURL("configChat.load"));{/php}&cod=1'">
      </td>
   </tr>

</table>
```

16.20. configChat/load.tpl

```
{if $codChat != 0 AND $cod == 0}

<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_62.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
   <br>

   <form name="form" method="post" action="{php}print(Util::getURL("configChat.save"));{/php}">
      <table align="center" class="form">

         <tr>
            <td colspan="2">

               <!-- INÍCIO Tabela de Sala - Recurso - Grupo -->
               <table class="list">
                  <tr>
                     <td class="list_header" width="35%">Grupo</td>
                     <td class="list_header" width="65%">Recurso</td>
                  </tr>

                  <tr>

                     <td class="list_body2">

                        <!-- INÍCIO Construção do formulário de Grupo -->
                        <select name="ckGrupo[]">
                           <option value="" selected>SELECIONE</option>
                           {foreach from=$arrObjGrupo item=grupo}
                              {html_options values=$grupo->getCod() output=$grupo->getNome()}
                           {/foreach}
                        </select>

                     </td>

                     <td class="list_body2">

                     <input type="hidden" name="codChat" value="{$codChat}">
                     <input type="hidden" name="nomeChat" value="{$nomeChat}">

                        <!-- INÍCIO Construção do formulário de Recursos -->
                        {foreach from=$arrObjRecurso item=recurso}

                           <label>
                           <input type="checkbox" name="ckRecurso[]" value="{$recurso-
>getCod()}">{$recurso->getNome()}
                           </label><br>

                        {foreachelse}
                           Não há Recurso cadastrado, por favor cadastre pelo menos um recurso
                        {/foreach}
                        <!-- FIM Construção do formulário de Recurso -->

                     </td>

                  </tr>
```

```
            </table>
            <!-- FIM Tabela de Sala - Recurso - Grupo -->

        </td>

    </tr>

    <tr>
        <td class="form_field" colspan="3" align="center">
            <input type="submit" class="button" value="Salvar">
            <input type="button" class="button" value="Finalizar"
onclick="javascript:location='{php}print(Util::getURL("chatUsuario.load"));{/php}&codChat={$codChat}"'
>
        </td>
    </tr>

    <br>

    <tr>
    {if $chave == 1 }
        <tr>
            <td>
                <table class="list">

                    <tr>
                        <td class="list_header" colspan="3" align="center">Configurações
Adicionadas</td>
                    </tr>

                    <tr>
                        <td class="list_header">Sala</td>
                        <td class="list_header">Recurso</td>
                        <td class="list_header">Grupo</td>
                    </tr>

                    {foreach from=$lista item=config name=loop key=ind}

                        <input type="hidden" name="listaArr[{$ind}][codChat]" value="{$config-
>getCodChat()}">
                        <input type="hidden" name="listaArr[{$ind}][codRecurso]" value="{$config-
>getCodRecurso()}">
                        <input type="hidden" name="listaArr[{$ind}][codGrupo]" value="{$config-
>getCodGrupo()}">

                        <tr>
                            <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                                {$config->getCodChat()}
                            </td>
                            <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                                {$config->getCodRecurso()}
                            </td>
                            <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                                {$config->getCodGrupo()}
                            </td>
                        </tr>

                    {/foreach}

                </table>
```

```
                </td>
            </tr>
        {/if}
        </tr>

        {if $errors|@count > 0}

        <tr><td colspan="2" class="error"><div class="error">

            {foreach from=$errors item=error}
                {$error}<br>
            {/foreach}

        </div></td></tr>

        {/if}

    </table>

</form>


{elseif $cod == 1}

<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_66.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
<br>

<form name="form" method="post" action="{php}print(Util::getURL("configChat.delete"));{/php}">
    <table align="center" class="form">

        <tr>
            <td colspan="2">

                <!-- INÍCIO Tabela de Sala - Recurso - Grupo -->
                <table class="list">
                    <tr>
                        <td class="list_header">Sala</td>
                        <td class="list_header">Recurso</td>
                        <td class="list_header">Grupo</td>
                    </tr>

                    {foreach from=$arrObj item=config name=loop key=i}

                        {if $config->getCodGrupo() != 'Criador'}

                            {assign var="codigo" value=$configChat[$i]}

                            <tr>

                                <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                                    <label>
                                    <input type="checkbox" name="ckConfigChat[]" value="{$codigo-
>getCodChat()}-{$codigo->getCodRecurso()}-{$codigo->getCodGrupo()} ">{$config->getCodChat()}
                                    </label><br>
                                </td>
```

```
                        <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                            <label>
                            {$config->getCodRecurso()}
                            </label><br>
                        </td>

                        <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                            <label>
                            {$config->getCodGrupo()}
                            </label><br>
                        </td>
                    </tr>

                {/if}

            {/foreach}

        </table>
        <!-- FIM Tabela de Sala - Recurso - Grupo -->

    </td>

</tr>

<tr>
    <td class="form_field" colspan="3" align="center">
        <input type="submit" class="button" value="Excluir">
        <input type="button" class="button" value="Novo"
onclick="javascript:location='{php}print(Util::getURL("configChat.load"));{/php}&codChat={$chat}&cod=
2'">
        <input type="button" class="button" value="Finalizar"
onclick="javascript:location='{php}print(Util::getURL("chatUsuario.load"));{/php}&chat={$chat}'">
    </td>
</tr>

</table>

</form>

{else}

<tr>
    <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_62.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
    <td colspan="8">
<br>

<form name="form" method="post"
action="{php}print(Util::getURL("configChat.save"));{/php}&cod=2">
<table align="center" class="form">

    <tr>
        <td colspan="2">

        <!-- INÍCIO Tabela de Sala - Recurso - Grupo -->
        <table class="list">
            <tr>
```

```
                    <td class="list_header" width="35%">Grupo</td>
                    <td class="list_header" width="65%">Recurso</td>
                </tr>

                <tr>

                    <td class="list_body2">

                        <!-- INÍCIO Construção do formulário de Grupo -->
                        <select name="ckGrupo[]">
                            <option value="" selected>SELECIONE</option>
                            {foreach from=$arrObjGrupo item=grupo}
                                {html_options values=$grupo->getCod() output=$grupo->getNome()}
                            {/foreach}
                        </select>

                    </td>

                    <td class="list_body2">

                    <input type="hidden" name="codChat" value="{$codChat}">
                    <input type="hidden" name="nomeChat" value="{$nomeChat}">

                        <!-- INÍCIO Construção do formulário de Recursos -->
                        {foreach from=$arrObjRecurso item=recurso}

                            <label>
                            <input type="checkbox" name="ckRecurso[]" value="{$recurso-
>getCod()}">{$recurso->getNome()}
                            </label><br>

                        {foreachelse}
                            Não há Recurso cadastrado, por favor cadastre pelo menos um recurso
                        {/foreach}
                        <!-- FIM Construção do formulário de Recurso -->

                    </td>

                </tr>

            </table>
            <!-- FIM Tabela de Sala - Recurso - Grupo -->

        </td>

    </tr>

    <tr>
        <td class="form_field" colspan="3" align="center">
            <input type="submit" class="button" value="Salvar">
            <input type="button" class="button" value="Finalizar"
onclick="javascript:location='{php}print(Util::getURL("chatUsuario.load"));{/php}&chat={$codChat}"'>
        </td>
    </tr>

    <br>

    <tr>
    {if $chave == 1 }
        <tr>
```

```
                <td>
                    <table class="list">

                        <tr>
                            <td class="list_header" colspan="3" align="center">Configurações
Adicionadas</td>
                        </tr>

                        <tr>
                            <td class="list_header">Sala</td>
                            <td class="list_header">Recurso</td>
                            <td class="list_header">Grupo</td>
                        </tr>

                        {foreach from=$lista item=config name=loop key=ind}

                            <input type="hidden" name="listaArr[{$ind}][codChat]" value="{$config-
>getCodChat()}">
                            <input type="hidden" name="listaArr[{$ind}][codRecurso]" value="{$config-
>getCodRecurso()}">
                            <input type="hidden" name="listaArr[{$ind}][codGrupo]" value="{$config-
>getCodGrupo()}">

                            <tr>
                                <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                                    {$config->getCodChat()}
                                </td>
                                <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                                    {$config->getCodRecurso()}
                                </td>
                                <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                                    {$config->getCodGrupo()}
                                </td>
                            </tr>

                        {/foreach}

                    </table>
                </td>
            </tr>
        {/if}
        </tr>

        {if $errors|@count > 0}

        <tr><td colspan="2" class="error"><div class="error">

            {foreach from=$errors item=error}
                {$error}<br>
            {/foreach}

        </div></td></tr>
        {/if}
    </table>

</form>
{/if}
</td>
</tr>
```

16.21. recurso/list.tpl

```
<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_36.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
        <br>

  {assign var="chave" value=0}
  {foreach from=$lista item=c name=loop}
    {if  $chave == 0 }
      <table class="form_peq" align="center">
      <tr>
        <td width="100%" colspan="2" class="list_header">Nome do Recurso</td>
      </tr>
      {assign var="chave" value=1}
    {/if}
    <tr>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}" align="left" width="80%">{$c-
>getNome()}</td>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}" align="center"><input type="button"
class="button" value="Editar" onClick="javascript:
location.href='{php}print(Util::getURL('recurso.load'));{/php}&cod={$c->getCod()}'"></td>
    </tr>
  {foreachelse}

    <table align="center" class="empty">
    <tr>
      <td align="center" colspan="3"
class="message"><br>{$smarty.const.MESSAGE_EMPTY_LIST}<br><br></td>
    </tr>
    </table>

  {/foreach}
  {if $chave == 1}
    </table>
  {/if}

  </td>
</tr>
```

16.22. recurso/load.tpl

```
{if $recurso->getCod() >= 0}

   {if $recurso->getCod() == 0}
      <tr>
         <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_35.jpg" width="780"
height="85"></td>
      </tr>
   {else}
      <tr>
         <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_65.jpg" width="780"
height="85"></td>
      </tr>
   {/if}

   <tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
         <td colspan="8">
         <br>
         <form name="form" method="post" action="{php}print(Util::getURL("recurso.save"));{/php}">
         <input type="hidden" name="cod" value="{$recurso->getCod()}">
         <table class="form_peq" align="center">
            <tr>
               <td class="form_label">Nome do Recurso:</td>
               <td class="form_field"><input type="text" name="nome" size="60" maxlength="20"
value="{$recurso->getNome()}"></td>
            </tr>
            <tr>
               <td colspan="2" align="center" class="form_field">
                  <input type="submit" class="button" value="Salvar">
                  {if $recurso->getCod() != 0}
                        <input type="button" class="button" value="Excluir"
onclick="javascript:confirmarExclusao('{php}print(Util::getURL("recurso.delete"));{/php}&cod={$recurso
->getCod()}');">
                  {/if}
               </td>
            </tr>


            {if $errors|@count > 0}

            <tr>
               <td colspan="2" class="error">
                  <div class="error">

               {foreach from=$errors item=error}
                  {$error}<br>
               {/foreach}

                  </div>
               </td>
            </tr>

            {/if}

            {literal}
            <script language="javascript">
              function confirmarExclusao(url) {
                 if (confirm('Este Recurso será excluído.\nVocê confirma esta ação?')) {
                    window.location = url;
```

```
                }
            }
            </script>
            {/literal}

        </table>
    </td>
  </tr>
</form>

{else}
  </td>

  <tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_35.jpg" width="780"
height="85"></td>
  </tr>

  <tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
          <td align="center" colspan="8">
        <br>
        <br>
        Recurso Inexistente!
        <br>
        <br>
    </td>
  </tr>
{/if}
</td>
</tr>
```

16.23. requisicaoChat/list.tpl

```
<tr>
        <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_36.jpg" width="780"
height="85"></td>
</tr>
<tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
        <td colspan="8">
        <br>

   {foreach from=$lista item=c name=loop}
   <table class="form_peq" align="center">
   <tr>
     <td width="35%" class="list_header">Nome do Recurso</td>
     <td width="55%" colspan="2" class="list_header">Lista de Ações</td>
   </tr>
   <tr>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}" align="left">{$c->getNome()}</td>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}" align="left">{$c->getAcao()}</td>
      <td class="list_body{$smarty.foreach.loop.iteration%2+1}" align="center"><input type="button"
class="button" value="Editar" onClick="javascript:
location.href='{php}print(Util::getURL('recurso.load'));{/php}&cod={$c->getCod()}'"></td>
   </tr>
   </table>
   {foreachelse}

   <table align="center" class="empty">
   <tr>
      <td align="center" colspan="2"
class="message"><br>{$smarty.const.MESSAGE_EMPTY_LIST}<br><br></td>
   </tr>
   </table>

   {/foreach}

   </td>
</tr>
```

16.24. requisicaoChat/load.tpl

```
{if $indice == 0 }
   <tr>
         <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_77.jpg" width="780"
height="85"></td>
   </tr>
   <tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
         <td colspan="8">
      <br>

      <form name="form" method="post"
action="{php}print(Util::getURL("requisicaoChat.save"));{/php}">
      <table align="center" class="form">

      <tr>
         <td colspan="2">

            <table class="list">

               <tr>
                  <td class="list_header" width="45%">Usuários</td>
                  <td class="list_header" width="25%">Grupos</td>
               </tr>

               <tr>

                  <input type="hidden" name="codChat" value="{$codChat}">
                  <input type="hidden" name="indice" value="0">

                  <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                     {foreach from=$arrUsuario item=usuario}
                        <label>
                           <input type="checkbox" name="ckUsuario[]" value="{$usuario[0]}">{$usuario[1]}
                        </label><br>
                     {foreachelse}
                        Não há mais nenhum Usuário aguardando liberação.
                     {/foreach}
                  </td>

                  <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                     <select name="ckGrupo[]">
                        <option value="" selected>SELECIONE</option>
                        {foreach from=$arrGrupo item=grupo}
                           {html_options values=$grupo->getCod() output=$grupo->getNome()}
                        {/foreach}
                     </select>
                  </td>

               </tr>

            </table>

         </td>
      </tr>

      <tr>
         <td class="form_field" align="center">
            <input type="submit" value="Salvar" class="button">
```

```
            <input type="button" class="button" value="Recusar"
onclick="javascript:location='{php}print(Util::getURL("requisicaoChat.delete"));{/php}&codChat={$codC
hat}&cod=1&indice=1'">
            <input type="button" class="button" value="Cancelar"
onclick="javascript:location='{php}print(Util::getURL("chat.list"));{/php}'">
            <input type="button" class="button" value="Finalizar"
onclick="javascript:location='{php}print(Util::getURL("chat.list"));{/php}'">
         </td>
      </tr>

      <br>

      <tr>
      {if $chave == 1 }
         <tr>
            <td>
               <table class="list">

                  <tr>
                     <td class="list_header" colspan="3" align="center">Configurações Adicionadas</td>
                  </tr>

                  <tr>
                     <td class="list_header">Sala</td>
                     <td class="list_header">Usuário</td>
                     <td class="list_header">Grupo</td>
                  </tr>

                  {foreach from=$lista item=acesso name=loop key=ind}

                     <input type="hidden" name="listaArr[{$ind}][codChat]" value="{$acesso-
>getCodChat()}">
                     <input type="hidden" name="listaArr[{$ind}][codUsuario]" value="{$acesso-
>getCodUsuario()}">
                     <input type="hidden" name="listaArr[{$ind}][codGrupo]" value="{$acesso-
>getCodGrupo()}">

                     <tr>
                        <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                           {$acesso->getCodChat()}
                        </td>
                        <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                           {$acesso->getCodUsuario()}
                        </td>
                        <td class="list_body{$smarty.foreach.loop.iteration%2+1}" width="33%">
                           {$acesso->getCodGrupo()}
                        </td>
                     </tr>

                  {/foreach}

               </table>
            </td>
         </tr>
      {/if}


      {if $errors|@count > 0}

      <tr><td colspan="2" class="error"><div class="error">
```

```
            {foreach from=$errors item=error}
                {$error}<br>
            {/foreach}

        </div></td></tr>

        {/if}

        </table>
        </form>
{else if $indice == 1}
    <tr>
            <td colspan="8"><img src="{$smarty.const._IMAGEM_DIR_}sysconf_78.jpg" width="780"
height="85"></td>
    </tr>
    <tr background="{$smarty.const._IMAGEM_DIR_}sysconf_16.jpg">
            <td colspan="8">
        <br>

        <form name="form" method="post"
action="{php}print(Util::getURL("requisicaoChat.delete"));{/php}&cod=0">
        <table align="center" class="form">

        <tr>
            <td colspan="2">

                <table class="list">

                    <tr>
                        <td class="list_header" width="45%">Usuários</td>
                    </tr>

                    <tr>

                        <input type="hidden" name="codChat" value="{$codChat}">
                        <input type="hidden" name="indice" value="1">

                        <td class="list_body{$smarty.foreach.loop.iteration%2+1}">
                            {foreach from=$arrUsuario item=usuario}
                                <label>
                                    <input type="checkbox" name="ckUsuario[]" value="{$usuario[0]}">{$usuario[1]}
                                </label><br>
                            {foreachelse}
                                Não há mais nenhum Usuário aguardando liberação.
                            {/foreach}
                        </td>

                     </tr>

                </table>

            </td>
        </tr>

        <tr>
            <td class="form_field" align="center">
                <input type="submit" value="Recusar" class="button">
                <input type="button" class="button" value="Cancelar"
onclick="javascript:location='{php}print(Util::getURL("chat.list"));{/php}'">
```

```
            <input type="button" class="button" value="Finalizar"
onclick="javascript:location='{php}print(Util::getURL("chat.list"));{/php}'">
        </td>
    </tr>

    <br>

    <tr>

    {if $errors|@count > 0}

    <tr><td colspan="2" class="error"><div class="error">

        {foreach from=$errors item=error}
            {$error}<br>
        {/foreach}

    </div></td></tr>

    {/if}

    </table>
    </form>
{/if}
```

17. JAVA – ÁUDIO CONFERÊNCIA

17.1. AudioStream.java

```
/*
 *
 * ArrayStream.java
 *
 */
import java.io.*;


public class ArrayStream implements Serializable {

    private byte [] audio;

    // Criando uma nova instância para o ArrayStream
    public ArrayStream() {
    }

    public ArrayStream(byte [] audioData){
        audio = audioData;
    }

    public void setAudio(byte [] audioData){
        audio = audioData;
    }

    public byte [] getAudio(){
        return audio;
    }
}
```

17.2. Client.java

```
import java.io.*;
import java.net.*;
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import javax.sound.sampled.*;
import java.util.StringTokenizer;
import java.util.LinkedList;

public class Client extends JFrame {

    private JTextField enterField;
    private JTextArea displayArea;
    private DatagramPacket sendPacket, receivePacket, sendPacketRTP, sendPacketP;
    private DatagramSocket socket;
    private JLabel Texto, Texto1;
    private Font fonte;
    private JButton TBotao, TBotao1, TBotao2;
    private Panel panel;
    private String message, port, nome;
    private int porta;

    // Variaveis conexão TCP
    private ObjectOutputStream enviar;
    private ObjectInputStream receber;
    private Socket cliente;

    boolean stopCapture = false, play = false;
    ByteArrayOutputStream byteArrayOutputStream, streamObject;
    AudioFormat audioFormat;
    TargetDataLine targetDataLine;
    AudioInputStream audioInputStream;
    SourceDataLine sourceDataLine;
    ArrayStream arrayStreamObject, arrayStreamRecebido, arrayStreamTeste;

    // Declaração da Fila
    private LinkedList listaObjetos = new LinkedList();

    int chave = 0;
    String ipServidor = "200.17.208.177";
//    String ipServidor = "10.61.6.246";
    String codSala = "";

    // Criando a Tela e o DatagramSocket
    public Client() {

        super( "Conferência Áudio" );

        //codSala = System.getProperty("codigo");
        codSala = "12";

        Container container = getContentPane();
        container.setLayout(null);

        // Transformação de Cores RGB para HSB
        float [] HSB = Color.RGBtoHSB(51, 139, 190, (new float[3]));
        float [] HSB1 = Color.RGBtoHSB(247, 199, 91, (new float[3]));
        float [] HSB2 = Color.RGBtoHSB(55, 74, 102, (new float[3]));
```

```
float [] HSB3 = Color.RGBtoHSB(223, 138, 29, (new float[3]));
float [] HSB4 = Color.RGBtoHSB(217, 215, 212, (new float[3]));

// Cores do Container
container.setBackground(Color.getHSBColor(HSB4[0], HSB4[1], HSB4[2]));
container.setForeground(Color.white);

panel = new Panel();

panel.setLayout(null);

fonte = new Font("Verdana",Font.CENTER_BASELINE,12);
TBotao = new JButton("Conectar");
TBotao1 = new JButton("Pedir Palavra");
TBotao2 = new JButton("Falar");
Texto = new JLabel("Desconectado");
Texto1 = new JLabel("Você está ");

Texto.setFont(fonte);
Texto1.setFont(fonte);
TBotao.setFont(fonte);
TBotao1.setFont(fonte);
TBotao1.setEnabled(false);
TBotao2.setFont(fonte);
TBotao2.setVisible(false);

// Cores dos Botões
TBotao.setBackground(Color.getHSBColor(HSB1[0], HSB1[1], HSB1[2]));
TBotao.setForeground(Color.getHSBColor(HSB2[0], HSB2[1], HSB2[2]));
TBotao1.setBackground(Color.getHSBColor(HSB1[0], HSB1[1], HSB1[2]));
TBotao1.setForeground(Color.getHSBColor(HSB2[0], HSB2[1], HSB2[2]));
TBotao2.setBackground(Color.getHSBColor(HSB1[0], HSB1[1], HSB1[2]));
TBotao2.setForeground(Color.getHSBColor(HSB2[0], HSB2[1], HSB2[2]));

// Posições dos Botões
TBotao.setBounds(10,10,93,20);
TBotao1.setBounds(120,10,128,20);
TBotao2.setBounds(137,35,93,20);

// Cor e Posição do Painel
panel.setBounds(10,10,260,81);
panel.setBackground(Color.getHSBColor(HSB[0], HSB[1], HSB[2]));

// Cores e Posições dos Textos
Texto.setBounds(120,60,100,10);
Texto.setForeground(Color.yellow);
Texto1.setForeground(Color.white);
Texto1.setBounds(48,60,100,10);

// Adicionando os Componentes ao Painel e ao Container
container.add(panel);
panel.add(TBotao);
panel.add(TBotao1);
panel.add(TBotao2);
panel.add(Texto);
panel.add(Texto1);

TBotao.addActionListener( new ActionListener() {

   public void actionPerformed( ActionEvent event ) {
```

```
        // Verificação do Label do Botão para Definir o Conteúdo da Mensagem
        if (TBotao.getLabel() == "Conectar"){
            try {
                // Mensagem
                message = "Conectar-" + codSala;
                byte data[] = message.getBytes();

                // Armazendo na Variavel o IP do Servidor
                InetAddress EnderecoIP = InetAddress.getByName(ipServidor);

                // Criando Pacote para o Envio
                sendPacket = new DatagramPacket(data, data.length, EnderecoIP, 5000);

                // Enviando o Pacote
                socket.send( sendPacket );
            }
            // Problemas de Criação ou Envio de Pacotes
            catch ( IOException ioException ) {
                System.out.println(ioException.toString());
            }

        } else {
            if (TBotao.getLabel() == "Sair") {

                try {
                    // Mensagem
                    message = "Desconectar";
                    byte data[] = message.getBytes();

                    // Armazendo na Variavel o IP do Servidor
                    InetAddress EnderecoIP = InetAddress.getByName(ipServidor);

                    // Criando Pacote para o Envio
                    sendPacket = new DatagramPacket(data, data.length, EnderecoIP, 5000);

                    // Enviando o Pacote
                    socket.send( sendPacket );

                    System.exit(0);

                }
                // Problemas de Criação ou Envio de Pacotes
                catch ( IOException ioException ) {
                    System.out.println(ioException.toString());
                }
            }
        }
    }
} );

TBotao1.addActionListener( new ActionListener() {

    public void actionPerformed( ActionEvent event ) {

        // Verificação do Label do Botão para Definir o Conteúdo da Mensagem
        if (TBotao1.getLabel() == "Pedir Palavra") {

            message = "Pedir Palavra";
```

```
            try {
               byte data[] = message.getBytes();

               // Armazendo na Variavel o IP do Servidor
               InetAddress EnderecoIP = InetAddress.getByName(ipServidor);

               // Criando Pacote para o Envio
               sendPacket = new DatagramPacket(data, data.length, EnderecoIP, 5000);

               // Enviando o Pacote
               socket.send( sendPacket );

               TBotao1.setLabel("Aguardando...");
               TBotao1.setEnabled(false);

            }
            catch ( IOException ioException ) {
               System.out.println(ioException.toString());
            }
         }
      }
   } );


   TBotao2.addActionListener( new ActionListener() {

      public void actionPerformed( ActionEvent event ) {

         // Verificação do Label do Botão para Definir o Conteúdo da Mensagem
         if (TBotao2.getLabel() == "Falar") {

            TBotao2.setLabel("Terminar");
            TBotao2.setBounds(134,35,98,20);
            TBotao1.setLabel("Falando");
            captureAudio();

         }
         else {
            if (TBotao2.getLabel() == "Terminar") {

               stopCapture = true;

               TBotao2.setLabel("Falar");
               TBotao2.setVisible(false);
               TBotao1.setEnabled(true);
               TBotao1.setLabel("Pedir Palavra");

               message = "Terminou";
               chave = 1;

               try {
                  byte data[] = message.getBytes();

                  // Armazendo na Variavel o IP do Servidor
                  InetAddress EnderecoIP = InetAddress.getByName(ipServidor);

                  // Criando Pacote para o Envio
                  sendPacketP = new DatagramPacket(data, data.length, EnderecoIP, 6000);

                  // Enviando o Pacote
```

```
                    socket.send( sendPacketP );

                }
                catch ( IOException ioException ) {
                    System.out.println(ioException.toString());
                }
            }
        }
    } );

    // Define a posição da Janela
    setLocation( 400, 10 );

    // Define o tamanho da Janela
    setSize( 290, 130 );
    setVisible( true );

    // Criando DatagramSocket para Enviar e Receber Pacotes
    try {
        socket = new DatagramSocket();
    }
    // Problemas na Criação do DatagramSocket
    catch( SocketException socketException ) {
        socketException.printStackTrace();
        System.exit( 1 );
    }

} // Fim do Construtor do Cliente

// Esperando Pacotes enviados pelo Servidor
public void waitForPackets() {

    // Loop Infinito
    while ( true ) {

        // Recebendo Pacotes
        try {

            // Setando o Pacote
            byte data[] = new byte[ 100 ];
            receivePacket = new DatagramPacket( data, data.length );

            // Esperando por Pacotes
            socket.receive( receivePacket );

            // Convertendo a Mensagem do pacote em uma variavel String
            String msg = new String(receivePacket.getData(), 0, receivePacket.getLength());

            // Dividindo a String
            StringTokenizer st = new StringTokenizer(msg.toString(),"-");
            int ct = st.countTokens();

            if (ct > 1) {
                // Extraindo as partes da String
                msg = st.nextToken();
                port = st.nextToken();
                nome = st.nextToken();
                porta = Integer.parseInt(port);
            }
```

```
            // Verificação do Conteúdo da Mensagem recebida
            if (msg.equals("Conectar")) {

               // Criando o ServerSocket para Conexão TCP/IP
               try {
                  InetAddress EnderecoIP = InetAddress.getByName(ipServidor);
                  cliente = new Socket(EnderecoIP, 5500);
               }

               // Problemas durante a Criação do ServerSocket
               catch (IOException e) {
                  e.printStackTrace();
               }

               try {
                  // Instânciando a classe RecebeThread
                  Thread recebeThread = new Thread(new RecebeThread());
                  // Inicia a Thread
                  recebeThread.start();
               }
               catch (Exception e) {
                  System.out.println(e);
               }// Fim do catch

               // Altera as Informações da Janela
               TBotao.setLabel("Sair");
               Texto.setText("Conectado");
               TBotao1.setEnabled(true);
               Texto.setForeground(Color.green);

               // Desabilita o Botão Fechar da Janela
               setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);

               // Dividindo o Endereço IP para retirar a barra da frente
               StringTokenizer st1 = new StringTokenizer(receivePacket.getAddress().toString(),"/");
               int ct1 = st1.countTokens();

               String end = st1.nextToken();
            }
            else {
               if (msg.equals("Permissao")) {
                  TBotao1.setText("Pode Falar");
                  TBotao2.setVisible(true);

                  // Instânciando a classe ColoredJOptionPane
                  ColoredJOptionPane warning = new ColoredJOptionPane();
                  warning.showMessageDialog(null, "Chegou a sua vez de Falar!", "WORLDCONF
INFORMA!", JOptionPane.WARNING_MESSAGE);

                  // Mostrando a Janela
                  warning.show();
               }
            }
         }
         // Processar Problemas do Recebimento das Mensagesn
         catch( IOException exception ) {
            System.out.println( exception.toString() + "\n" );
            exception.printStackTrace();
         }
```

```
    }  // Fim do while

} // Fim do método waitForPackets

// Método que captura o som do microfone e armazena no objeto ByteArrayOutputStream
private void captureAudio(){

    try{
        // Setando as variaveis para a captura do som
        audioFormat = getAudioFormat();
        DataLine.Info dataLineInfo = new DataLine.Info(TargetDataLine.class,audioFormat);
        targetDataLine = (TargetDataLine)AudioSystem.getLine(dataLineInfo);
        targetDataLine.open(audioFormat);
        targetDataLine.start();

        // Criando a thread para capturar dados do microfone
        Thread captureThread = new Thread(new CaptureThread());
        // Iniciando a Thread
        captureThread.start();

    }
    catch (Exception e) {
        System.out.println(e);
        System.exit(0);
    }// Fim do catch

} // Fim do método captureAudio

// Método que transmite o Objeto Serializado para o Servidor
private void transmitAudio(){

    try {
        // Criando a thread para enviar os dados capturados
        Thread transmitThread = new Thread(new TransmitThread());
        // Iniciando a Thread
        transmitThread.start();
    }
    catch (Exception e) {
        System.out.println(e);
        System.exit(0);
    }// Fim do catch

} // Fim do método transmitAudio

// Método que reproduz o áudio recebido por TCP dentro de um objeto
private void playAudio() {

    try {

        arrayStreamTeste = (ArrayStream)listaObjetos.removeFirst();
        System.out.println(arrayStreamTeste);

        // Armazenando o som recebido através da classe ArrayStream em um array de bytes
        byte audioData[] = arrayStreamTeste.getAudio();

        // Setando um Input Stream para o array de bytes que contém os dados
        InputStream byteArrayInputStream = new ByteArrayInputStream(audioData);
        AudioFormat audioFormat = getAudioFormat();
```

```
            audioInputStream = new AudioInputStream(
byteArrayInputStream,audioFormat,audioData.length/audioFormat.getFrameSize());
            DataLine.Info dataLineInfo = new DataLine.Info( SourceDataLine.class, audioFormat);
            sourceDataLine = (SourceDataLine) AudioSystem.getLine(dataLineInfo);
            sourceDataLine.open(audioFormat);
            sourceDataLine.start();

            // Criando a thread para Reproduzir o som recebido
            Thread playThread = new Thread(new PlayThread());
            // Iniciando a Thread
            playThread.start();

        }
        catch (Exception e) {
            System.out.println(e);
            System.exit(0);
        }// Fim do catch

    } // Fim do método playAudio

    // Define as especificações de qualidade do áudio
    private AudioFormat getAudioFormat(){

        float sampleRate = 8000.0F;
        //8000,11025,16000,22050,44100
        int sampleSizeInBits = 16;
        //8,16
        int channels = 1;
        //1,2
        boolean signed = true;
        //true,false
        boolean bigEndian = false;
        //true,false
        return new AudioFormat(sampleRate, sampleSizeInBits, channels, signed, bigEndian);

    } // Fim do getAudioFormat

    // Classe interna que captura som do microfone
    class CaptureThread implements Runnable {

        // Tamanho do buffer temporário
        byte tempBuffer[] = new byte[1000];

        public void run() {

            // Objeto que será armazenado os dados capturados
            byteArrayOutputStream = new ByteArrayOutputStream();
            stopCapture = false;

            try {

                try {
                    Thread transmitThread = new Thread(new TransmitThread());
                    transmitThread.start();
                }
                catch (Exception e) {
                    System.out.println(e);
                }// Fim do catch
```

```
            // Loop até que o botão Terminar seja pressionado mudando o valor da variavel
stopCapture para true
            while (!stopCapture) {

               // Lê dados do buffer interno por linhas
               int cnt = targetDataLine.read( tempBuffer, 0, tempBuffer.length);

               if (cnt > 0) {

                  // Salva os dados em Objeto Output Stream
                  byteArrayOutputStream.write( tempBuffer, 0, cnt );

               }// Fim do if

            }// Fim do while

            // Fecha o Objeto de Armazenamento
            byteArrayOutputStream.close();

         }
         catch (Exception e) {
            System.out.println(e);
         }// Fim do catch

      }// Fim do run

   } // Fim da classe interna CaptureThread

   // Classe interna para transmitir o som capturado para o servidor
   class TransmitThread extends Thread {

      public void run() {

         int c = 0;

         // A thread fica alguns segundos esperando para continuar a execução
         try {
            TransmitThread.sleep(750);
         }
         catch (Exception e) {
            System.out.println(e.getMessage());
         }

         // Loop para o envio dos Objetos serializados ao servidor
         while (!stopCapture) {

            // Nova paralização para evitar processamento excedente
            try {
               TransmitThread.sleep(125);
            }
            catch (Exception e) {
               System.out.println(e.getMessage());
            }

            //System.out.println("t: "+byteArrayOutputStream.);

            // Verifica o tamanho do buffer do objeto para ser transmitido a cada 20000
            if ( byteArrayOutputStream.size() >= (160000 + c)) {

               // Armazenando o conteúdo do Objeto em um array de bytes
```

```
            byte [] teste = byteArrayOutputStream.toByteArray();
            int tam = teste.length;
            byte [] teste1 = new byte[tam - c];

            // Armazena apenas a parte que ainda não foi enviada
            for (int i=c; i < tam; i++) {
               teste1[i - c] = teste[i];
            }

            // Atualizando a variavel do tamanho já enviado
            c = (tam - 100);
            System.out.println(c);

            // Instância do objeto ArrayStream
            arrayStreamObject = new ArrayStream();
            // Setando o array de bytes, a ser enviado, dentro do objeto serializado
            arrayStreamObject.setAudio(teste1);

            try {

               System.out.println("Enviado:" + arrayStreamObject);

               // Variaveis para Envio e Recebimento de Informações
               enviar = new ObjectOutputStream(cliente.getOutputStream());
               enviar.writeObject(arrayStreamObject);
               enviar.flush();

               System.out.println("Enviou");

            }
            catch (IOException e) {
               System.out.println("Erro:" + e.getMessage());
            }// Fim do catch

         }// Fim do if
      }// Fim do while

      // Verificação para garantir que o final do arquivo seja enviado
      if (chave == 1) {

         // Armazenando o conteúdo do Objeto em um array de bytes
         byte [] teste = byteArrayOutputStream.toByteArray();
         int tam = teste.length;
         byte [] teste1 = new byte[tam - c];

         // Armazena apenas a parte que ainda não foi enviada
         for (int i=c; i < tam; i++) {
            teste1[i - c] = teste[i];
         }

         // Atualizando a variavel do tamanho já enviado
         c = tam;

         // Instância do objeto ArrayStream
         arrayStreamObject = new ArrayStream();
         // Setando o array de bytes, a ser enviado, dentro do objeto serializado
         arrayStreamObject.setAudio(teste1);

         try {
```

```
            System.out.println("Enviado:" + arrayStreamObject);

            // Variaveis para Envio e Recebimento de Informações
            enviar = new ObjectOutputStream(cliente.getOutputStream());
            enviar.writeObject(arrayStreamObject);
            enviar.flush();

            System.out.println("Enviou");

        }
        catch (IOException e) {
            System.out.println("Erro:" + e.getMessage());
        }// Fim do catch

        chave = 0;
    }

}// Fim do run

} // Fim da classe interna TransmitThread

// Classe interna para reproduzir os dados que chegaram através da rede
class PlayThread implements Runnable {

    byte tempBuffer[] = new byte[10000];

    public void run(){

        try {
            int cnt;

            System.out.println("Reproduziu");

            // Mantém o loop até que o método de leitura do read retornar -1
            while((cnt = audioInputStream.read(tempBuffer, 0, (tempBuffer.length - 100))) != -1) {

                if(cnt > 0) {
                    // Escreve dados para um buffer interno, ele será enviado para o saída do som
                    sourceDataLine.write(tempBuffer, 0, cnt);
                }// Fim do if

            }// Fim do while

            // Bloqueia e aguarda por um buffer interno da linha de dados para esvaziar
            sourceDataLine.drain();
            sourceDataLine.close();

            if (listaObjetos.size() > 0) {
                playAudio();
            }
            else {
                play = false;
            }

        }
        catch (Exception e) {
            System.out.println(e);
            System.exit(0);
        }// Fim do catch
```

```
    }// Fim do run

  } // Fim da classe interna PlayThread

  // Classe interna que recebe o objeto da rede
  class RecebeThread implements Runnable {

     public void run() {

        try {

           while (true) {

              receber = new ObjectInputStream(cliente.getInputStream());
              arrayStreamRecebido = (ArrayStream)receber.readObject();
              System.out.println("Recebido:" + arrayStreamRecebido);

              listaObjetos.addLast(arrayStreamRecebido);

              if (!play) {
                 playAudio();
                 play = true;
              }

           }

        }
        catch (Exception e) {
           System.out.println("Erro recebimento:" + e.getMessage());
        }

     }

  } // Fim da classe interna RecebeThread

  // Classe que personaliza o componente JOptionPane
  class ColoredJOptionPane extends JOptionPane {

     public ColoredJOptionPane() {

        // Definindo as cores do JOptionPane
        float [] HSB = Color.RGBtoHSB(51, 139, 190, (new float[3]));
        float [] HSB1 = Color.RGBtoHSB(247, 199, 91, (new float[3]));

        // Setando as configurações da JOptionPane
        UIManager.put("OptionPane.background",Color.getHSBColor(HSB[0],HSB[1],HSB[2]));
        UIManager.put("OptionPane.messageForeground", Color.white);
        UIManager.put("Panel.background",Color.getHSBColor(HSB[0],HSB[1],HSB[2]));
        UIManager.put("Button.background",Color.getHSBColor(HSB1[0], HSB1[1], HSB1[2]));
        UIManager.put("Button.foreground",Color.DARK_GRAY);

     } // Fim do método ColoredJOptionPane

  } // Fim da classe ColoredJOptionPane

  // Executa a Aplicação
  public static void main(String args[]) {

     Client application = new Client();
     application.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
```

```
        application.setResizable(false);
        application.waitForPackets();

    } // Fim do main da aplicação

}  // Fim da Classe Client
```

17.3. Server.java

```
import java.io.*;
import java.net.*;
import java.awt.*;
import java.util.StringTokenizer;
import java.awt.event.*;
import javax.swing.*;
import javax.sound.sampled.*;
import java.util.LinkedList;

public class Server extends JFrame {

    private JTextArea displayArea;
    private JButton botao;
    private JLabel label;
    private Panel panel;
    private Font fonte;
    private JScrollPane scroll;

    private DatagramPacket sendPacket, receivePacket, sendPacketP, receivePacketP;
    private DatagramSocket socket, socketP, socketS;
    private InetAddress [] Ips = new InetAddress[100];
    private int [] portaUsuario = new int[100];
    private int []  numResp;
    private int porta, codSala;
    private String comp, msg, nome, tela, codigo;

    private ObjectOutputStream enviar;
    private ObjectInputStream [] receber = new ObjectInputStream[100];
    private ServerSocket servidor;
    private Socket conexao;
    private Socket [] cliente = new Socket[100];

    private ByteArrayOutputStream byteArrayOutputStream;
    private ArrayStream [] arrayStreamObject = new ArrayStream[100];

    private LinkedList [] listaPermissao = new LinkedList[100];
    int contador = 0, contadorUsuario = 0, cg = 0, chaveSocketRecebe = 0;
    int [] chave = new int[100];
    int [] Sala = new int[100];
    String ipPermissao = "";

    // Criando Interface e o DatagramSocket
    public Server() {

        // Interface do Servidor
        super( "Servidor Conferência Áudio" );

        // Definição das Cores
        float [] HSB = Color.RGBtoHSB(51, 139, 190, (new float[3]));
        float [] HSB1 = Color.RGBtoHSB(247, 199, 91, (new float[3]));
        float [] HSB2 = Color.RGBtoHSB(55, 74, 102, (new float[3]));
        float [] HSB4 = Color.RGBtoHSB(217, 215, 212, (new float[3]));

        // Instância e Configura o container
        Container container = getContentPane();
        container.setLayout(null);
        container.setBackground(Color.getHSBColor(HSB4[0], HSB4[1], HSB4[2]));
        container.setForeground(Color.white);
```

```
// Instância e Configura o Painel
panel = new Panel();
panel.setLayout(null);
panel.setBounds(10,10,300,250);
panel.setBackground(Color.getHSBColor(HSB[0], HSB[1], HSB[2]));

// Instância e Configura a Fonte
fonte = new Font("Verdana",Font.CENTER_BASELINE,12);

// Instância e Configura o TextArea
displayArea = new JTextArea();

// Instância e Configura o ScrollPane
scroll = new JScrollPane(displayArea);
scroll.setBounds(10,59,281,181);

// Instância e Configuraç o Botão
botao = new JButton("Lista Usuários");
botao.setBounds(85,10,130,20);
botao.setBackground(Color.getHSBColor(HSB1[0], HSB1[1], HSB1[2]));
botao.setForeground(Color.getHSBColor(HSB2[0], HSB2[1], HSB2[2]));
botao.setFont(fonte);

// Instância e Configura o Texto
label = new JLabel("Informações");
label.setFont(fonte);
label.setForeground(Color.white);
label.setBounds(107,40,100,10);

// Atribuições dos componentes na tela
container.add(panel);
panel.add(botao);
panel.add(label);
panel.add(scroll);

botao.addActionListener( new ActionListener() {

   public void actionPerformed( ActionEvent event ) {

      int c = 1;
      for (int cont = 0; cont < Ips.length; cont++) {
         if (Ips[cont] != null){
            if (c == 1) {
               displayArea.append("\n----------------------------------------------------------" +
                                  "\nLista de IPs Conectados:\n" +
                                  "----------------------------------------------------------\n");
               c = 0;
            }

            // Dividindo o Endereço IP para retirar a barra da frente
            StringTokenizer st1 = new StringTokenizer(Ips[cont].toString(),"/");
            int ct1 = st1.countTokens();

            String end = st1.nextToken();

            displayArea.append( end +"\n" );
         }
      }
      if (c == 0) {
```

```
            displayArea.append("\n");
         }
         if (c == 1) {
            displayArea.append("\n-----------------------------------------------------------" +
                  "\nNenhum Usuário Conectado\n" +
                  "-----------------------------------------------------------\n");
         }
      }
   } );

   setSize( 330, 300 );
   setVisible( true );

   // Criando o ServerSocket para Conexão TCP/IP
   try {
      servidor = new ServerSocket(5500);
   }

   // Problemas durante a Criação do ServerSocket
   catch (IOException e) {
      e.printStackTrace();
   }

   // Criando o DatagramSocket para Enviar e Receber Pacotes
   try {
      socket = new DatagramSocket(5000);
      socketP = new DatagramSocket(6000);
      socketS = new DatagramSocket(7000);
   }

   // Problemas durante a Criação do DatagramSocket
   catch( SocketException socketException ) {
      socketException.printStackTrace();
      System.exit( 1 );
   }

}  // Fim Construtor do Server

// Esperando por Pacotes enviados
public void waitForPackets() {

   int x = 0;

   // Declaração do Array de Respostas
   int [] numResp = new int[100];

   // Declaração das Listas de Permissões e Variavel de flag com índice zero
   for (int i=0; i < 100; i++) {
      chave[i] = 0;
      listaPermissao[i] = new LinkedList();
   }

   String tela = "";

   Thread PermissaoThread = new Thread(new PermissaoThread());
   PermissaoThread.start();

   // Loop Sem Fim
   while ( true ) {
```

```
// Recebe o pacote e apresenta o conteúdo
try {

   // Definindo o Pacote a ser Recebido
   byte data[] = new byte[ 100 ];
   receivePacket = new DatagramPacket(data, data.length);

   // Esperando pelos Pacotes
   socket.receive( receivePacket );

   // Armazenando o IP do Remetente do Pacote
   InetAddress IP = receivePacket.getAddress();

   // Armazenando em String o conteúdo da mensagem do pacote
   msg = new String(receivePacket.getData(), 0, receivePacket.getLength());

   // Dividindo a String
   StringTokenizer st = new StringTokenizer(msg.toString(),"-");
   int ct = st.countTokens();

   if (ct > 1) {
      // Extraindo as partes da String
      msg = st.nextToken();
      codigo = st.nextToken();
      codSala = Integer.parseInt(codigo);
      System.out.println("Código da Sala:" + codSala);
   }

   // Verifica o conteúdo da mensagem
   if (msg.equals("Conectar")) {

      // Armazena o IP do usuário em um Array
      Ips[x] = receivePacket.getAddress();

      // Armazena a Porta do Usuario UDP
      portaUsuario[x] = receivePacket.getPort();

      // Armazena a Porta do Usuario UDP
      Sala[x] = codSala;

      // Compõe o conteúdo da mensagem a ser enviada para o usuário
      comp = msg + "-" + Integer.toString(receivePacket.getPort()) + "-user" + x;

      // Variavel apresentada na tela do servidor
      tela = " conectou\n";

      // Envia o Pacote de Resposta para o Usuário
      sendPacketToClient();

      // Aguarda por conexões TCP do Usuário
      cliente[x] = servidor.accept();

      System.out.println(cliente[x].getRemoteSocketAddress().toString() + " Sala:" + Sala[x]);

      // Incremento das Variaveis
      x++;
      contadorUsuario++;

      // Mostra o Pacote na Tela do Servidor
      displayPacket(tela);
```

```
}
else {
    if (msg.equals("Desconectar")) {

        // Loop para encontrar o IP do Usuário que desconectou
        for (int cont = 0; cont < Ips.length; cont++){
            if (Ips[cont] != null){
                if (Ips[cont].equals(IP)){
                    int t = listaPermissao[Sala[cont]].size();
                    System.out.println(t);
                    if (t > 0) {
                        for (int i=0; i < t; i++) {
                            System.out.println("teste");
                            System.out.println(listaPermissao[Sala[cont]].get(i).toString());
                            if (listaPermissao[Sala[cont]].get(i).toString().equals(IP.toString())) {
                                listaPermissao[Sala[cont]].remove(i);
                                System.out.println("Aqui");
                            }
                        }
                    }

                    // Apagando o IP e a Porta do Usuário dos Arrays
                    Ips[cont] = null;
                    numResp[cont] = 0;
                    Sala[cont] = 0;
                    cliente[cont].close();
                }
            }
        }

        // Variavel apresentada na tela do servidor
        tela = " desconectou\n";

        // Mostra o Pacote na Tela do Servidor
        displayPacket(tela);

    }
    else {
        if (msg.equals("Pedir Palavra")) {

            String listaIP = receivePacket.getAddress().toString();

            for (int d=0; d < Ips.length; d++) {
                if ((Ips[d] != null) && (Ips[d].toString().equals(listaIP))) {
                    codSala = Sala[d];
                }
            }

            System.out.println(codSala);
            listaPermissao[codSala].addLast(listaIP);

            if ((listaPermissao[codSala].size() == 1) && (chave[codSala] == 0)) {
                comp = "Permissao";
                ipPermissao = listaPermissao[codSala].removeFirst().toString();

                for (int t=0; t < Ips.length; t++) {
                    if ( (Ips[t] != null) && (Ips[t].toString().equals(ipPermissao))) {
                        cg = 0;
                        Thread SocketRecebe = new SocketRecebe(t, Sala[t]);
```

```
                        SocketRecebe.start();
                    }
                }
                sendPacketToClient();
                chave[codSala] = 1;
            }
        }
    }
}

// Problemas durante a Manipulação dos Pacotes
catch( IOException ioException ) {
    displayArea.append( ioException.toString() + "\n" );
    ioException.printStackTrace();
}

} // Fim do while

} // Fim do Método waitForPackets

// Mostra o conteúdo dos Pacotes
private void displayPacket(String texto) {

    // Retira a barra da frente do endereço IP
    StringTokenizer st = new StringTokenizer(receivePacket.getAddress().toString(),"/");
    int ct = st.countTokens();

    String end = st.nextToken();

    // Mostra na tela do servidor informações do pacote
    displayArea.append( "Endereço IP " + end + texto );

} // Fim do Método displayPacket

// Envia o Pacote para o Usuário
private void sendPacketToClient() throws IOException {

    // Verifica o conteúdo da mensagem do pacote
    if ((msg.equals("Conectar")) || (comp.equals("Permissao"))) {

        // Envia a variavel comp, que possui a mensagem, porta e o nome do seu usuário no servidor
        byte dataP[] = comp.getBytes();
        sendPacket = new DatagramPacket( dataP, dataP.length, receivePacket.getAddress(),
receivePacket.getPort() );
        comp = "";
    }
    else {
        sendPacket = new DatagramPacket( receivePacket.getData(), receivePacket.getLength(),
receivePacket.getAddress(), receivePacket.getPort() );
    }

    // Envia Pacote
    socket.send( sendPacket );

} // Fim do Método sendPacketToClient

// Classe que gerencia as permissões de fala
class PermissaoThread extends Thread {
```

```
public void run() {

   while (true) {

      try {

         int portaEnvio = 0;
         int aux = 0;

         // Definindo o Pacote a ser Recebido
         byte data[] = new byte[ 100 ];
         receivePacketP = new DatagramPacket(data, data.length);

         // Esperando pelos Pacotes
         socketP.receive(receivePacketP);

         // Armazenando em String o conteúdo da mensagem do pacote
         msg = new String(receivePacketP.getData(), 0, receivePacketP.getLength());

         for (int d=0; d < Ips.length; d++) {
            if ((Ips[d] != null) && (Ips[d].equals(receivePacketP.getAddress().toString()))) {
               codSala = Sala[d];
            }
         }

         cg = 1;

         if ( msg.equals("Terminou") && (listaPermissao[codSala].size()>0) ) {

            System.out.println("Proximo");
            ipPermissao = listaPermissao[codSala].removeFirst().toString();

            StringTokenizer st = new StringTokenizer(ipPermissao,"/");
            int ct = st.countTokens();

            String end = st.nextToken();

            InetAddress ipTemp = InetAddress.getByName(end);

            for (int i=0; i < Ips.length; i++){
               if ((Ips[i] != null) && (Ips[i].toString().equals(ipPermissao))) {
                  portaEnvio = portaUsuario[i];
                  aux = i;
               }
            }

            String mensagem = "Permissao";
            byte dataP[] = mensagem.getBytes();
            sendPacketP = new DatagramPacket(dataP, dataP.length, ipTemp, portaEnvio);

            // Envia Pacote
            socketP.send( sendPacketP );

            Thread SocketRecebe = new SocketRecebe(aux, Sala[aux]);
            SocketRecebe.start();

         }
         else {
            chave[codSala] = 0;
         }
```

```
            }
            catch( IOException ioException ) {
                System.out.println("teste" + ioException.toString());
            }
        }
    }
} // Fim da Classe PermissaoThread

// Classe para receber objetos pela conexão TCP com o usuário
class SocketRecebe extends Thread {

    public int con, sala;
    public boolean ch, ch1;
    public ObjectInputStream recebe;

    public SocketRecebe (int c, int cod) {
        con = c;
        sala = cod;
    }

    public void run() {

        ch = true;
        ch1 = false;

        try {
            while (ch) {

                System.out.println("Aguardando..." + con);
                receber[con] = new ObjectInputStream (cliente[con].getInputStream());
                arrayStreamObject[sala] = (ArrayStream)receber[con].readObject();
                System.out.println("Stream Recebido:" + arrayStreamObject[sala].getAudio().toString());

                System.out.println("Thread Envia...");
                Thread SocketEnvia = new SocketEnvia(con, arrayStreamObject[sala], sala);
                SocketEnvia.start();
                System.out.println("Inicio a Thread...");

                if (cg == 1) {
                    System.out.println("Terminou");
                    ch = false;
                }

            }
            cg = 0;
        }
        catch (Exception erro) {
            if (erro.toString() == "java.io.EOFException") {
                try {
                    //receber[con].close();
                }
                catch (Exception e) {
                    System.out.println("Erro1:" + e.getMessage());
                }
            }
            else {
                System.out.println("Erro:" + erro);
            }
        }
    }
```

```
} // Fim da Classe SocketRecebe

// Classe para enviar objeto pela conexão TCP com o usuário
class SocketEnvia extends Thread {

    public int con, sala;
    public ObjectOutputStream envia;
    public ArrayStream arrayStreamEnvio;
    int chaveCont = 0;

    public SocketEnvia (int c, ArrayStream as, int s) {
        con = c;
        arrayStreamEnvio = as;
        sala = s;
    }

    public void run() {

        try {

            for (int a=0; a < Ips.length; a++) {
                if ((con != a) && (Ips[a] != null) && (sala == Sala[a])) {
                    Thread SocketEnviaUsuario = new SocketEnviaUsuario(a, arrayStreamEnvio);
                    SocketEnviaUsuario.start();
                }
            }

        }
        catch (Exception erro) {
            System.out.println("Erro de Envio 1:" + erro);
        }
    }
} // Fim da Classe SocketEnvia

// Classe para enviar objeto pela conexão TCP com o usuário
class SocketEnviaUsuario extends Thread {

    public int con;
    public ObjectOutputStream envia;
    public ArrayStream arrayStreamEnvio;

    public SocketEnviaUsuario (int c, ArrayStream as) {
        con = c;
        arrayStreamEnvio = as;
    }

    public void run() {

        try {

            envia = new ObjectOutputStream (cliente[con].getOutputStream());
            envia.writeObject(arrayStreamEnvio);
            envia.flush();
            System.out.println("Enviou "+cliente[con].getRemoteSocketAddress().toString());

        }
        catch (Exception erro) {
            if (erro.toString() == "java.io.EOFException") {
                try {
                    envia.close();
```

```
                }
                catch (Exception e) {
                    System.out.println("Erro2: " + e.getMessage());
                }
            }
            else {
                System.out.println(erro);
            }
        }
    }
  } // Fim da Classe SocketEnviaUsuario

  // Executa a Aplicação
  public static void main(String args[]) {

      Server application = new Server();
      application.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      application.waitForPackets();

  } // Fim da Execução da Aplicação

}  // Fim da classe Server
```