Universidade de Aveiro
2009
Departamento de Electrónica, Telecomunicações e Informática

Américo José de Oliveira Campos

Tecnologias GRID: serviços de apoio a Bibliotecas Digitais

Universidade de Aveiro
2009
Departamento de Electrónica, Telecomunicações e Informática

Américo José de Oliveira Campos

Tecnologias GRID: serviços de apoio a Bibliotecas Digitais

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob a orientação científica do Dr. Joaquim Arnaldo Martins, Professor Catedrático do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

Para a minha família, namorada e a todos os que me ajudaram a chegar até aqui…

o júri

Presidente	Doutor Armando José Formoso de Pinho Professor Associado com Agregação da Universidade de Aveiro
Vogal – Arguente Principal	Doutor Fernando Joaquim Lopes Moreira Departamento Inovação, Ciência e Tecnologia da Universidade Portucalense.
Vogal – Orientador	Doutor Joaquim Arnaldo Carvalho Martins Professor Catedrático da Universidade de Aveiro

agradecimentos

Agradeço aos meus pais pelo esforço que fizeram, aos meus irmãos pela paciência, e à minha namorada Helena por toda a dedicação, tolerância e compreensão.

Agradeço toda a colaboração obtida por parte dos meus colegas e amigos.

Para finalizar agradeço também ao meu orientador o Dr. Joaquim Arnaldo Martins, assim como, aos coordenadores Marco Fernandes e Pedro Almeida pelo apoio e compreensão na realização deste projecto.

palavras-chave Computação Grid, Bibliotecas Digitais,Alchemi

resumo

Esta dissertação tem como ponto de partida o estudo da tecnologia Grid e sua aplicação no âmbito das Bibliotecas Digitais. A razão para o uso da tecnologia Grid deve-se ao aumento do interesse e da divulgação desta, em grande parte por causa da interligação entre sistemas computacionais ser cada vez mais rápida e eficiente. Tornando-se mais vantajoso o uso de recursos distribuídos em algumas aplicações.

Este projecto visa desenvolver um conjunto de aplicações *Grid* a ser usada nas Bibliotecas Digitais. Para este ser possível foi efectuado um estudo da tecnologia Grid, assim como, alguns dos sistemas de middleware Grid de modo a escolher o sistema que mais se adequava aos requisitos do SInBAD.

keywords Grid computing, Digital Libraries, Alchemi

abstract

Due to the increase of interest and disclosure of Grid technology, coupled with the fact that it is ever developed, largely because of the interconnection between computer systems become faster and more efficient. It became the most advantageous use of resources distributed in some applications.

This project aims at developing a Grid application to be used in Digital Libraries. For this possible was being done a study of Grid technology, as well as some of the Grid middleware systems to choose the system that are best suited to the requirements of Sinbad.

Índice

Índice de Figuras

Índice de Tabelas

1 Introdução

1.1 Projecto

O SInBAD, Sistema Integrado para Bibliotecas e Arquivos Digitais (sinbad.ua.pt), é um projecto da Universidade de Aveiro, tendo este como finalidade a construção de um sistema de informação com capacidade para armazenar os diferentes tipos de documentos produzidos pela universidade, tais como teses, dissertações, fotografias, vídeos, etc.

O SInBAD suporta vários tipos de ficheiros, mas tem primazia por documentos em formato PDF e imagens em TIFF. Para realizar o armazenamento de textos e imagens em formato PDF é necessário converter para diferentes formatos e resoluções quer fotografias quer vídeos.

1.2 Motivação

Embora o conceito de tecnologia *Grid* já exista há algum tempo, só nos últimos tempos se tem dada a devida importância a esta tecnologia, como tal a investigação e o desenvolvimento nesta área têm tido um progresso enorme, suscitando o interesse de grandes corporações, como a IBM, a Sun entre outras.

Assim, e no âmbito do projecto SInBAD, pretende-se estudar a aplicação desta tecnologia, através do desenvolvimento de quatro serviços *Web.* A intenção de estudar esta tecnologia para posterior aplicação, deve-se ao facto de embora o projecto SInBAD usar preferencialmente o formato PDF e TIFF, muitas vezes ser necessário proceder a outros tipos de conversão. Assim pretende-se desenvolver um conjunto de serviços de apoio aos já existentes.

Este conjunto de serviços irá ser constituído por quatro serviços, sendo que dois deles serão para converter documentos em formato PDF para outro formato qualquer, e outro serviço de conversão de imagens.

Outro serviço desenvolvido será para retirar as margens brancas, dos documentos para desta forma os tornar mais pequenos, e mais um serviço para reconhecimento dos dados utilizando o reconhecimento óptico de caracteres.

Um factor comum a todos estes serviços é o tempo que demora a processar grandes documentos. A tecnologia *Grid*, pretende minimizar o tempo dispendido no processamento do documento, uma vez que várias máquinas estão a trabalhar conjuntamente para o mesmo fim.

1.3 Estrutura da Dissertação

Esta Dissertação está dividida em seis capítulos:

No primeiro capítulo efectua-se uma descrição resumida do trabalho a desenvolver, assim como, da tecnologia utilizada.

No segundo capítulo deste documento descreve-se a tecnologia *Grid* (conceitos, componentes, etc.) para compreender a sua filosofia e modo de funcionamento.

No terceiro capítulo é realizada a descrição das diferentes *middleware* estudadas, assim como, explicação da escolha do Alchemi.

No quarto capítulo efectua-se a descrição do projecto proposto.

No quinto capítulo apresenta-se os resultados das aplicações desenvolvidas.

No sexto capítulo temos uma discussão acerca do que era esperado com a realização deste projecto e da experiência obtida com o mesmo, assim como, referência a possíveis caminhos a tomar para a continuação do projecto e para o seu aperfeiçoamento.

2 Computação *Grid*

2.1 Computação Distribuída

Uma vez que a computação *Grid* trata um tipo de computação distribuída começa-se por explicar o que é a computação distribuída. Segundo a definição de Andrew Tanenbaum é uma "colecção de computadores independentes que se apresenta ao utilizador como um sistema único e consistente" [32], ou seja, computação distribuída refere-se aos meios pelos quais um único programa de computador é executado em mais de um computador ao mesmo tempo. Em particular, os diferentes elementos e objectos de um programa que estão a ser executados ou processados usando diferentes processadores.

2.2 O Início do *Grid*

Na última década houve muitas mudanças no modo de utilização dos recursos e serviços computacionais. Antigamente um computador, de modo geral, apenas necessitava de um sistema computacional e uma infra-estrutura localizada. No entanto, esta situação já não se verifica devido, entre outros factores, à interligação entre sistemas computacionais ser cada vez mais rápida e eficiente. Assim, tornou-se mais vantajoso o uso de recursos distribuídos em algumas aplicações.

Conseguimos obter um sistema distribuído através da ligação dos sistemas computacionais e da sua respectiva comunicação. Vários grupos de investigação estão a projectar, desenvolver e a conceber sistemas distribuídos de computação. Estes grupos têm desenvolvido *middleware*, bibliotecas e ferramentas que permitem a cooperação entre sistemas geograficamente distribuídos que funcionam como uma única plataforma para execução de aplicações. Este tipo de abordagem é conhecida por diferentes tipos de designações, tais como, metacomputação, computação escalável entre outras e mais recentemente como computação em grelha (*Grid*).

Primeira geração

O início da *Grid* começou em projectos para interligação de super computadores, que na altura se denominava por metacomputação. Usualmente o objectivo desses projectos era providenciar recursos computacionais para aplicações de alto desempenho.

Nesta área houve dois projectos que se destacaram, o FAFNER [12] e o I-WAY [20]. Estes são dois projectos diferentes mas com objectivos comuns, ambos ambicionavam a comunicação, a gestão de recursos e a manipulação de informação remota para obter um trabalho mais eficiente.

FAFNER

O algoritmo de encriptação de chave pública RSA é baseado na premissa de que é muito difícil factorizar um número muito grande e muito complexo computacionalmente, então foram desenvolvidos algoritmos de factorização paralelos que pudessem ser distribuídos. Surgiu em 1995, num consórcio liderado pela Bellcor, um projecto de factorização via *Web* designado por FAFNER (*Factoring via Network Enabled Recursion).*

I-WAY

O projecto I-WAY (*Information Wide Area Year*) foi uma experiência de uma rede de alto desempenho para interligar alguns super computadores, nos Estados Unidos, num avançado ambiente visual. O I-WAY foi concebido em 1995 e pretendia integrar uma rede com grande largura de banda já existente em vez de criar uma de raiz.

Ambos os projectos foram importantes na criação de ambientes de metacomputação através da integração de recursos.

Segunda Geração

Hoje em dia a *Grid* já não é, apenas, constituída por alguns super computadores específicos. A grande largura de banda existente e adopção de normas permitiram que a *Grid* fosse vista como um sistema distribuído viável a nível global, que pode suportar diversos requisitos de diversas aplicações em larga escala.

A visão da *Grid* foi apresentada por Ian Foster no livro "*The Grid – Blueprint for New Computing Infrastructure*"[18]. Existiam três grandes questões principais que tinham de ser confrontadas:

1. Heterogeneidade: uma vez que a *Grid* é composta por uma grande multiplicidade de recursos heterogéneos e domínios administrativos;
2. Escalabilidade: uma vez que pode ser composta apenas por alguns recursos ou por muitos;
3. Adaptabilidade: uma vez que a possibilidade de falhas é grande e o sistema deve adaptar-se de forma dinâmica aos recursos que ainda se encontrem disponíveis.

Assim, de forma a resolver este problema, usa-se uma camada entre os recursos e as aplicações designada de *middleware*, que cria um ambiente homogéneo para a camada de aplicação e fornece um conjunto de interfaces normalizadas para os diferentes serviços. Este assunto será desenvolvido mais à frente na secção 2.6.

Na segunda geração promove-se a interoperabilidade que é essencial para a computação em larga escala.

Terceira Geração

Nesta fase optou-se por reutilizar componentes e recursos já existentes. Pode-se então afirmar que a terceira geração está mais direccionada para abordar o modelo de serviços orientados. Estes serviços necessitam de dispor de informação acerca das funcionalidades, disponibilidade e interfaces dos diferentes componentes e essa informação tem de estar de acordo com o que pode ser processado pela máquina.

Surge nesta geração a noção de automação, isto porque deixou de ser humanamente possível lidar com escala e heterogeneidade, assim passou a ser necessário que, por exemplo, os sistemas fossem capazes de, até certo ponto, se tornarem independentes.

2.3 Conceitos

Para computação em *Grid* existem diferentes definições. Esta secção propõe-se a apresentar alguns destes conceitos, benefícios e componentes necessários para esta computação.

Um dos conceitos de computação *Grid* é a partilha, de forma dinâmica, de recursos computacionais em que os utilizadores podem estar separados geograficamente e formada por uma rede heterogénea de recursos partilhados por múltiplas organizações [25]. Para Berstis [5] a definição de computação *Grid* seria uma evolução da computação distribuída. A finalidade é criar uma ilusão de um computador virtual, de fácil acesso e com um grande poder computacional e dispositivos que são partilhados.

As *Grids* removem as ligações fixas entre aplicações, servidores, bases de dados, máquinas, armazenamento, entre outros, tratando tudo como um serviço virtual [24]. Assim, os recursos computacionais podem estar no mesmo ambiente ou alojados em diferentes locais geograficamente distantes.

Um *Cluster*, sinteticamente, é um sistema onde mais de dois computadores trabalham conjuntamente, a fim de realizarem tarefas, parecendo que se trata de uma só máquina de grande capacidade. A cada computador chamamos de nó. Este sistema difere da *Grid*, na medida em que, no *Cluster* existe um controlador central que permite utilizar todo o poder de processamento deste.

Na tecnologia *Grid,* o processamento poderá ser efectuado por máquinas com diferente *hardware* e a correr diferentes sistemas operativos.

Imagine-se o seguinte cenário, duas empresas sediadas em países com fuso horário diferentes poderiam formar uma *Grid*, combinando os seus servidores *Web*, de modo que uma consiga utilizar os ciclos de processamento ociosos da outra nos seus horários de pico, já que com horários diferentes, os picos de acessos aos servidores de cada empresa ocorrerão em horários diferentes.

Existe também a analogia com centrais eléctricas, em que utilizador solicita um dado ou processamento e ele é entregue, independente de onde esteja ou quando o requisite, como a electricidade em que se liga uma tomada e se faz uso dela sem se saber onde foi gerada, sendo completamente transparente e confiável para o utilizador.

De acordo com Foster *et al* [17], todas estas informações e conceitos sobre a tecnologia *Grid* são antigas. Desde a década de 60 que esta tecnologia está a ser estudada em faculdades norte americanas, onde se analisavam e desenvolviam a computação em vários dispositivos computacionais. Todos os conceitos da tecnologia *Grid* só podem ser praticados em plenitude nos dias de hoje, devido aos avanços na área de computação.

Tal como Internet, a computação *Grid* tornou-se popular nas comunidades académicas e de investigação. Entre os factores que influenciaram seu desenvolvimento destacam-se:

- As pesquisas envolviam um grande número de pessoas e estas, geralmente, encontravam-se em locais afastados, por isso foi necessário desenvolver um ambiente computacional para partilhar recursos e resultados dinamicamente;
- Também era importante escalar facilmente, para poder acomodar uma quantidade cada vez maior de dados e poder computacional;
- Manter os custos baixos.

A arquitectura *Grid* consegue solucionar estes requisitos. Estes podem escalar de forma simples e dinâmica, podem englobar máquinas localizadas em lugares diferentes, utilizando seus recursos e tempo de processamento ociosos, e podem utilizar *hardware* comum, não necessitando da utilização de máquinas de grande porte, como super computadores.

2.4 Benefícios

Um dos principais benefícios da computação *Grid* é que recursos computacionais disponíveis e espalhados dentro de uma empresa poderão ser melhor utilizados para atender as necessidades computacionais da organização.

A tecnologia *Grid*, juntamente com a computação de dados, veio para revolucionar a utilização dos computadores. Estando a computação *Grid* em crescimento, cria-se uma grande expectativa em relação ao que se pode ou não fazer com ela. Berstis [5] afirma que a tecnologia *Grid* possibilita: explorar recursos, capacidade de execução em paralelo, dispositivos e organizações virtuais e apresenta credibilidade.

Explorar recursos: Além dos recursos para execução de *jobs*, muitas máquinas também possuem discos rígidos, sendo utilizados para armazenamento de dados. Assim, a *Grid* pode ser utilizada como uma alocação de espaço disponível, como se fosse um disco apenas. Outro modo de alocar o espaço é segmentar os dados de forma que as aplicações possam ser executadas numa máquina mais próxima de onde se encontram os dados que executa, ou para garantir uma maior disponibilidade caso alguma máquina falhe.

Capacidade de execução em paralelo: Faculdade de tornar mais rápida a execução de aplicações científicas, financeiras, processamento de imagens e simulações, no aproveitamento de execução paralela.

Dispositivos e organizações virtuais: A colaboração entre os mais diversos tipos de utilizador e aplicações é outra capacidade que pode ser desenvolvida com o aparecimento da *Grid*. Recursos e máquinas podem ser agrupados para trabalharem juntos, formando o que pode ser intitulado de uma Organização Virtual (OV). Uma OV é uma entidade que partilha recursos através da *Grid* utilizando uma determinada política. Comparando-se com a Internet, seria semelhante a um *site*, mas com a diferença de poder fornecer serviços solicitados pelos utilizadores.

Credibilidade: Existem diversas maneiras de aumentar a credibilidade num sistema computacional. Processadores e discos são duplicados, de modo que caso um falhe o outro assuma o seu lugar. Fontes de energia e circuitos redundantes, geradores eléctricos, entre outros. Todas estas formas aumentam a disponibilidade e confiança num sistema, mas os seus altos custos podem torná-las impraticáveis.

Os benefícios da computação Grid atingirão diversas áreas de pesquisa, de que são exemplo a física, bio-informática, entre outras, além de permitir a realização de grandes pesquisas experimentais nas áreas ambientais, desportivas e educacionais. Os problemas tratados por estas áreas de pesquisa envolvem o consumo de muitos ciclos de processamento e/ou geram várias bases de dados independentes, que precisam de ser integradas para que pesquisadores de instituições dispersas pelo mundo possam realizar análises.

Exemplos de Sistema de Computação *GRID:*

Como já foi referido anteriormente, a computação *Grid*, tornou-se "famosa" entre as comunidades científicas e académicas. A sua utilidade já foi comprovada em diversos projectos desenvolvidos nas mais variadas áreas.

O projecto SETI@home, *Search for Extraterrestrial Intelligence* [31], é um dos projectos de maior referência sobre *Grids* no meio científico. Este projecto pretendia recolher dados obtidos por sinais de telescópios, receptores de rádio e outras fontes de controlo e distribuí-los, via Internet, para computadores. Para serem processados estes computadores, executavam os dados em busca de sinais que pudessem comprovar a existência de seres extra terrestres, aproveitando seu tempo ocioso de execução.

Outros projectos em desenvolvimento:

- China National Grid Project [7] (interligação de universidades e governo)
- Projecto eDiamond [10] (processamento de mamografias)
- DEISA [8] (interligação de laboratórios científicos)
- AccessGrid [1] (vídeo conferência e *e-learning*)
- TeraGrid [33] (interligação de universidades)

Portugal aderiu em 2001, com o projecto *European DataGrid* (mais tarde denominado projecto EGEE – *Enabling Grids for E-sciencE in Europe*) [11], que é coordenado pelo CERN. Fazem parte deste projecto, a nível nacional, o LIP – Laboratório de Instrumentação e Física Experimental de Partículas (Lisboa e Coimbra), as universidades do Porto e do Minho, do Centro de Física de Plasmas do Instituto Superior Técnico, do Instituto de Engenharia Electrónica e Telemática (IEETA) da Universidade de Aveiro e da Universidade Lusíada (Famalicão).

O EGEE criou uma colecção de metadados multimédia em apenas 16 semanas. Durante este tempo foram processadas 37 milhões de imagens, 5 Terabytes de informação. Em Portugal, o LIP participou disponibilizando recursos de computação *GRID* para efectuar a criação de metadados descritivos das imagens a partir da sua análise computacional.

2.5 Componentes

A computação *Grid* possui dois componentes principais [4]: recursos computacionais utilizados e arquitectura. Seguidamente analisamos estes dois componentes.

Recursos computacionais

Todos os recursos computacionais existentes no ambiente *Grid*, podem ser partilhados. Estes recursos são:

Computação: os processadores podem variar na velocidade, arquitectura, software, plataforma, e outros factores associados, tais como a memória, armazenamento e conectividade. Um dos modos de explorar os recursos de computação de uma *Grid* é usá-la para executar uma aplicação existente numa máquina disponível da *Grid*, em vez de executar numa máquina localmente.

Armazenamento: a *Grid* proporciona uma visão integrada de armazenamento de dados. Cada máquina na rede, normalmente, fornece uma quantidade de armazenamento para utilização da *Grid*, mesmo que temporária. O aproveitamento do espaço de

armazenamento de cada máquina pela *Grid*, é uma forma interessante de partilha deste recurso.

Comunicações: a velocidade das comunicações pode influenciar na execução de diversas tarefas. Através da *Grid* pode-se partilhar ligações, tanto internas quanto externas.

Software e licenças: alguns softwares possuem licenças caras e seria impraticável a sua instalação em todas as máquinas da *Grid*. Assim, requisições para empregar estes softwares seriam direccionadas para as máquinas que os possuem instalados. Deste modo, pode-se realizar uma melhor utilização das licenças adquiridas.

Arquitectura

A *Grid* pode ser dividida em arquitecturas, existindo várias definições para a sua descrição. De seguida serão apresentados as duas definições mais comuns para sua compreensão. A primeira será dividir uma *Grid* por camadas conforme o seu recurso. Assim sendo, podemos dividi-la em quatro camadas:

Rede: define a conectividade entre os membros da *Grid*, e pode ser considerado o sistema nervoso de uma *Grid*. É interessante notar, também, que na grande maioria das vezes os sistemas em *Grid* utilizam os mesmos tipos de *links* disponíveis para qualquer utilizador comum: *links internet*, *Ethernet* 10/100/1000Mbps, etc.

Recursos: define os recursos membros da *Grid*, como computadores, sistemas de armazenamento, sensores, etc.

***Middleware*:** é o responsável pela interconectividade dos recursos da *Grid*, bem como a segurança dos dados e comunicação, etc. Entre as suas funções também se pode referir as negociações entre máquinas. Esta camada será discutida à frente mais em pormenor.

Aplicação e Serviços: aplicações (científicas, económicas, de engenharia, etc) que trabalham na *Grid*, ferramentas de desenvolvimento, portais, etc. Pode-se também explicar a *Grid* através da sua estrutura física (hardware, redes, aplicação, etc), deste modo obtemos uma visão clara sobre as diversas camadas que constituem uma *Grid*.

Outra definição é dada por Foster et al [17] que define a arquitectura de uma *Grid* na forma de uma pilha de protocolos, semelhante à pilha de protocolos TCP/IP, como ilustra a Figura 1.

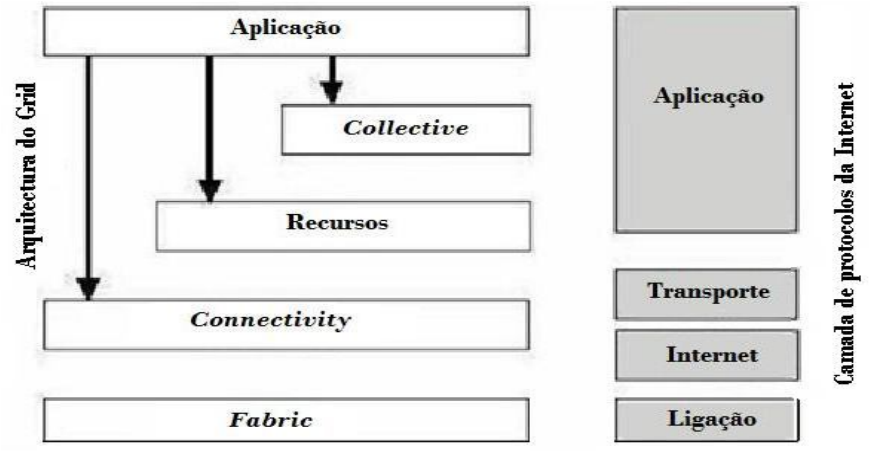

Figura 1 - Arquitectura de um Grid comparado com a arquitectura TCP/IP [17]

Fabric: interfaces para controlo local.

Esta camada fornece funcionalidades com as quais é possível a partilha de recursos pela *Grid*. Implementa operações locais e específicas para cada tipo de recurso partilhado pela *Grid*, fornecendo suporte às funções das camadas superiores.

Connectivity: comunicação fácil e segura.

A camada *Connectivity* define os protocolos de comunicação e de segurança necessários para transferências de rede específicas das *Grid*, permitindo transferência de dados entre recursos, utilizando as funcionalidades da camada *Fabric*. As necessidades supridas pela camada incluem transporte, *router* e nomeação. Actualmente estes protocolos são mapeados para a pilha de protocolos TCP/IP, especificamente Internet (IP e ICMP), transporte (TCP e UDP) e aplicação (DNS).

Resource: partilha de recursos simples.

A camada *Resource* utiliza as funcionalidades da camada *Connectivity* para a negociação segura, controlo e contabilização de operações de partilha em recursos individuais.

Collective: coordenando múltiplos recursos.

Diferente da camada *Resource*, que interage com um recurso simples, a camada *Collective* fornece protocolos e serviços que não estão associados a um recurso específico e sim a colecções destes. Para isto, ela disponibiliza algumas facilidades de partilha.

Application: a camada final da arquitectura *Grid*.

Compreende as aplicações de utilizador que operam no ambiente de uma *Grid*. As aplicações são construídas através da utilização de serviços providos por cada camada. Em cada uma destas, existem protocolos definidos que fornecem serviços como gestão e localização de recursos, acesso a dados, entre outros.

2.6 Middleware

A computação *Grid* distingue-se dos outros tipos de computação distribuída, uma vez que na computação *Grid* é introduzida uma camada de software designada por *Middleware*.

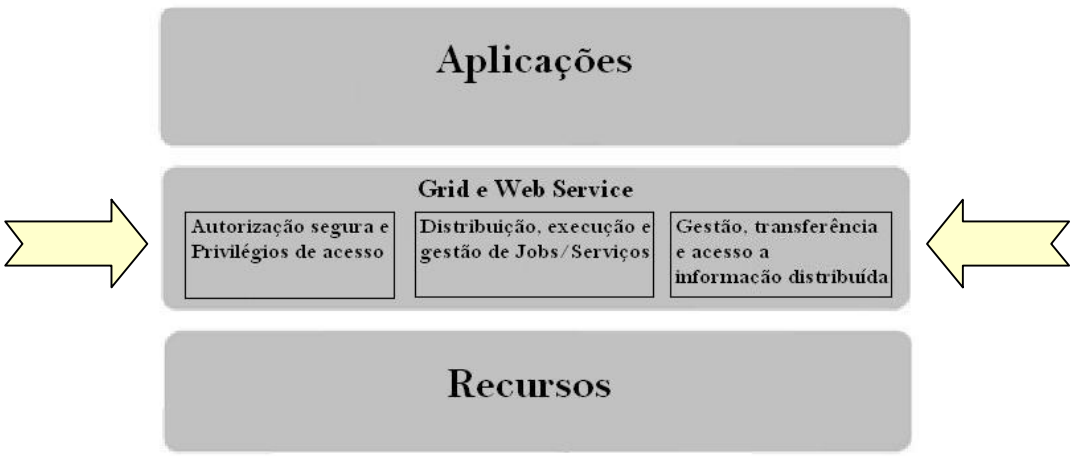

Figura 2 - *Middleware*

O *middleware*, como se pode constatar na Figura 2, é uma camada que serve para abstrair os mecanismos e protocolos usados como recursos, dos utilizadores na camada de aplicações de modo a que vejam os recursos de forma transparente. Entre as suas funções também se pode referir as negociações entre máquinas. Esta camada é, muitas vezes, constituída por um grande conjunto de *softwares*. Muitos desses *softwares* actuam

negociando transacções de dados e outros recebendo e gerindo-os. Fazendo uma analogia com corpo humano, pode-se dizer que a camada *middleware* é o cérebro da *Grid*.

Devido à heterogeneidade de recursos torna-se necessária a existência de uma camada que fomente a interoperabilidade e transparência entre os diferentes recursos, daí a necessidade do *middleware* que se deve ter de conseguir gerir com os requisitos da Tabela 1.

Autenticação e autorização	O utilizador possui uma única credencial de autenticação que lhe dá acesso a todos os recursos.
Input/Output	Pode ser efectuado de forma eficiente, segura e transparente.
Execução de programas	Pode ser efectuado de forma eficiente, segura e transparente.
Localização de recursos	Sistemas de informação permitem obter informação sobre a disponibilidade e estado da infra-estrutura.
Balanceamento de carga	Sistemas sofisticados permitem descobrir quais os melhores recursos para executar cada programa e tomam conta dos detalhes da execução.
Ferramentas e interfaces	Ajudam os utilizadores a tirar o máximo partido do *Grid*.

Tabela 1 – Requisitos do *middleware*

2.7 Normalizações

Em 2001 foi criado o *Global Grid Fórum* (GGF) [13], uma comunidade que pretende normalizar a computação *Grid*. Este tipo de computação está a ser relevante no desenvolvimento de empresas e comunidades científicas, como tal, era necessária a adopção de *standards*. O GGF realiza o seu trabalho de modo a partilhar as melhores práticas e consolidar estas normas criadas.

Uma vez que a criação de um sistema *Grid* passa pela integração de uma vasta gama de recursos heterogéneos, são referenciados alguns dos protocolos que mais contribuíram para a criação de regras e perfis que tornaram o desenvolvimento da computação *Grid*.

Uma vez que o principal problema na implementação de uma infra-estrutura em grande escala e complexa, prende-se com a integração de recursos e serviços distribuídos de forma transparente e eficiente. Assim foram estudados e desenvolvidos padrões em termos da arquitectura e interface, sendo que posteriormente serão explicados os mais importantes.

2.7.1 OGSA

De forma a conseguir integrar, virtualizar e administrar recursos e sistemas de organizações virtuais distribuídas, heterogéneas e dinâmicas, é necessário eliminar inúmeros obstáculos que usualmente separam os diferentes sistemas de computação, de modo a que estes consigam comunicar entre si independentemente da localização geográfica. A resposta a esta questão passa pela uniformização, ou seja, pela criação de normas para a *Grid*. Através de normas, torna-se possível a interoperabilidade entre componentes e sistemas heterogéneos, assim como fomentar o desenvolvimento de sistemas *Grid* mais escaláveis e robustos.

Open Grid Services Architecture [28] (OGSA) descreve uma arquitectura de serviços orientados em ambiente de computação *Grid* para uso empresarial e científico, desenvolvido dentro *Global Grid Forum* (GGF). OGSA é baseado em *Web Services*, principalmente WSDL e SOAP.

De acordo com *OGSA Roadmap document* [15], OGSA é:

- Um processo arquitectural num grupo de trabalho GGF's OGSA *Working Group* recolhe requisitos e mantém um conjunto de documentos informativos, que descrevem a arquitectura;
- Um conjunto de normas e protocolos que documentam os requisitos necessários, para um determinado componente de *hardware* ou *software*;
- Componentes de *Software* que adiram as especificações e protocolos da OGSA, tem de permitir a interoperabilidade para uso em soluções *Grid,* mesmo que sejam baseadas em implementações de várias fontes.

Resumidamente, a OGSA é uma interacção distribuída e arquitectura computacional baseada em serviços, assegurando a interoperabilidade em sistemas heterogéneos de modo a diferentes tipos de recursos possam comunicar e partilhar informação. OGSA trata-se portanto de um aperfeiçoamento de requisitos específicos necessários, para suportar a computação *Grid* através de *Web Services*.

2.7.2 WSRF

Web Services Resource Framework (WSRF) [21] é da família das especificações publicadas para *Web Services* OAISIS. Os *Web Services* normalmente são *sateless*, ou seja, não retêm informação entre invocações. Mas para aplicações em *Grid*, onde o conhecimento dos estados dos recursos é necessário, a tecnologia *Web* Service, sozinha, não consegue atender esse requisito [3]. Por isso, a *Organization for the Advancement of Structured Information Standards* (OASIS), definiu o padrão *Web Service Resource Framework* (WSRF), um *framework* aberto à utilização, para a modelagem e ingresso aos estados dos recursos usando *Web Service* [27].

O WSRF não foi associado a tecnologia *Web Service*, tendo sido criado para ser usado em conjunto com os serviços *Web*, adicionando essa nova funcionalidade. Foram criados separadamente por motivo de simplicidade.

O WSRF proporciona um conjunto de operações que permitem a implementação de *Web Services sateful*; Clientes *Web services* comunicam com serviços de recursos que permitem o armazenamento e recuperação de dados. Quando clientes comunicam com *Web services* incluem um identificador que especifica o recurso que deverá estar dentro do pedido, para haver a identificação dos recursos há o uso de uma chave única, para que todas as vezes que for necessária uma interacção com o recurso basta instruir o *Web Service* a empregar um recurso em particular através de sua chave.

Os *resources* (recursos) são entidades lógicas que devem ser identificáveis. Possuem um conjunto de propriedades que podem ser expressas num conjunto de informações XML e podem ter um tempo de vida (*lifecycle*). Juntamente com o *Web*

Service, os recursos formam o *WS-Resource* que são referenciados pela especificação *WS-Addressing*. O endereço de um *WS-Resource* é denominado de *endpoint reference* [28].

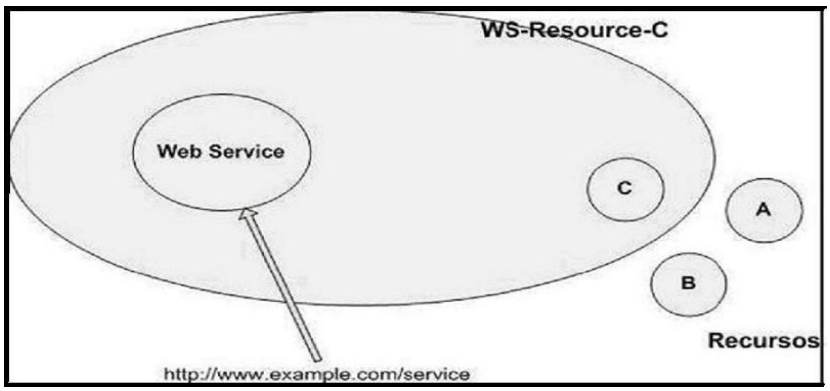

Figura 3 - Representação de um resource [28].

A Figura 3, mostra o *WR-Resource-C* que é formado pelos serviços *Web* localizados em: http://www.example.com/service e pelo *resource* "C", sendo referenciado pelo esquema XML da Figura 4.

```
<wsa:EndpointReference>
        <wsa:Address>
                http://www.example.com/service?res=C
        </wsa:Address>
        ...
</wsa:EndpointReference>
```

Figura 4 - Ficheiro XML representando o *resource* C [28].

O WRSF *framework* é um conjunto de cinco especificações de *Web Services* que definem como o *WS-Resource*, dentro do contexto de *Web Services*, modela e controla o estado dos mesmos. As especificações da WSRF são:

- *WS-ResourceLifetime*, que define mecanismos para a destruição de *WS-Resource*, incluindo a troca de mensagens, que permite a quem solicita, a destruição imediata ou programada temporalmente de *WS-Resource.*
- *WS-ResourceProperties*, definição de *WS-Resource* e mecanismos para alterar, recolher e apagar propriedades *WS-Resource*.
- *WS-RenewableReferences*, define uma forma convencional de referenciar através de uma política de informação, como obter a actualização da versão de um *Endpoint* quando este fica inválido.

- *WS-ServiceGroup* define mecanismos para subscrever eventos e notificações usando um padrão baseado em tópicos de publicação/subscrição.
- *WS-BaseFaults*, define um tipo de base para erros em XML, para usar quando ocorram erros na troca de mensagens entre *Web Services*.

2.8 Cloud Computing

Cloud computing descreve um ambiente de computação baseado numa rede de servidores, sejam virtuais ou físicos. *Cloud computing* aloja as *cloud applications*, que são as aplicações que residem nesta nuvem (*cloud*). *Cloud computing* pode ser visto como o estágio mais evoluído do conceito de virtualização.

Uma arquitectura em *cloud* é mais do que um conjunto (embora massivo) de computadores. Isto porque deve dispor de uma infra-estrutura de gestão que inclua funções de alocação dos recursos computacionais, balanceamento dinâmico do *workload* e gestão de desempenho. Daí que muitas vezes este conceito seja confundido com a computação *Grid*, aliás pode-se dizer que em parte o conceito de *cloud* deriva da *Grid* mas esta pode ou não fazer parte da *cloud* até porque, a *Grid* está mais orientada para o tratamento de dados em grande escala e de forma intensiva não sendo a opção mais acertada para o tratamento de pequenos objectos.

O primeiro benefício é uma melhor utilização dos recursos computacionais, potencializando os conceitos de consolidação e virtualização. Com *Cloud computing*, as companhias podem escalar a sua infra-estruturas rapidamente, sem ter de investir em novas infra-estruturas, treinar pessoal novo ou adquirir novas licenças software. Assim passam a ser utilizadores de serviços e como tal não possuem a infra-estrutura, software ou plataforma uma vez que esta se encontra na *Cloud* [26].

Hoje, algumas empresas do mundo Internet como a Google e a Amazon já oferecem o seu parque computacional para outras empresas. A IBM, com o *Blue Cloud*, não entra directamente no negócio de oferecer uma nuvem de servidores para aluguer, mas sim em possibilitar que as empresas construam sua própria nuvem, seja para uso interno ou para ser comercializado externamente.

3 Sistemas *Grid* Analisados

No âmbito deste projecto foram analisados os seguintes sistemas:

- Globus Toolkit;
- gLite;
- iRods;
- Alchemi;

O Globus Toolkit foi estudado pelo facto de estar estreitamente relacionado com o início da tecnologia *Grid*, daí que pareceu sensato tentar perceber as suas características e funcionalidades. Relativamente ao gLite o seu estudo deve-se ao facto de, embora a sua génese ser de projectos que estão intimamente ligados ao início da tecnologia (como por exemplo o Globus), o seu lançamento como um *middleware* independente é relativamente recente, assim como o gLite também o iRODS é muito recente daí o seu estudo. Por último o Alchemi que usa a plataforma Windows que era uma necessidade do projecto.

3.1 Aliança Globus

A aliança Globus [14] trabalha com pesquisa e desenvolvimento, para elaborar a tecnologia padrão dos sistemas que formam a *Grid*, através de uma arquitectura computacional que permita a colaboração distribuída para negócios, ciência, engenharia e demais áreas. Permite que pessoas partilhem poder computacional, base de dados e outras ferramentas *on-line* envolvendo desde corporações, instituições, ultrapassando os limites geográficos sem sacrificar a autonomia local.

Esta aliança, baseada no Laboratório Nacional da Argonne, no Instituto de Ciência da Informação da Universidade do Sul da Califórnia, na Universidade de Chicago, na Universidade de Edinburgh, no Centro Sueco para Computadores Paralelos e no Centro Nacional em Aplicações de Super Computação (NCSA), produz *software open-source* que é central para actividades da ciência e engenharia. A aliança Globus também reconhece a participação de outras organizações importantes, além de contribuintes ou mesmo utilizadores.

O Globus *Toolkit* abrange softwares de serviços, biblioteca para a segurança distribuída, gestão e monitorização de recursos e gestão de dados.

Na sua última versão, GT4, inclui componentes para a construção de sistemas que seguem a Arquitectura Aberta de Serviços para *Grid* (OGSA), tem um *Framework* definido pelo *Global Grid Forum* (GGF), sendo a aliança Globus o membro que lidera.

Assim como a *Web* tem revolucionado o acesso à informação, a aliança Globus procura alcançar um resultado similar na computação. Serão possíveis novos tipos de aplicações quando se aceder a super computadores, imagens ao vivo por satélite, armazenamento em massa e recursos *on-line* tornar-se-á tão prático como usar a *Web*.

Globus ToolKit

O *Globus ToolKit*, é um software *open source* [14] usado para desenvolver sistemas e aplicações *Grid*, actualmente está na versão 4 (GT4) e consiste em vários componentes que servem de base à implementação de um ambiente de computação *Grid*. Não se trata de uma solução final mas proporciona ferramentas para muitos dos requisitos para computação *Grid*.

Muitos dos seus componentes baseam-se em normas já existentes, e encontra-se em forma de pacotes para serem usados separadamente ou juntos no desenvolvimento de aplicações. Cada organização tem um modo único de trabalhar, gerando obstáculos na colaboração entre múltiplas organizações, tais como a incompatibilidade no armazenamento dos dados, plataformas de computadores e mesmo nas redes. O Globus *Toolkit* foi concebido para remover obstáculos que inibem a colaboração distribuída. Tendo no seu núcleo de serviços, interfaces e protocolos capazes de permitir aos utilizadores acesso remoto aos recursos, como se os recursos estivessem localizados localmente, simultaneamente ainda preserva o controlo sobre aqueles que podem usar e quando podem usar.

Outra perspectiva da estrutura do GT4, está apresentada na Figura 5, onde estão representados os principais componentes disponibilizados pelo GT4. Eles estão agrupados em 5 categorias:

- *Security* - baseada no *Grid Security Infrastructure* (GSI), são os componentes que asseguram a segurança nas comunicações,
- *Data Management* - permite a administração de grandes aglomerados de dados dentro da organização virtual.
- *Execution Management* - lida com a iniciação, monitorização, administração, organização temporal e coordenação de programas executáveis, designados normalmente, dentro da *Grid*, por *jobs*.
- *Information Services* - inclui um agregado de componentes para a descoberta e monitorização de recursos dentro da organização virtual, muitas vezes designados por *Monitoring and Discovering Services* (MDS).
- *Common Runtime* - oferece um conjunto de bibliotecas e ferramentas fundamentais essenciais para a construção tanto de *Web Services*, como de serviços de outro tipo.

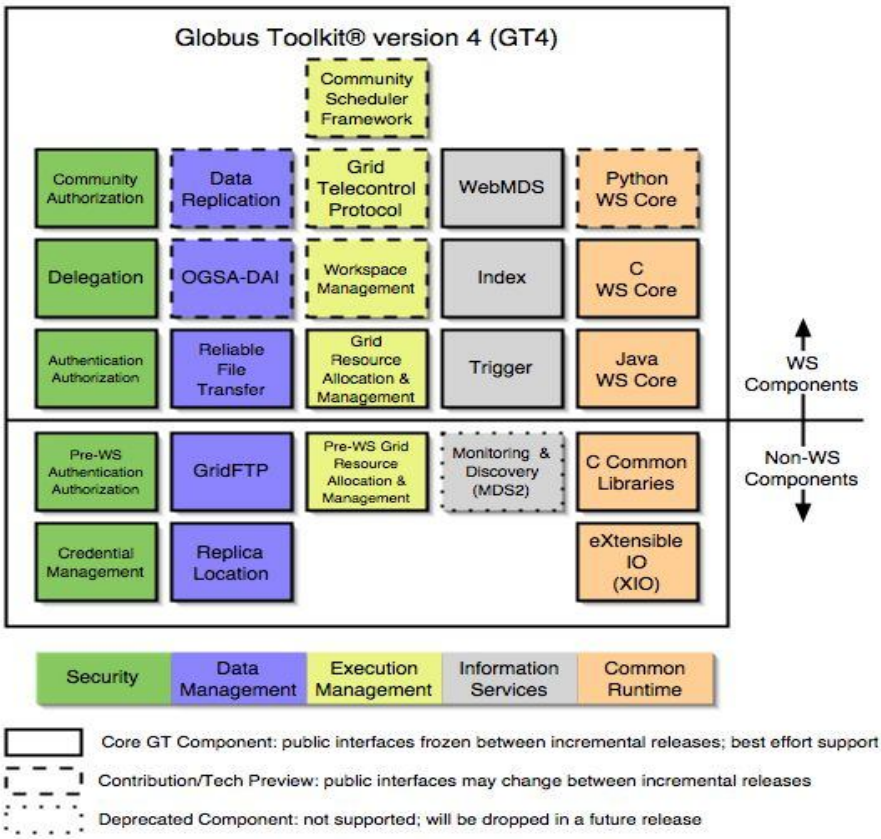

Figura 5 - Arquitectura do GT4.

Plataformas e Linguagens de programação

Globus Alliance informa que o GT4 *toolkit* foi instalado com sucesso nas seguintes plataforma: Apple Mac OS X, Debian Linux, Fedora Core Linux, HP/UX, IBM AIX, Red Hat Linux, Sun Solaris and SuSE Linux em arquitecturas implementadas para 32 e 64 bit.

A maior parte do código encontra-se em Java mas são suportadas as linguagens de programação C e Python.

3.2 gLite

EGEE (*Enabling Grids for* E-*sciencE*) [11] decidiu desenvolver o glite em duas fases, inicialmente baseou-se no trabalho desenvolvido no projecto anterior da *European Data Grid* (EDG), mais tarde desenvolvido para o LCG *Middleware* stack, que foi usado na estrutura inicial pelo EGEE. Em paralelo o EGEE ia desenvolvendo a maior parte de uma nova solução para *middleware* stack, o glite.

O gLite é distribuído de acordo com uma licença de código aberto (*open source code license*) e é usado pelas aplicações científicas e empresariais executadas na rede do EGEE, o gLite integra componentes de outros projectos de *middleware*, tais como o Condor, Globus *Toolkit* entre outros e com as entidades que estabelecem procedimentos padrões para a implementação das tecnologias *Grid*, como o *Open Grid Forum (OGF)*, que assegura a interoperabilidade das diversas implementações de *middleware* disponíveis e que trabalha de forma a conciliar mecanismos padrões universalmente aceites por todas as comunidades *Grid*.

Os diferentes serviços do *middleware* gLite proporcionam um compromisso fiável entre segurança, monitorização, gestão e manuseamento de dados e foram desenvolvidos de modo a seguirem uma arquitectura orientada para serviços *(Service Oriented Architecture (SOA)),*Figura 6. A maioria destas funcionalidades segue também as recomendações para uma boa interoperabilidade entre serviços *Web (Web Services Interoperability (WSI))*, constituindo desta forma uma camada padrão de serviços *Grid* em que a adição de novos componentes acabará por se tornar um procedimento trivial.

Todos os serviços *Grid* necessitam de ser desenvolvidos segundo procedimentos padrão para que as suas funcionalidades sejam reconhecidas como estáveis e fiáveis.

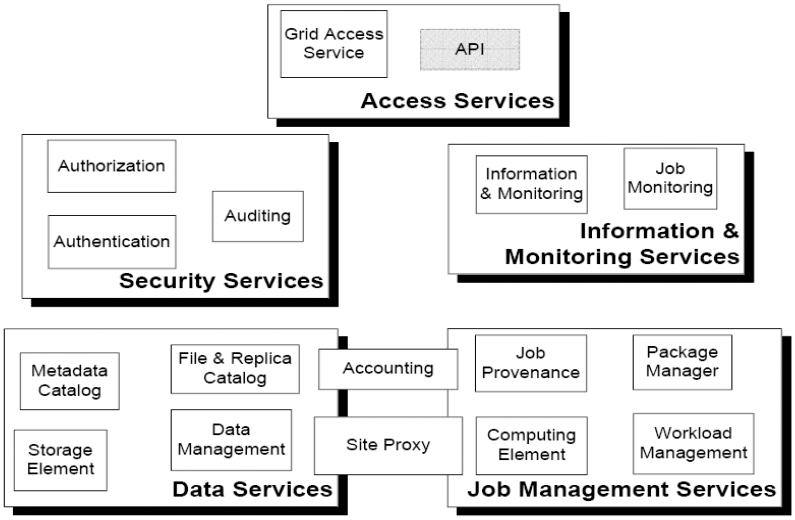

Figura 6 - Arquitectura gLite.

API and Grid Access Service fornece um *Framework* comum através do qual o utilizador pode obter acesso. Irá gerir os ciclos de vida da *Grid Service* disponíveis para um utilizador, de acordo com os seus privilégios. A *API* não é, obviamente, um serviço por si só.

Security services engloba autenticação, autorização, auditoria e serviços que permitem a identificação das entidades (utilizadores, sistemas e serviços), permite ou nega o acesso a serviços e recursos assim como fornece informações para a análise *post-mortem* de eventos relacionados à segurança. Ele também fornece funcionalidades para a confidencialidade dos dados e um *site proxy*, ou seja, através do *site* pode-se controlar o acesso a padrões de aplicações de rede e serviços *Grid* utilizando os seus recursos.

Information and Monitoring Services fornece um mecanismo para publicar e consumir informações e usá-la para efeitos de controlo. O sistema de acompanhamento e de informação pode ser usado directamente para publicar, por exemplo, informações sobre os recursos na *Grid*.

Mais serviços especializados, tais como o *Job Monitoring Service*, serão construídos em cima. O serviço de informação subjacentes serão capazes de lidar com os fluxos de dados que se fundem e republicam esses fluxos. O sistema depende do registo da localização dos editores de informação e qual o subconjunto do total das informações que estão publicadas. Isto permite aos consumidores consultar o sistema de informação sem ter que saber onde a informação foi publicada.

No entanto, todas as informações publicadas carregam com elas a hora e a data em que foi publicado pela primeira vez (ou seja, quando a "medida" foi feita), de modo a identifica-las, como também a identidade do editor e de onde ela foi publicada. Esta informação não é modificável, mesmo que os dados sejam republicados. Na verdade os dados não podem ser modificadas no sistema, de modo a evitar eventuais inconsistências quando os dados são republicados.

Finalmente existe uma regra, baseada no regime de autorização para garantir que as pessoas só podem ler ou escrever dentro de sua autoridade.

Job Management Services são os principais serviços associados ao trabalho de gestão e execução: *computing element*, o *workload management*, *job provenance*, e *package manager services*.

Computing Element (CE) proporciona a virtualização de um recurso de computação (normalmente numa fila de um cluster, mas também super computadores ou mesmo uma única *workstations*). Fornece informações sobre os recursos subjacentes, e oferece uma interface comum a apresentar e gerir *Jobs* nos recursos.

Workload Management System (WMS) é um nível *metascheduler Grid* que faz o escalonamento de *Jobs* disponíveis sobre o CE, de acordo com as preferências do utilizador e várias políticas. Ele também mantém registo dos *Jobs* que gere de forma consistente através de *logging* e *bookkeeping service*.

Job Provenance (JP) *service* fornece informações constantes sobre os *Jobs* executados na infra-estrutura *Grid* para mais tarde inspeccionar e eventualmente voltar a ser executado.

Finalmente, o *Package Manager* (PM*) service* confere dinâmica na implantação de aplicações de *software*.

Data Services os três principais grupos de serviços que dizem respeito aos dados de ficheiros e acesso são: *Storage Element*, *Catalog Services* e *Data Management*. Estreitamente relacionados com os dados estão os serviços de segurança e os serviços relacionados com o *Package Manager*.

O *Storage Element* (SE), proporciona a virtualização de armazenamento de um recurso (que pode chegar a partir de um disco simples ou de sistemas complexos de servidores de armazenamento hierárquico em fita), tanto quanto a CE faz por recursos computacionais.

Os *catalog services* rastreiam a localização dos dados pertinentes, bem como metadados (por exemplo, *checksums* e *filesizes*) e os dados permitindo a circulação de serviços eficientes de dados gerindo transferências entre SE. O acesso aos ficheiros é controlado pelo *Access Control List* (ACL). Uma aplicação não deve estar à espera que os gLite *Services* especifiquem onde devem ser armazenados os metadados, estes devem ser especificados através de *metadata catalogs*.

Para o utilizador do EGEE *data services* a abstracção que está a ser apresentada é a de um sistema de ficheiros global. Uma aplicação cliente pode ser parecida com um *shell* Unix (como no *AliEn*), que pode navegar sem dificuldades neste sistema de ficheiros virtual, listando ficheiros, alterando directórios, etc.

Plataformas e linguagens de programação

O único sistema operativo actualmente suportado pelo gLite 3.1 é *Scientific Linux 3* e a arquitectura suportada é IA32. prevê-se que em breve esteja disponível para Linux4 Científico e os x86 64 e arquitecturas IA64.

O *Job Description Language* (JDL) é uma linguagem de alto nível, baseado na *Classified Advertisement* (*ClassAd*), que serve para designar *Jobs* e agregados de *Jobs* com relações arbitrárias de dependência. A JDL é utilizada em WLCG / EGEE para especificar o

Job desejado as características e limitações, que são tidas em conta pela *Workload Management System* (WMS) para seleccionar os melhores recursos para executar o trabalho.

3.3 iRODS

No *Data Intenssive Cyber Enviromments* (DICE) [9], com base na Universidade da Carolina do Norte e na Universidade Californiana de San Diego, foram realizadas investigações em políticas de gestão de dados em larga escala, que designaram de iRODS (*integrated Rules-Oriented Data Systems*). O iRODS [22] é um *middleware* adaptável que fornece uma forma flexível, extensível e prática para gestão de dados. Funciona como um conjunto de micro serviços que são controlados por regras de um utilizador ou de administrador. Estas regras são construídas a partir de pequenas funções pré-construídas que associadas a um conjunto de condições de activação criam uma macro tarefa. Esta pode ser utilizada para um determinado acontecimento ou ser publicada para uso de vários utilizadores, acrescentando mais blocos para construção de uma tarefa mais complexa.

O controlo pode ser decidido pelos utilizadores dentro limitações impostas pelo sistema. Assim, as mudanças para um determinado processo ou política pode ser facilmente construída pelo utilizador, testado e implementado sem o auxílio de sistema. O utilizador também pode definir quando é que essas regras devem ser usadas

A programação das regras no iRODS pode ser encarada como um bloco de lego, sendo que esses blocos no iRODS são os micro-serviços (pequenas, bem definidas procedimentos/funções, já referidos atrás, que executam uma determinada tarefa).

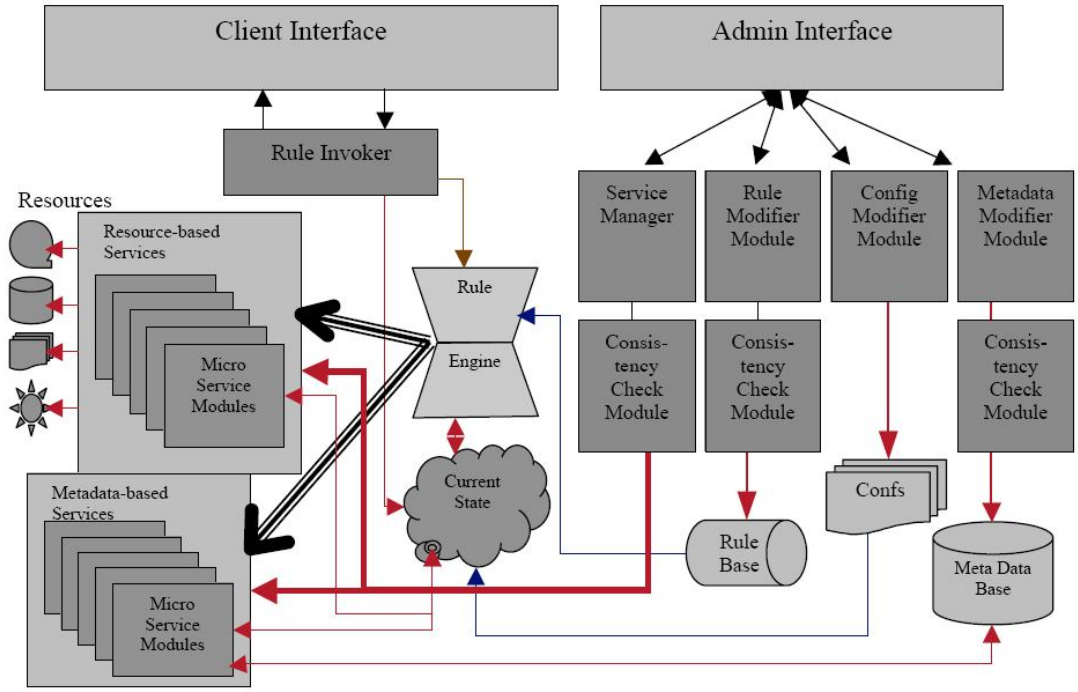

Figura 7 - Arquitectura iRODS

Como se pode constatar na Figura 7 a arquitectura do iRODS tem uma interface para clientes e outra para administradores, sendo que a principal diferença entre elas reside no facto, de em "modo de administrador" ter acesso a comandos para conceber e alterar regras, enquanto em "modo de utilizador" apenas faz uso destas, ou seja, em "modo administrador" faz a gestão das regras através de configuração destas, enquanto o utilizador quando invoca um serviço, dispara uma regra que usa a informação da base de dados das regras (*Rule Base*), estado e a base de dados de Metadados para invocar um micro-serviço.

Plataformas e Linguagens de Programação

As plataformas suportadas são: Linux, Solaris, Macintosh, AIX, Windows (Vista, XP).

3.4 Alchemi

Uma vez que um grande número de máquinas, privadas e de empresas, correm sistemas Windows parece bastante razoável que seja desenvolvido para este tipo de plataforma uma solução para computação *Grid*, foi isso que a *GridBus* decidiu desenvolver, o Alchemi [2].

O projecto Alchemi está a ser desenvolvido com base no Microsoft's *.NET framework*, oferecendo mecanismos de execução e desenvolvimento necessários para progressão aplicações voltadas ao ambiente de computação *Grid*. O Microsoft .NET *framework* tem um conjunto de ferramentas que permite resolver os problemas atrás mencionados, uma vez que suporta execução remota (via .Net *Remoting* e *Web Services*), *Multithreading*, segurança, programação assíncrona, administração de execução e desenvolvimento em várias linguagens, tornando-se a plataforma ideal para a camada de *middleware* da computação *Grid*.

Apesar dos conceitos da computação *Grid* serem simples a sua implementação prática tem uma série de desafios. Há questões fundamentais que precisam de ser tratadas tais como a segurança, a heterogeneidade, a fiabilidade, programação e gestão de recursos.

O desenvolvimento do Alchemi teve como objectivo tornar a construção e desenvolvimento da computação *Grid* o mais fácil possível, de uma maneira a não sacrificar a flexibilidade, escalabilidade, robustez e extensibilidade.

A arquitectura do Alchemi Figura 8, segue o paradigma de programação paralela *master-worker*, em que um componente central distribui unidades independentes de execução paralela para nós de trabalho e executa-os. À unidade mais pequena de execução paralela independente dá-se o nome de *Grid Thread*, sendo esta conceptualmente idêntica a uma *thread* dos sistemas operativos multitarefa. Uma aplicação *Grid* é então designada como uma aplicação que é executada em *Grid* e consiste num determinado número de *Grid threads*.

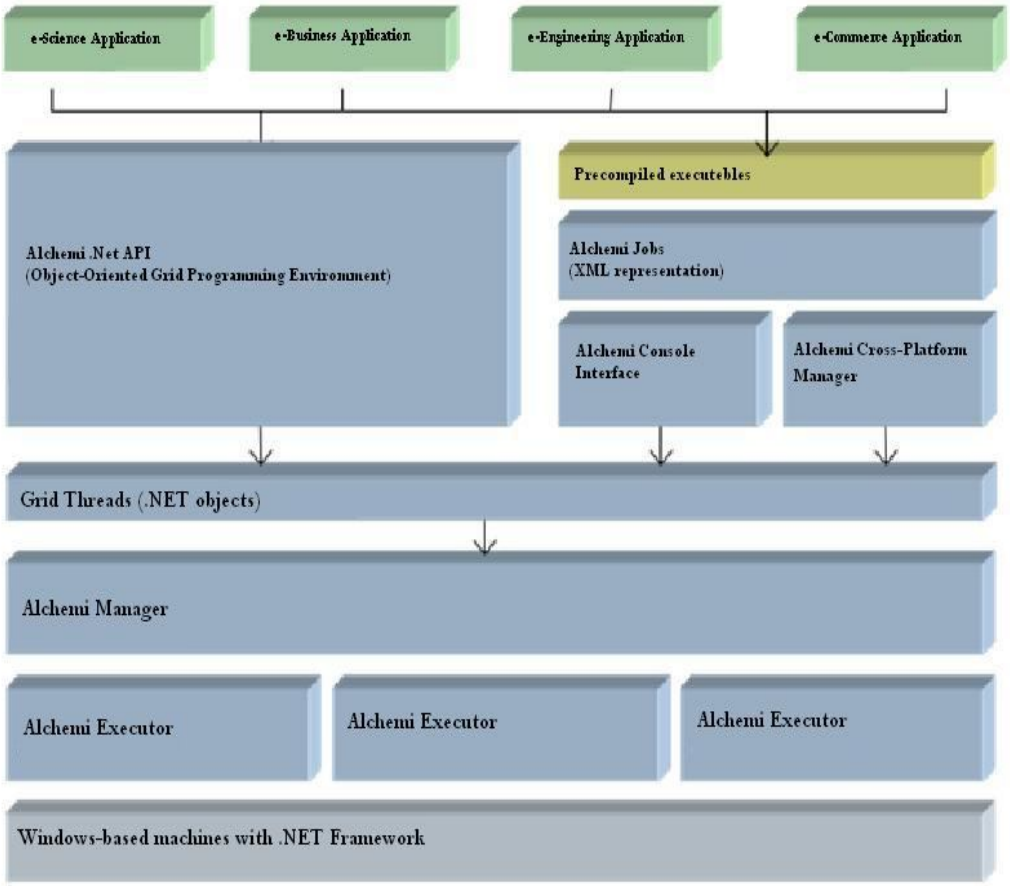

Figura 8 – Arquitectura Alchemi

O Alchemi é constituído por quatro componentes distribuídos como se pode constatar na Figura 9:

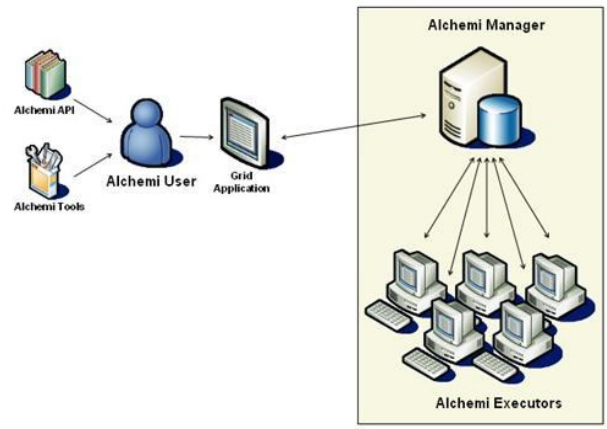

Figura 9 – Componentes do Alchemi

Manager

O *Manager* gere a execução de aplicações e fornece serviços de rede associadas à gestão da execução dos *thread*. Os executores registam-se com o *Manager* que, por sua vez, mantém um registo da sua disponibilidade. *Threads* recebidas do *Owner* são colocadas numa *"pool"* e programados para serem executados sobre os vários executores disponíveis. A prioridade de cada thread pode ser explicitamente especificada quando é criada dentro do *Owner*, mas é atribuída a máxima prioridade, por padrão, se nenhuma for especificada as *Threads* são colocadas numa fila, do tipo "primeiro a chegar primeiro a ser atendido" (*First Come First Served* (FCFS)). Os executores devolvem *threads* para o *Manager* que são posteriormente recolhidos pelo respectivo *Owner*.

Executor

O *Executor* aceita *threads* a partir do *Manager* e executa-as. Pode ser configurado para trabalharem em modo dedicado, ou seja, o recurso é gerido pelo *Manager* ou em modo não dedicado em que o recurso é gerido pelo utilizador.

Quando uma comunicação bidireccional é possível, e a execução dedicada for desejada o executor expõe uma interface (*IExecutor*), para que o *Manager* possa comunicar directamente com ele. Neste caso, o *Manager* explicitamente instrui o executor para executar *threads*, resultando numa gestão centralizada dos recursos em que o executor reside. Assim, a execução Alchemi prevê o duplo benefício de:

- Gestão flexível dos recursos, ou seja, com gestão centralizado e execução dedicada *versus* gestão descentralizada e organizações execução não-dedicada;
- Implantação do componente pode ser não-dedicada quando uma comunicação bidireccional não é desejada ou possível (por exemplo, quando está atrás de um *firewall* ou NAT / servidor *proxy*).

Assim, a execução dedicada é mais adequada quando o *Manager* e o *Executor* se encontram na mesma rede local (LAN) enquanto execução não dedicada é mais apropriado quando o *Manager* e o *Executor* estão ligados através da Internet.

Owner

Aplicações *Grid* são criadas utilizando o Alchemi API e são executadas sobre o componente *Owner*. *Owner* "possui" a aplicação e fornece serviços relacionados com a propriedade de um aplicativo e suas *threads*. O *Owner* submete *threads* ao *Manager* recolhendo-as quando estão concluídas por intermédio do promotor Alchemi API.

The Cross-Platform Manager

Cross-Platform Manager, é um sub-componente opcional do *Manager*. Trata-se de um serviço *Web* que implementa uma interface genérica que expõe uma parte da funcionalidade do *Manager* de forma a permitir ao Alchemi gerir a execução de *Grid Jobs* (em oposição a aplicações *Grid* utilizando o modelo do Alchemi de *Grid thread*). *Jobs* submetidos ao *Cross-Platform Manager* são traduzidos para uma forma que seja aceite pelo *Manager* (ou seja, *Grid threads*), que é então programado e executado como descrito acima. Assim, para além de apoiar o *grid-enabling* nas aplicações existentes, o *Cross-Platform Manager* permite a interoperabilidade entre outras *Grid middleware* e o Alchemi em qualquer plataforma que suporte serviços *Web*.

Plataformas e Linguagens de Programação

Proporciona uma API para C# e C++.

Foi desenvolvido para .NET, assim todas as máquinas a correr qualquer *software* Alchemi devem ter .NET *framework* instalado.

Nota: Alchemi apenas foi testado no Windows

3.5 Comparativo

	Plataforma	Linguagem de programação	Licença
Globus Toolkit	Linux	Java, C/C++	Sim
gLite	Linux	JDL	Sim
iRODS	Várias	Java, C/C++	Não
Alchemi	Windows	C# (.NET)	Não

Tabela 2 - Comparativo dos diferentes *middleware´s.*

A Tabela 2 faz uma síntese dos *middleware* analisados, além disso pode-se dizer que todos se encontravam muito bem documentados. Tanto o *Globus toolkit* como o gLite necessitam de uma licença para se poder ter acesso à *Grid*, o que já não é necessário para o iRODS nem para o Alchemi. Relativamente às características de cada um constatou-se que são bastante semelhantes entre eles, sendo a principal semelhança o facto de todos seguirem uma arquitectura orientada aos serviços (SOA), bem como de implementarem o standard WSRF. Sendo que a principal diferença entre o Alchemi e os outros se prende com a linguagem usada, que no caso do gLite usa uma linguagem específica.

Uma vez que se pretendia um sistema para ser executado em Windows, dispensando deste modo o Globus e o gLite, decidiu-se optar pelo Alchemi como *middleware* a adoptar para este projecto dada a sua simplicidade de instalação e o facto de usar como linguagem de programação C# (.NET) mais orientada para um ambiente *Windows*.

4 *Grid* para Suporte de Bibliotecas Digitais

Este projecto tem como objectivo a implementação de uma aplicação que aproveite todas as potencialidades da tecnologia *Grid*, para aplicação nas bibliotecas digitais.

A primeira parte deste projecto, descrita nos capítulos anteriores, refere-se à necessidade de compreensão da tecnologia *Grid* e estudo de sistemas *Grid*, uma vez que ela é fundamental para a aplicação deste projecto.

Antes de mais, pode-se definir uma Biblioteca Digital como "um agrupamento de meios informáticos, de armazenamento e de comunicação, conjuntamente com o conteúdo e software necessários para reproduzir, emular e estender os serviços fornecidos pelas bibliotecas convencionais baseadas em papel e em outros meios de colecção, catalogação, busca e disseminação da informação. Uma Biblioteca Digital de serviço completo, terá de alcançar todos os serviços das bibliotecas tradicionais e também de explorar as conhecidas vantagens do armazenamento digital, pesquisa e comunicação" [16].

Uma definição mais recente e objectiva é a seguinte: "Uma Biblioteca Digital é a colecção de serviços e de objectos de informação, sua organização, estrutura e apresentação, suporta o relacionamento dos utilizadores com os objectos de informação, disponíveis directa ou indirectamente via meio electrónico/digital" [23].

O SinBAD (Sistema Integrado para Bibliotecas e Arquivos Digitais) é um sistema integrado que permite o acesso à Biblioteca Digital da Universidade de Aveiro. Baseado nas tecnologias *Web*, este sistema está integrado com todos os sistemas já existentes na UA, nomeadamente o sistema bibliográfico, constituindo, desta forma, o portal de entrada da Biblioteca Digital da UA. Assim sendo, para o desenvolvimento das Bibliotecas Digitais existe a necessidade de conversão para formato digital dos mais diferentes objectos, tais como publicações, livros, fotografias, etc. Uma vez que alguns destes objectos são de grandes dimensões a sua conversão pode levar imenso tempo, sendo

esta a parte crucial deste projecto, uma vez que as potencialidades da computação *Grid* podem ser testadas de forma a tornar este processo mais rápido.

A segunda parte do projecto consiste na instalação do software Alchemi de forma a construir o sistema *Grid* e no desenvolvimento de quatro *Web Services*; o primeiro *Web Service* que permite a extracção do texto de ficheiros PDF, *Web service* em que se usa o OCR (*Optical Character Recognition*) para reconhecimento de texto, o terceiro que redimensiona-se e converte-se imagens, e por último um *Web Service* para remoção das margens brancas de documentos.

4.1 Apresentação da Aplicação

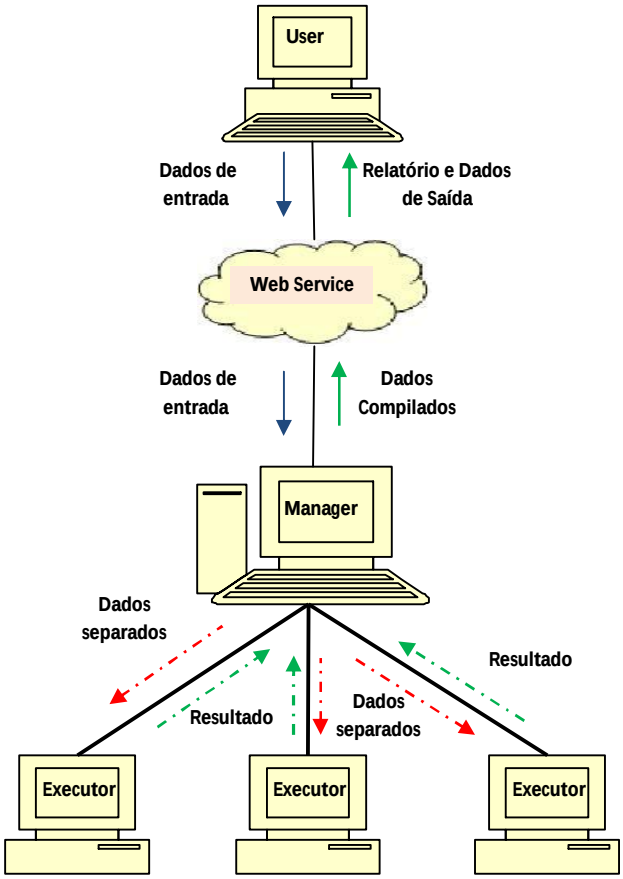

Figura 10 - Diagrama dos *Web service*

Pretende-se com esta Dissertação o desenvolvimento de um conjunto de serviços para transformação de ficheiros. Como se tratam de ficheiros de tamanho razoável e o processo de transformação é intensivo e de certa forma repetitivo, sendo cada repetição

independente das anteriores e das seguintes, levando por isso muito tempo a ser executado. Sendo assim, para minimizar o tempo dispendido na execução, irá usar-se o Alchemi para distribuir a tarefas de cada um dos serviços, através da Figura 10 será esquematizado todo o processo.

Assim para cada um dos serviços o utilizador terá de fazer o *upload* dos documentos para a máquina onde se encontra instalado o *Manager,* que tem a informação acerca do número de *Executor's* livres e dos métodos, de forma a poder atribuir tarefas a cada um. Assim o *Manager* divide o (s) documento (s) de maneira a criar subtarefas independentes entre si incluindo cada uma numa *thread* que será enviada pelo *Manager* para os *Executor's* disponíveis. Com as *threads*, é também transferido o código remoto de modo a *thread* ser tratada como se estivesse no computador que iniciou todo o processo.

Os *Executor's* por sua vez procedem à transformação das *threads* recebidas e enviam de volta o resultado ao *Manager*.

Finalmente, o *Manager,* depois de recolhido os resultados dos *Executor´s*, agrega estes resultados e envia-os ao *User,* obtendo um resultado efectuado de forma paralela e distribuída através de um processo simples, sem recorrer a configurações da parte do utilizador.

4.2 Abordagem

A abordagem usada para implementação dos *Web Services* foi modular, ou seja, partindo-se do paradigma orientado aos objectos. Embora os serviços não apresentem nenhum tipo de estrutura em termos de persistência, isto é, não é necessário transmitir dados entre as duas execuções diferentes do mesmo serviço, a hipótese de poder empregar um paradigma eficaz através do desenho de tarefas simples e estruturadas, levaram ao uso de classes C# com métodos o mais simples possíveis, sendo posteriormente usadas na função *main*.

Seguidamente irá proceder-se à explicação de cada um dos serviços nomeadamente;

4.2.1 ExtPDFtoTXT;

Objectivo: A partir de um ficheiro *Portable Document Format* (PDF), incluir metadados textuais, sendo estes extraídos e devolvidos como uma lista.

Recursos: Usou-se como biblioteca PDFBox [30] para acesso aos ficheiros PDF, mais especificamente usou-se a capacidade de carregar um ficheiro em memória, separá-lo em várias páginas e extraindo o texto numa página de cada vez.

Algoritmo: Com o uso Biblioteca PDFBox carregou-se o documento, usando de seguida uma classe que separa o documento em páginas, assim como fornecer a informação da quantidade de páginas, para de seguida cada pagina ser trabalhada separadamente.

Código Local:

1. *Upload* do ficheiro .pdf;
2. Carregar ficheiro em memória;
3. Separar ficheiro em páginas, sendo cada página um ficheiro pdf em memória inserida num objecto C# que inclui também o número de página;
4. Criar *threads*, cada uma com n número de objectos, n variável tendo em conta um parâmetro atribuído;
5. Enviar *threads* para o *Manager*;
6. Receber *threads* do *Manager*;
7. Cria um ficheiro com o número de páginas seguido do texto e outro ficheiro de relatório (*log*) com os tempos de execução;

Código Remoto:

1. Extrair o PDF do objecto C#;
2. Extrair texto do PDF;
3. Introduzir texto em objecto C# com número de página;
4. Devolver ao *Manager* o objecto.

4.2.2 ImageConverter

Objectivo: Alterar um grupo de ficheiros de imagem, comprimidos num ficheiro ZIP, para um formato de imagem e resolução atribuída pelo utilizador.

Recursos: As Bibliotecas nativas do ambiente .Net apresentaram métodos suficientes para a conversão de imagens, assim como a alteração da sua resolução.

Algoritmo: Com recurso às bibliotecas nativas .Net carregam-se os vários ficheiros para memória para um formato de dados que contem a informação da imagem, sendo depois alterada a resolução, o seu tamanho e finalmente os dados da imagem são guardados no formato de dados escolhido pelo utilizador.

Código Local:

1. *Upload* do ficheiro ZIP de imagens;
2. Carregar ficheiros em memória;
3. Criar threads, cada uma com n número de objectos, n variável tendo em conta um parâmetro atribuído;
4. Enviar *threads* para o *Manager*;
5. Receber *threads* do *Manager*;
6. Devolve o ficheiro Zip com as imagens processadas e um ficheiro de relatório;

Código Remoto:

1. Extrair as imagens das *threads*;
2. Transformar resolução da imagem;
3. Transforma o tamanho da imagem para corresponder a nova resolução;
4. Converte para formato desejado;
5. Insere nova imagem na memória da *thread*;
6. Devolver *thread* ao *Manager*.

4.2.3 FrameRemover

Objectivo: Enquadrar uma imagem, retirando as margens ficando só com os dados necessários.

Recursos: As Bibliotecas nativas do ambiente .Net apresentaram métodos suficientes para a remoção da moldura das imagens.

Algoritmo: Com recurso às bibliotecas nativas .Net carrega-se os vários ficheiros para memória. Assume-se primeiro que na moldura são todos os pixéis iguais ao pixel (0,0) o mais alto à esquerda. Sendo assim pode-se retirar qualquer moldura, sem restrições à sua cor. Depois faz-se varrimentos horizontais em todas linhas de pixéis da imagem da esquerda para a direita, para se obter o tamanho da moldura a esquerda. Fazendo de seguida o processo análogo, mas da direita para a esquerda, obtendo o tamanho da moldura à direita. Analogamente fazem-se varrimentos verticais para detectar os tamanhos da moldura superior e inferior. De seguida faz-se uma cópia da imagem original pixel a pixel, sendo o pixel (0,0) equivalente ao pixel (n,m) da página original, onde n é a largura da moldura esquerda e m a da moldura superior copiando até atingir a moldura direita e a moldura inferior.

Código Local:

1. *Upload* do ficheiro ZIP de imagens;
2. Carregar ficheiros em memória;
3. Criar th*reads*, cada uma com n número de objectos, n variável tendo em conta um parâmetro atribuído;
4. Enviar *threads* para o *Manager*;
5. Receber *threads* do *Manager*;
6. Devolve o ficheiro Zip com as imagens processadas e um ficheiro de relatório.

Código Remoto:

1. Extrair as imagens das *threads*;

2. Varrimento horizontal da esquerda para direita detecção largura da margem esquerda;
3. Varrimento horizontal da direita para esquerda detecção largura da margem direita;
4. Varrimento vertical de cima para baixo detecção largura da margem superior;
5. Varrimento vertical da baixo para cima detecção largura da margem inferior;
6. Copiar Imagem ignorando moldura;
7. Inserir Imagem numa nova *thread*;
8. Devolver *thread* ao *Manager*.

4.2.4 OCR

Objectivo: Tendo um grupo de imagens como dado de entrada, extrair texto que esteja "desenhado" nos dados de entrada, este tipo de operação é normalmente descrito como *Optical Character Recognition* (OCR).

Recursos: Usou-se a biblioteca Tesseract através da extensão tessnet, usando como reconhecimento de palavras o dicionário para a língua portuguesa disponível na página da biblioteca.

Algoritmo: Com as bibliotecas nativas .Net carregam-se as imagens, usando depois para cada imagem a biblioteca tesseract, passando como parâmetros a localização do dicionário de palavras e a linguagem a filtrar, obtendo depois palavra a palavra o texto da imagem.

Código Local:

1. *Upload* do ficheiro ZIP de imagens;
2. Carregar ficheiros em memória;
3. Criar *threads*, cada uma com n número de objectos, n variável tendo em conta um parâmetro atribuído;
4. Enviar *threads* para o *Manager*;
5. Receber *threads* do *Manager*;

6. Devolve o ficheiro .txt com o texto extraído das imagens processadas .

Código Remoto:

1. Extrair as imagens das *threads*;
2. Extrair palavras da imagem
3. Juntar palavras numa variável de texto
4. Inserir variável de texto na *thread*;
5. Devolver *thread* ao *Manager*.

5 Resultados

Os resultados a seguir apresentados foram obtidos fazendo o upload de ficheiros, PDF e ZIP, de tamanho variável e em diferentes modalidades, com um, dois e três *Executor´s* de modo a avaliar o desempenho dos diferentes serviços. Seguidamente serão apresentados os resultados obtidos em cada serviço. Em que surge os tempos de preparação e de paralelização, sendo o tempo de preparação a designação atribuída ao tempo de criação das *threads* e envio destas, do *Manager* para cada *Executor* e o tempo de paralelização os tempos de execução de cada *thread* e seu retorno ao *Manager*, bem como da apresentação do resultado.

Os ficheiros usados para realização dos testes no *Web Service* ExtPDFtoTXT foram, um ficheiro com 229KB, 463KB e um outro de 3,6MB, em que existiu a preocupação de o tamanho destes ir aumentando para deste modo melhor avaliar o desempenho do Alchemi.

Para os outros *Web Services* foram usados os seguintes ficheiros ZIP:

- 3,25MB – com 10 imagens;
- 7,65MB – com 25 imagens;
- 14,2MB – com 50 imagens (este apenas usado no ImageConverter);

De referir também que nos testes efectuados, apenas com três *Executor´s* é que um deles se encontrava na mesma máquina com o *Manager*, ou seja, nos testes efectuados com um e dois *Executor´s* o *Manager* e os *Executor´s* encontravam-se em máquinas distintas.

5.1 ExtPDFtoTXTWS

Depois de lançado o *Web Service* surgem quatro funções (Figura 11) para serem invocadas:

- MySetFileExecuting – função de teste para ficheiros locais;
- ReturnTXTOfByte – função auxiliar do processo de execução;
- ReturnTxtResultOfGuid – devolve o texto da execução (GUID);
- SetFileExecuting – usada para submeter o PDF a ser tratado;

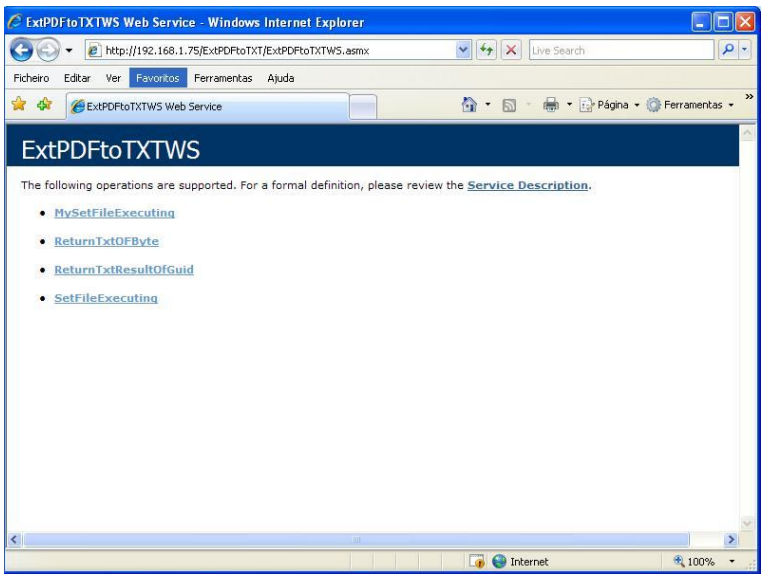

Figura 11 - *Web Service* ExtPDFtoTXTWS.

Seleccionada a função MySetFileExecuting será necessário introduzir o caminho (*path*) até ao ficheiro PDF a ser processado (Figura 12).

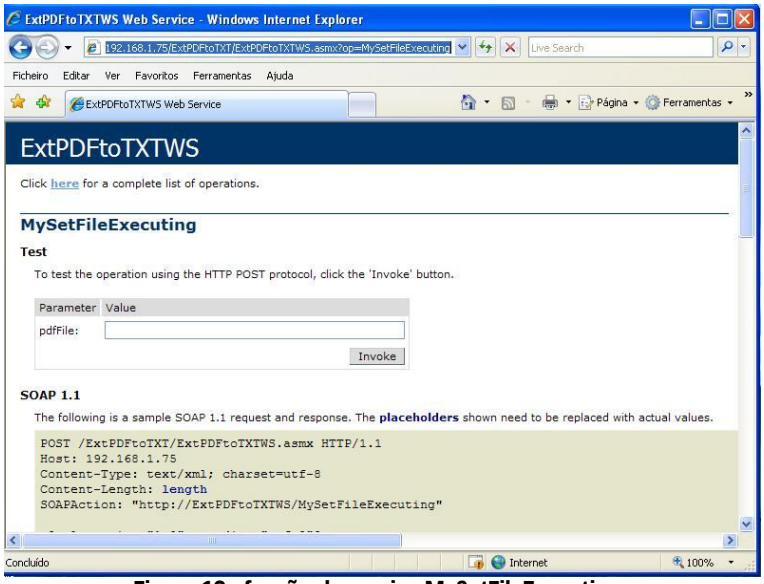

Figura 12 - função do serviço MySetFileExecuting.

Na Tabela 3 apresentam-se os resultados temporais dos três ficheiros com apenas um *Executor*.

	Capacidade de Processamento = 2,235 GHz					
	Documentos					
	229KB		**463KB**		**3,6MB**	
	Preparação	Paralelização	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:03,344s	00m:06,379s	00m:03,675s	00m:09,834s	00m:04,396s	00m:23,383s
2ª Execução	00m:03,575s	00m:06,619s	00m:03,665s	00m:09,453s	00m:04,326s	00m:24,064s
3ª Execução	00m:04,866s	00m:07,991s	00m:03,485s	00m:09,533s	00m:04,877s	00m:26,267s
4ª Execução	00m:04,055s	00m:06,248s	00m:03,266s	00m:10,595s	00m:04,406s	00m:23,834s
5ª Execução	00m:03,705s	00m:06,539s	00m:03,745s	00m:10,284s	00m:04,316s	00m:23,864s

Tabela 3 - Resultados da aplicação ExtPDFtoTXT com um *Executor*.

Na Tabela 3 apresentam-se os resultados temporais dos três ficheiros com dois *Executor´s*.

	Capacidade de Processamento = 5,242 GHz					
	Documentos					
	229KB		**463KB**		**3,6MB**	
	Preparação	Paralelização	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:03,885s	00m:05,678s	00m:04,346s	00m:07,370s	00m:04,897s	00m:12,157s
2ª Execução	00m:04,045s	00m:07,040s	00m:04,346s	00m:07,060s	00m:04,376s	00m:10,665s
3ª Execução	00m:03,234s	00m:05,688s	00m:04,216s	00m:07,380s	00m:04,426s	00m:11,326s
4ª Execução	00m:03,875s	00m:06,299s	00m:03,575s	00m:06,899s	00m:04,616s	00m:12,798s
5ª Execução	00m:04,135s	00m:06,269s	00m:03,625s	00m:06,729s	00m:04,506s	00m:13,339s

Tabela 4 - Resultados da aplicação ExtPDFtoTXT com dois *Executor´s*.

Na Tabela 5 apresentam-se os resultados temporais dos três ficheiros com três *Executor´s*.

	Capacidade de Processamento = 7,813GHz					
	Documentos					
	229KB		463KB		3,6MB	
	Preparação	Paralelização	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:04,085s	00m:08,372s	00m:04,095s	00m:08,281s	00m:05,127s	00m:14,811s
2ª Execução	00m:03,344s	00m:05,427s	00m:03,815s	00m:07,140s	00m:04,536s	00m:10,965s
3ª Execução	00m:03,424s	00m:04,917s	00m:03,645s	00m:08,071s	00m:04,236s	00m:11,746s
4ª Execução	00m:03,134s	00m:05,628s	00m:03,595s	00m:07,050s	00m:05,097s	00m:10,545s
5ª Execução	00m:03,404s	00m:05,227s	00m:04,236s	00m:07,030s	00m:04,125s	00m:11,837s

Tabela 5 - Resultados da aplicação ExtPDFtoTXT com três *Executor´s*.

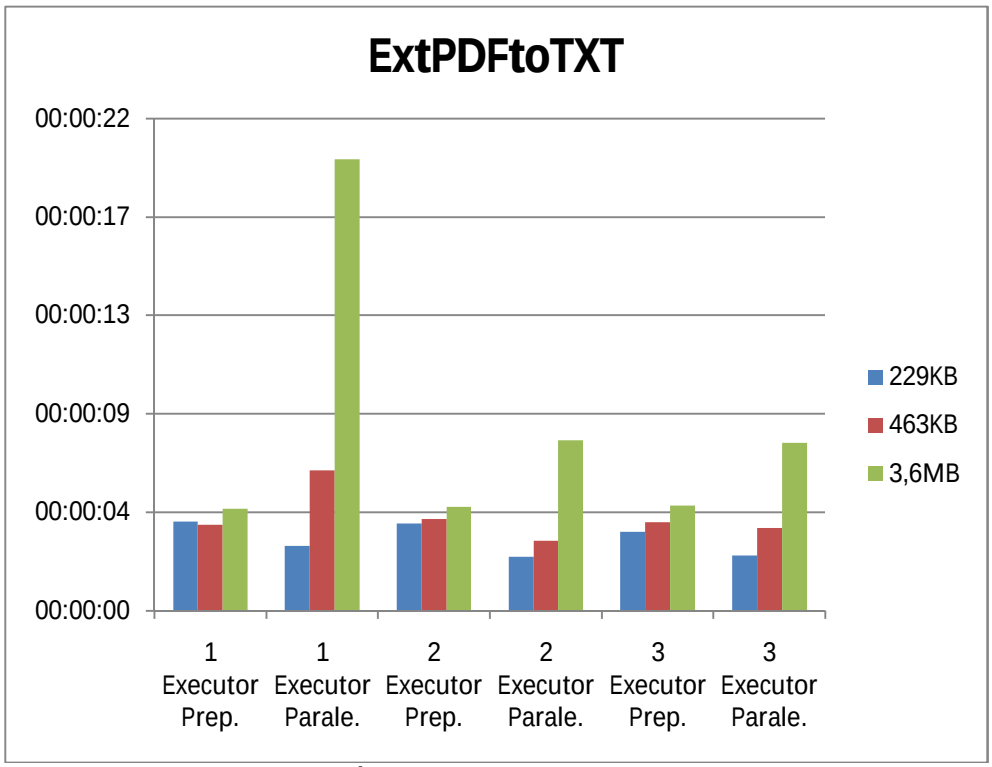

Figura 13 - Gráfico dos resultados obtidos ExtPDFtoTXT.

Pode-se constatar através da Figura 13 que o tempo necessário para processar os diferentes ficheiros diminui à medida que o número de *executer´s* aumenta, sendo esta melhoria mais evidente no ficheiro maior, ou seja, com mais páginas para serem processadas.

5.2 FrameRemover

Depois de lançado o *Web Service* surgem três funções (Figura 14) para serem invocadas:

- SetExecutionFrameRemover – usada para submeter o ficheiro ZIP contendo as imagens a serem tratadas.
- GetExecutionFrameRemover – devolve o ficheiro Zip com as imagens tratadas na execução;
- MySetExecutionFrameRemover – função de teste para ficheiros locais;

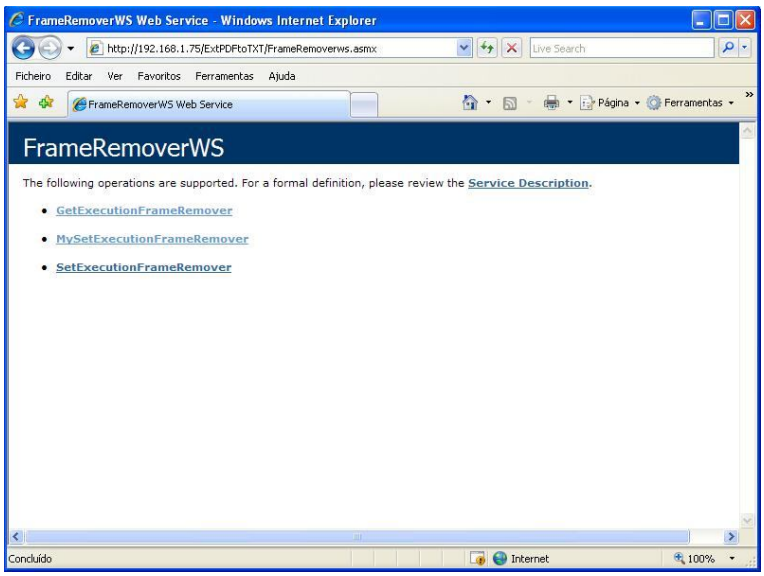

Figura 14 - *Web* Service FrameRemover.

Seleccionada a função MySetFileExecutionFrameRemover será necessário introduzir o caminho (*path*) até ao ficheiro ZIP a ser processado (Figura 15).

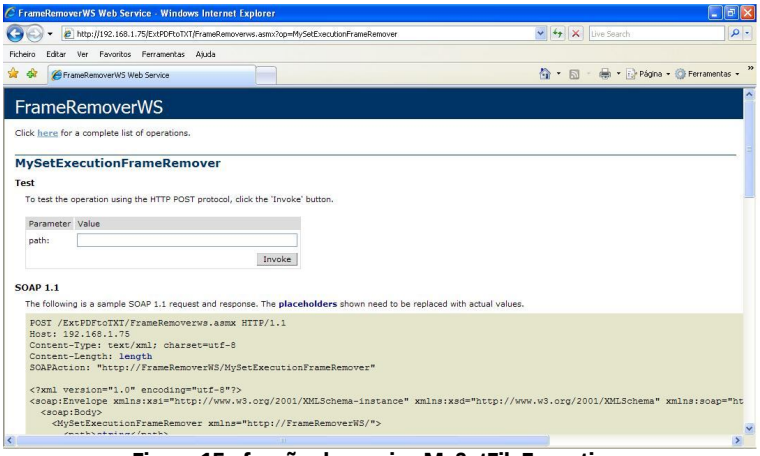

Figura 15 - função do serviço MySetFileExecuting.

Na Tabela 6 apresentam-se os resultados temporais dos três ficheiros com apenas um *Executor*.

	Capacidade de Processamento = 2,235 GHz			
	Documentos			
	3,25MB		**7,65MB**	
	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:22,692s	04m:23,428s	00m:26,227s	10m:52,388s
2ª Execução	00m:23,223s	04m:22,677s	00m:23,944s	10m:49,153s
3ª Execução	00m:22,842s	03m:59,924s	00m:24,575s	11m:44,352s
4ª Execução	00m:22,181s	04m:05,623s	00m:24,615s	11m:05,807s
5ª Execução	00m:22,862s	04m:09,819s	00m:23,423s	10m:37,266s

Tabela 6 - Resultados da aplicação FrameRemover com um *Executor*.

Na Tabela 7 apresentam-se os resultados temporais dos três ficheiros com dois *Executor´s*.

	Capacidade de Processamento = 5,402 GHz			
	Documentos			
	3,25MB		**7,65MB**	
	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:16,493s	01m:33,935s	00m:11,866s	03m:27,666s
2ª Execução	00m:20,439s	01m:36,689s	00m:12,557s	03m:12,593s
3ª Execução	00m:14,200s	01m:38,411s	00m:12,668s	03m:13,275s
4ª Execução	00m:13,779s	01m:36,488s	00m:12,097s	03m:03,261s
5ª Execução	00m:16,874s	01m:36,689s	00m:12,237s	03m:23,490s

Tabela 7 - Resultados da aplicação FrameRemover com dois *Executor´s*.

Na Tabela 8 apresentam-se os resultados temporais dos três ficheiros com três *Executor´s*.

	Capacidade de Processamento = 7,813GHz			
	Documentos			
	3,25MB		7,65MB	
	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:11,246s	01m:11,002s	00m:13,208s	02m:03,702s
2ª Execução	00m:10,925s	01m:12,724s	00m:12,767s	02m:04,955s
3ª Execução	00m:10,835s	01m:10,060s	00m:12,757s	02m:06,548s
4ª Execução	00m:10,795s	01m:08,067s	00m:12,757s	02m:05,458s
5ª Execução	00m:11,166s	01m:08,298s	00m:13,529s	02m:01,821s

Tabela 8 - Resultados da aplicação FrameRemover com três *Executor´s*.

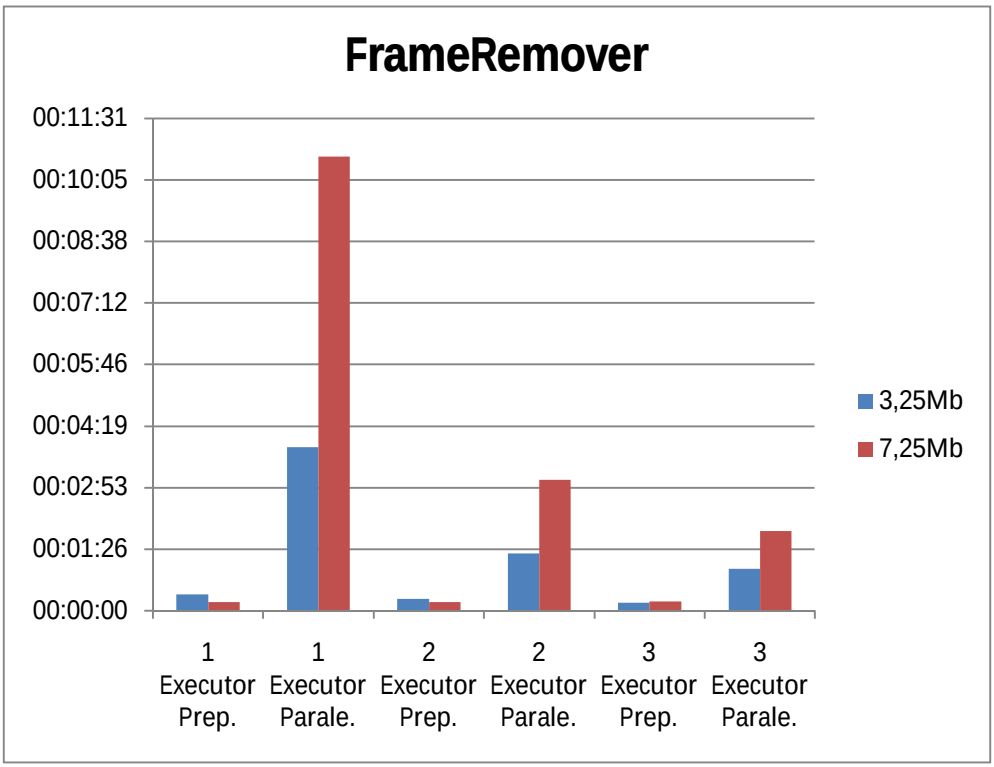

Figura 16 - Gráfico dos resultados obtidos FrameRemover.

Verifica-se na Figura 16 que o tempo necessário para processar os dois ficheiros ZIP diminui à medida que o número *executer´s* aumenta, sendo esta melhoria mais visível no ficheiro maior, ou seja, com mais imagens para serem processadas.

5.3 ImageConverter

Depois de lançado o *Web Service* surgem três funções (Figura 17) para serem invocadas:

- SetExecutionImageConverter – usada para submeter o ficheiro ZIP contendo as imagens a serem tratadas.
- GetExecutionImageConverter – devolve o ficheiro ZIP com as imagens tratadas na execução;
- MySetExecutionImageConverter – função de teste para ficheiros locais;

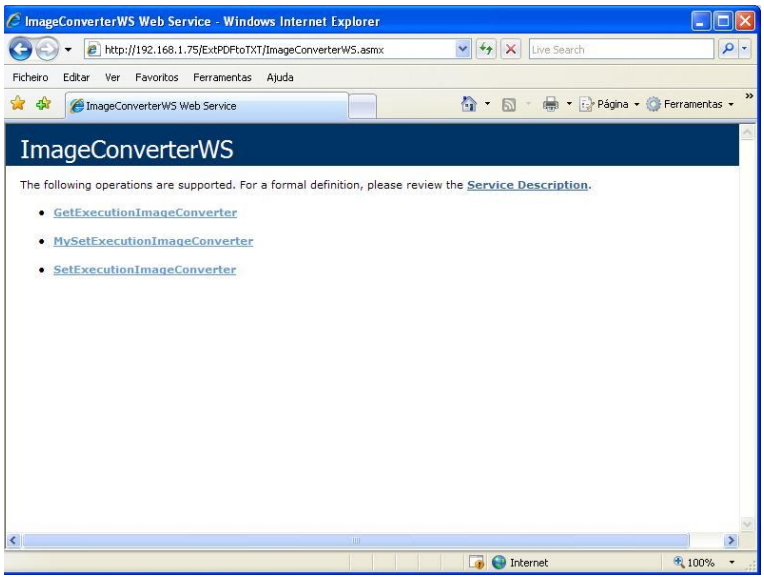

Figura 17 - *Web* Service ImageConverterWS.

Seleccionada a função MySetFileExecutionImageConverter será necessário introduzir o caminho (*path*) até ao ficheiro ZIP a ser processado, a resolução horizontal e vertical, assim como, a extensão para que se pretende converter (Figura 18).

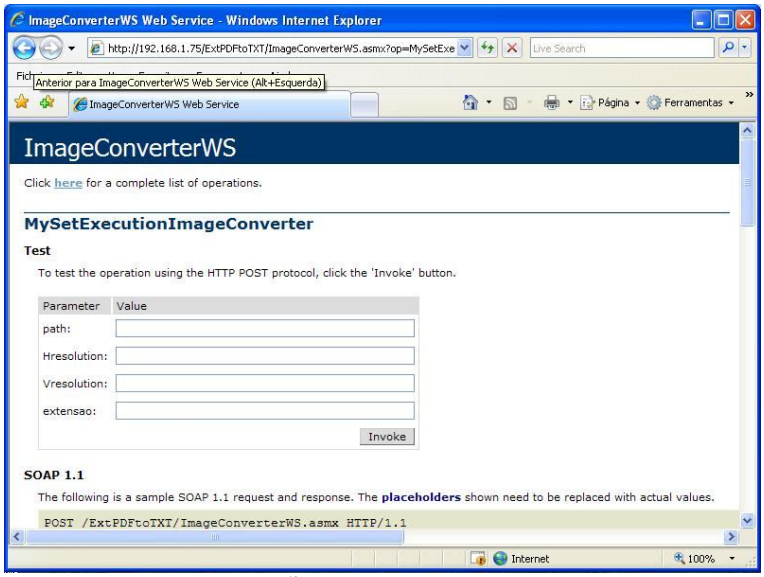

Figura 18 - função do serviço MySetFileExecuting.

Na Tabela 9 apresentam-se os resultados temporais dos três ficheiros com apenas um *Executor*.

	Capacidade de Processamento = 2,402 GHz					
	Documentos					
	229KB		**7,25MB**		**14,4MB**	
	Preparação	Paralelização	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:02,483s	00m:09,063s	00m:04,826s	00m:18,366s	00m:10,424s	00m:37,984s
2ª Execução	00m:02,673s	00m:09,093s	00m:04,576s	00m:17,795s	00m:08,432s	00m:35,240s
3ª Execução	00m:02,353s	00m:08,822s	00m:04,746s	00m:17,905s	00m:12,367s	00m:41,709s
4ª Execução	00m:02,603s	00m:08,972s	00m:04,306s	00m:18,186s	00m:10,615s	00m:37,574s
5ª Execução	00m:02,323s	00m:08,772s	00m:04,426s	00m:18,636s	00m:08,442s	00m:34,659s

Tabela 9 - Resultados da aplicação ImageConverter com um *Executor*.

Na Tabela 10 apresentam-se os resultados temporais dos três ficheiros com dois *Executor´s*.

	Capacidade de Processamento = 5,402 GHz					
	Documentos					
	229KB		**7,25MB**		**14,4MB**	
	Preparação	Paralelização	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:02,223s	00m:04,396s	00m:04,506s	00m:08,081s	00m:10,725s	00m:14,180s
2ª Execução	00m:02,473s	00m:03,885s	00m:04,796s	00m:07,310s	00m:11,156s	00m:13,769s
3ª Execução	00m:02,263s	00m:03,745s	00m:05,027s	00m:07,100s	00m:10,274s	00m:13,749s
4ª Execução	00m:02,253s	00m:04,005s	00m:17,014s	00m:07,180s	00m:07,400s	00m:13,138s
5ª Execução	00m:03,104s	00m:03,665s	00m:04,536s	00m:07,661s	00m:07,410s	00m:12,748s

Tabela 10 - Resultados da aplicação ImageConverter com dois *Executor´s*.

Na Tabela 11 apresentam-se os resultados temporais dos três ficheiros com três *Executor´s*.

	Capacidade de Processamento = 7,813GHz					
	Documentos					
	229KB		7,25MB		14,4MB	
	Preparação	Paralelização	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:02,874s	00m:04,396s	00m:05,017s	00m:08,081s	00m:08,732s	00m:14,180s
2ª Execução	00m:02,533s	00m:03,885s	00m:05,357s	00m:07,310s	00m:08,812s	00m:13,769s
3ª Execução	00m:02,483s	00m:03,745s	00m:05,057s	00m:07,100s	00m:07,811s	00m:13,749s
4ª Execução	00m:02,603s	00m:04,005s	00m:04,406s	00m:07,180s	00m:08,472s	00m:13,138s
5ª Execução	00m:02,313s	00m:03,665s	00m:04,786s	00m:07,661s	00m:08,542s	00m:12,748s

Tabela 11 - Resultados da aplicação ImageConverter com três *Executor´s*.

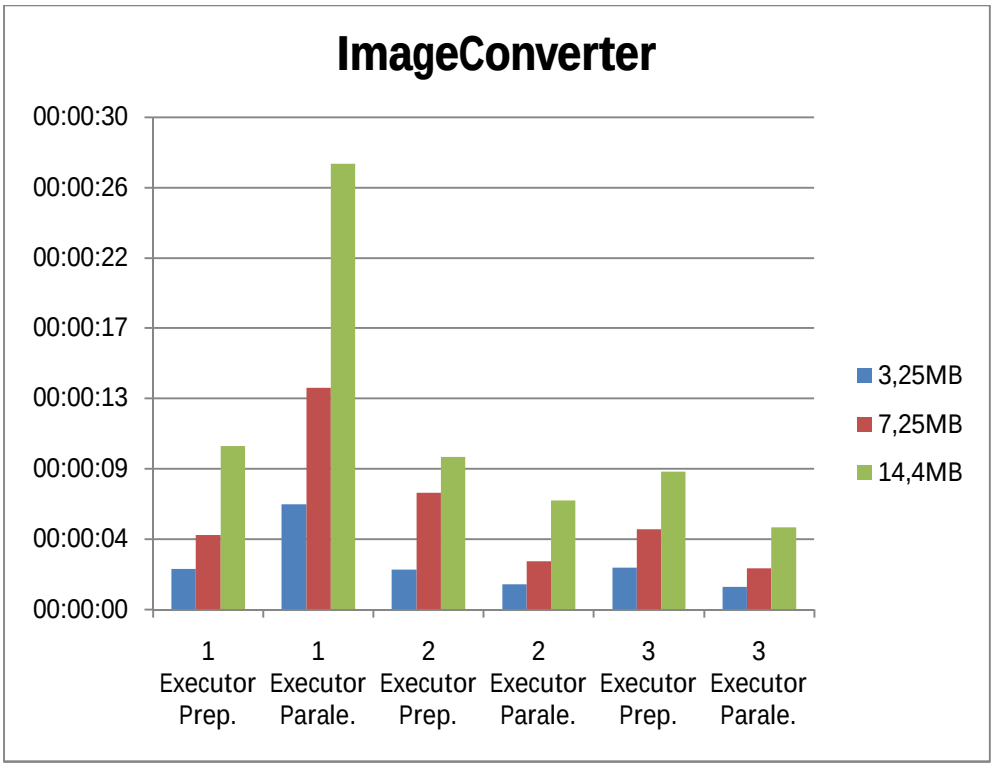

Figura 19 -Gráfico dos resultados obtidos ImageConverter.

Pode-se constatar através da Figura 19 que o tempo necessário para processar os diferentes ficheiros diminui à medida que o número de *executer´s* aumenta, sendo esta melhoria mais visível à medida que o tamanho dos ficheiros ZIP aumenta, ou seja, com mais páginas para serem processadas.

5.4 OCR

Depois de lançado o *Web Service* surgem quatro funções (Figura 20) para serem invocadas:

- GetExecution – devolve o texto das imagens processadas;
- MySetExecution – função de teste para ficheiros locais;
- ReturnTXTOfImage – função auxiliar do processo de execução;
- SetExecution – usada para submeter o ficheiro ZIP contendo as imagens a serem tratadas.

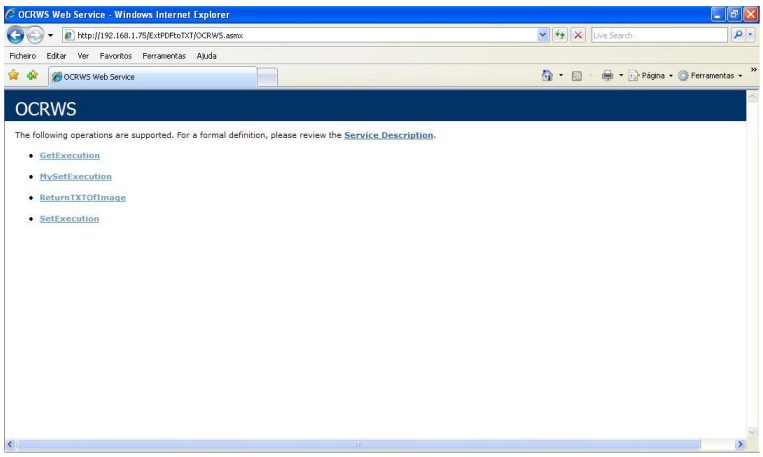

Figura 20 - *Web* Service OCRWS.

Seleccionada a função MySetFileExecution será necessário introduzir o caminho (*path*) até ao ficheiro ZIP a ser processado, assim como, a idioma do texto que se pretende extrair (Figura 21).

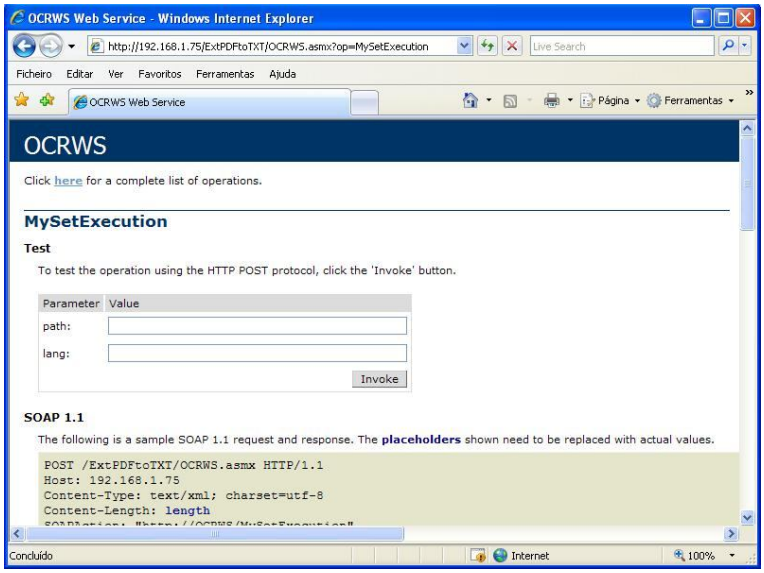

Figura 21 - função do serviço MySetFileExecuting.

Na Tabela 12 apresentam-se os resultados temporais dos três ficheiros com apenas um *Executor*.

	Capacidade de Processamento = 2,235 GHz			
	Documentos			
	3,25MB		7,65MB	
	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:22,692s	04m:23,428s	00m:26,227s	10m:52,388s
2ª Execução	00m:23,223s	04m:22,677s	00m:23,944s	10m:49,153s
3ª Execução	00m:22,842s	03m:59,924s	00m:24,575s	11m:44,352s
4ª Execução	00m:22,181s	04m:05,623s	00m:24,615s	11m:05,807s
5ª Execução	00m:22,862s	04m:09,819s	00m:23,423s	10m:37,266

Tabela 12 - Resultados da aplicação OCR com um *Executor*.

Na Tabela 13 apresentam-se os resultados temporais dos três ficheiros com dois *Executor´s*.

	Capacidade de Processamento = 5,402 GHz			
	Documentos			
	3,25MB		7,65MB	
	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:17,114s	01m:45,471s	00m:13,208s	03m:27,666s
2ª Execução	00m:14,981s	01m:40,530s	00m:12,767s	03m:12,593s
3ª Execução	00m:17,094s	01m:44,219s	00m:12,757s	03m:13,275s
4ª Execução	00m:18,088s	01m:43,869s	00m:12,757s	03m:03,261s
5ª Execução	00m:16,745s	01m:36,729s	00m:13,529s	03m:23,490s

Tabela 13 - Resultados da aplicação OCR com dois *Executor´s*.

Na Tabela 14 apresentam-se os resultados temporais dos três ficheiros com três *Executor´s*.

	Capacidade de Processamento = 7,813GHz			
	Documentos			
	3,25MB		7,65MB	
	Preparação	Paralelização	Preparação	Paralelização
1ª Execução	00m:17,114s	01m:42,467s	00m:13,208s	02m:03,702s
2ª Execução	00m:14,981s	01m:45,231s	00m:12,767s	02m:04,955s
3ª Execução	00m:17,094s	01m:30,590s	00m:12,757s	02m:06,548s
4ª Execução	00m:18,088s	01m:37,735s	00m:12,757s	02m:05,458s
5ª Execução	00m:16,745s	01m:39,978s	00m:13,529s	02m:01,821s

Tabela 14 - Resultados da aplicação OCR com três *Executor´s*.

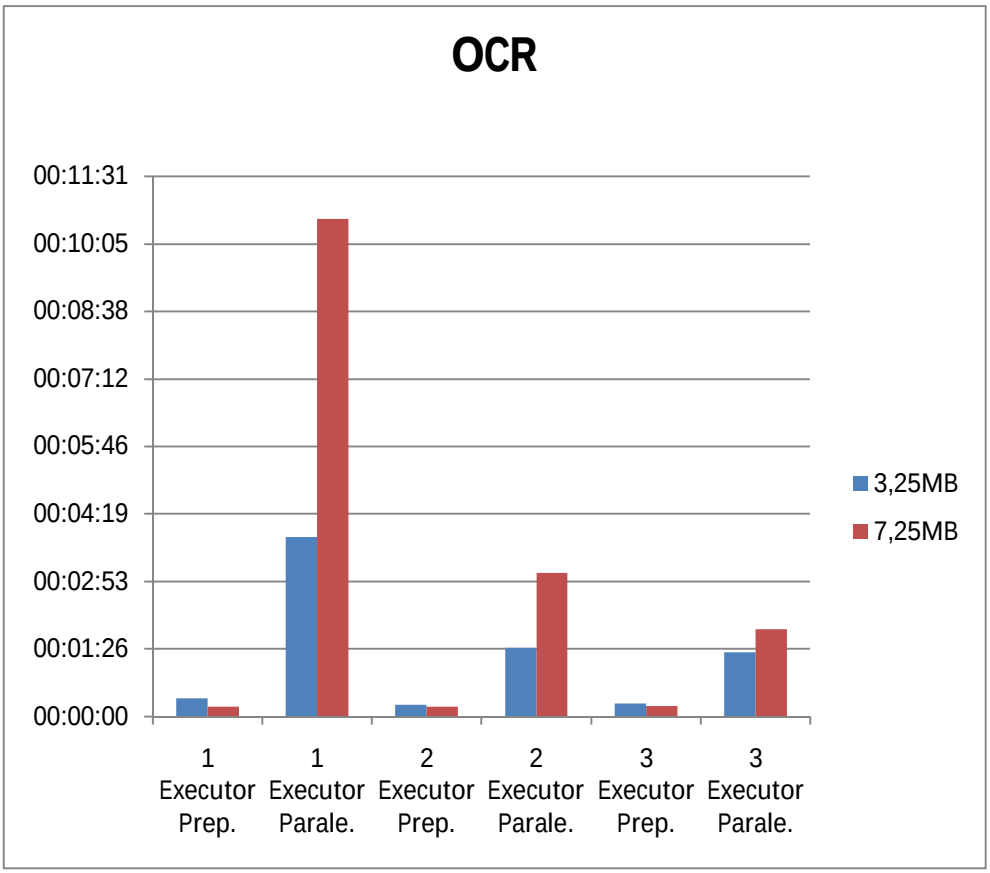

Figura 22 - Gráfico dos resultados obtidos OCR.

Através da Figura 22 constata-se que o tempo necessário para processar os dois ficheiros ZIP diminui à medida que o número *executer´s* aumenta, sendo esta melhoria evidente no ficheiro maior, ou seja, com mais imagens para serem processadas.

6 Conclusões e Trabalho Futuro

6.1 Conclusões

Quanto aos resultados obtidos nos *Web Services* desenvolvidos, verifica-se uma grande melhoria da prestação da aplicação de um para dois *Executer´s*. Já as melhorias relativas à introdução de um *Executor* a um cenário que já continha dois *Executor´s*, não são assim tão significativas, isto deve-se ao facto de nos testes efectuados, terem no cenário de três *Executer´s* o *Manager* e um dos *Executer´s* se encontrarem na mesma máquina. Se fizermos uma análise relativa às melhorias da prestação comparativamente ao aumento do tamanho do ficheiro, conclui-se que ao adicionar *Executor´s* na *Grid*, as melhorias são mais significativas para ficheiros com mais páginas.

A nível de programação utilizada o Alchemi é basicamente uma ferramenta bastante simples e de fácil compreensão para o programador, já que todo motor de *threads* e gestão destas é feita através deste.

No Alchemi os *Executor's* são executados sem contexto, logo no caso do serviço ExtPDFtoTXT as livrarias utilizadas para serem executadas têm de ser instaladas no *Assembly* de cada máquina dos *Executor's*.

Para além destas limitações o Alchemi tem um poder de execução bastante satisfatório. O que no exemplo do ImageConverter permite obter resultados na tecnologia *Grid* bastante bons. Embora nos restantes também podemos tirar esta conclusão.

Com este trabalho pretendia-se a criação de um conjunto de serviços *Web*, desenvolvidos com tecnologia *Grid*, para aplicação nas Bibliotecas Digitais. Uma vez que os resultados obtidos foram bastante interessantes do ponto de vista do tempo de execução, pode-se dizer então que a tecnologia *Grid* pode ser muito vantajosa para as Bibliotecas Digitais.

6.2 Trabalho Futuro

Como trabalho futuro, relativamente ao FrameRemover, poderão ser desenvolvidos métodos para obter melhor tolerância de cor em relação à moldura envolvente, isto porque o FrameRemover remove uma moldura de qualquer cor, até ao comprimento maior de igualdades de pixéis. Portanto, imaginando que a esquerda começa com cor (255,255,255) completamente branco e pelo meio encontra (254,254,254) um branco alterado, o que para um olho humano é igualmente branco, assim acrescentar-se-ia um método para induzir tolerância de modo a colmatar esta possível falha, passando a ter mais um parâmetro.

Relativamente ao ImageConverter poder-se-ia aplicar em imagens grandes, o método de *crop,* que consiste em transformar uma imagem grande em várias mais pequenas e poder obter ganhos de paralelização com uma única Imagem.

Relativamente ExtPDFtoTXT foram utilizadas bibliotecas que necessitaram de ser instaladas em cada máquina dos *Executor´s*, para colmatar essa limitação futuramente poder-se-ia encontrar uma solução para que isto não sucedesse, o que poderia passar pela construção de uma dll que se instalasse em cada *Executor*.

Relativamente ao OCR a tecnologia testada utilizada era *Open Source* e o texto que é reconhecido não é 100% perceptível, nesta questão poder-se-ia ter utilizado outras tecnologias melhores e mais precisas que não foi possível devido a limitação de uso *open source*.

Em relação ao *Web Service*, foi usada uma tecnologia básica de comunicação, em trabalho futuro podia-se usar as novas tecnologias de *Web Services,* tais como o *Comunication Foundation* da .NET *framework* 3.5. Esta permitiria uma comunicação mais segura, no que diz respeito à autenticação e encriptação dos dados trocados.

7 Bibliografia

[1] AccessGrid. (s.d.). Obtido de http://www.accessgrid.org/

[2] Alchemi. (s.d.). Obtido de http://www.alchemi.net/

[3] B.Sotomayor. (s.d.). Obtido em Abril de 2008, de The globus toolkit programmers tutorial: http://gdp.globus.org/gt4-tutorial/

[4] B.Sotomayor, L. (2005). *Globus Toolkit 4: Programming Java Services.* Morgan Kaufmann Publishers.

[5] Berstis, V. (s.d.). *Fundamentals of grid computing.* Obtido em Março de 2008, de http://www.redbooks.ibm.com/redpapers/pdfs/redp3613.pdf

[6] Bombonato, F. (s.d.). *Computação em grid. uma introdução.* Obtido em Março de 2008, de http://www.geleira.org/pdf/grid_computing.pdf

[7] ChinaGrid. (s.d.). *China National Grid Project*. Obtido de http://i.cs.hku.hk/~clwang/grid/CNGrid.html

[8] DEISA. (s.d.). Obtido de http://www.deisa.eu/

[9] *DICE*. (s.d.). Obtido de http://diceresearch.org/DICE_Site/Home/Home.html

[10] eDiaMoND. (s.d.). Obtido de http://www.ediamond.ox.ac.uk/

[11] EGEE. (s.d.). Obtido de http://www.eu-egee.org/

[12] *FAFNER*. (s.d.). Obtido de http://www.npac.syr.edu/factoring.html

[13] *Global Grid Forum*. (s.d.). Obtido de http://www.gridforum.org/

[14] *Globus.* (s.d.). Obtido em Fevereiro de 2008, de Globus toolkit: http://www.globus.org/

[15] H.Kishimoto, J. (2005). Defining the Grid: Roadmap for OGSA Standards . *OGSA working group* .

[16] H.M.Gladney, E. Z. (1994). Digital library: gross structure and requirements. *Conference Digital Libraries.* Texas.

[17] I. Foster C. Kesselman, S. T. (s.d.). The anatomy of the grid: Enabling scalable virtual.

[18] I. Foster, C. K. (July 1998). *"The Grid: Blueprint for a New Computing Infrastructure".* Morgan Kaufmann.

[19] I. Foster, C. K. (2002, January). The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems.

[20] I.Foster, J. W. "Software Infrastructure for the I-WAY High Performance Distributed Computing Experiment". *5th IEEE Symposium on High Performance Distributed Computing*, (pp. 562-571).

[21] I.Foster, K. D. (s.d.). *The WS-Resource Framework. Version 1.0*. Obtido em Março de 2008, de http://www-106.ibm.com/developerworks/library/ws-resource/ws-wsrf.pdf

[22] iRODS. (s.d.). Obtido de www.irods.org

[23] Leiner, B. (1998). Metrics and digital libraries. *Lib Magazine* .

[24] M.Nash. (s.d.). *Oracle 10g: Infrastructure for grid computing.* Obtido em Março de 2008,de http://otn.oracle.com/tech/grid/collateral/GridTechWhitePaper_final.pdf

[25] Miley, M. (2003). The grid: Bringing computing power to the masses. *Stokie: Oracle Magazine* , pp. pag. 39-44.

[26] Myerson, J. M. (s.d.). *Cloud computing versus grid computing.* Obtido de IBM: http://www.ibm.com/developerworks/*Web*/library/wa-cloudgrid/

[27] OASIS. (s.d.). *Oasis Web service resource framework*. Obtido em Abril de 2008, de http://www.oasisopen.org/home/index.php

[28] OASIS Standard. *Web* services resource 1.2 (ws-resource). Organization for the Advancement of Structured Information Standards.).

[29] OGSA. (s.d.). Obtido de http://www.globus.org/ogsa/

[30] PDFBox. (s.d.). Obtido de http://www.pdfbox.org

[31] *Search for extraterrestrial intelligence*. (s.d.). Obtido em Março de 2008, de SETI@home: http://setiathome.berkeley.edu/

[32] Tanenbaum, A. S. *Distributed Systems: Principles and Paradigms.*

[33] TeraGrid. (s.d.). Obtido de http://www.teragrid.org/

8 Lista de acrónimos

ACL	Access Control List
API	Application Programming Interface
CERN	Conseil Européen pour lá Recherche Nucléaire
DEISA	Distributed European Infrastructure for Supercomputing Applications
DICE	Data Intenssive Cyber Enviromments
DNS	Domain Name System
EDG	European Data Grid's
EGEE	Enabling Grids for E-sciencE in Europe
FAFNER	Factoring via Network Enabled Recursion
FCFS	First Come First Served
GGF	Global Grid Forum
GSI	Grid Security Infrastructure
GT4	Globus Toolkit
GUID	Globally unique identifier
IBM	International Business Machines
ICMP	Internet Control Message Protocol
IEETA	Instituto de Engenharia Electrónica e Telemática
IP	Internet Protocol
iRODS	Integrated Rules-Oriented Data Systems
I-WAY	Information Wide Area Year
LHC	Large Hadron Collider
LIP	Laboratório de Instrumentação e Física Experimental de Partículas
MDS	Monitoring and Discovering Services
OASIS	Organization for the Advancement of Structured Information Standards
OCR	Optical Character Recognition
OGSA	Open Grid Services Architecture
OSPF	Open Shortest Path First
OV	Organização Virtual
PDF	Portable Document Format
PM	Package Manager
SDK	Software Development Kit
SETI@home	Search for Extraterrestrial Intelligence
SinBAD	Sistema Integrado para Bibliotecas e Arquivos Digitais
SOA	Service Oriented Architecture
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TIFF	Tagged Image File Format
UDP	User Datagram Protocol
WMS	Workload Management System
WSI	*Web* Services Interoperability
WSRF	*Web* Services Resource Framework
XML	Extensible Markup Language

9 Apêndice

9.1 Instalar o Alchemi

Para a implantação do sistema *Grid* era necessário:

- Microsoft .NET *framework* 1.1.
- SQL Server 2000 ou MSDE 2000.

Assim, o primeiro passo foi instalar o Alchemi numa das máquinas, em que para isso é necessário ter uma base dados. Concluída esta parte procedimentos configurou-se o *Manager* como uma aplicação normal de *Windows Desktop*.

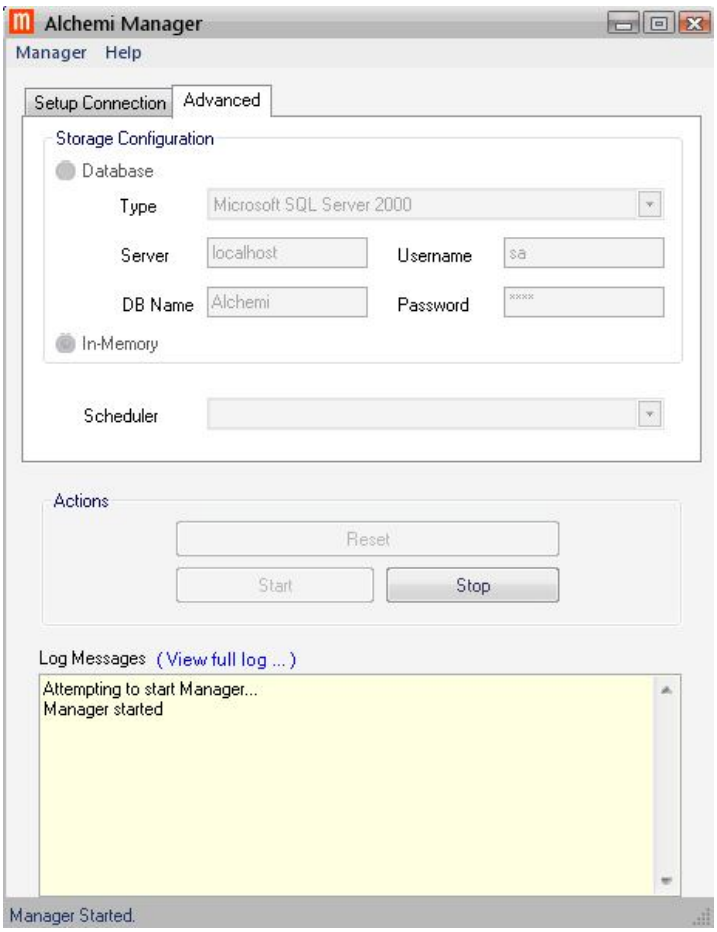

Figura 23 - Alchemi *Manager.*

Para iniciar a execução do *Manager* basta pressionar o botão de *Start*, Figura 23. De modo a proporcionar uma ferramenta para consulta de eventuais problemas na normal execução o *Manager* guarda os seus erros e produtos num ficheiro chamado "alchemi-manager.log".

Ao instalar o *Manager*, são criadas por defeito, três contas, cada uma com um *login* e uma *palavra-chave*, essas contas são: *executor*, *user* e *admin*, que pertencem, respectivamente, ao grupo dos '*Executor´s'*, '*Users'* e '*Administrators'*.

Após verificar que o *Manager* se encontrava a funcionar correctamente, instalou-se o *Executor* nas três máquinas que iriam ser usadas neste projecto, assim como no *Manager* foram configurados como uma aplicação normal de ambiente Windows.

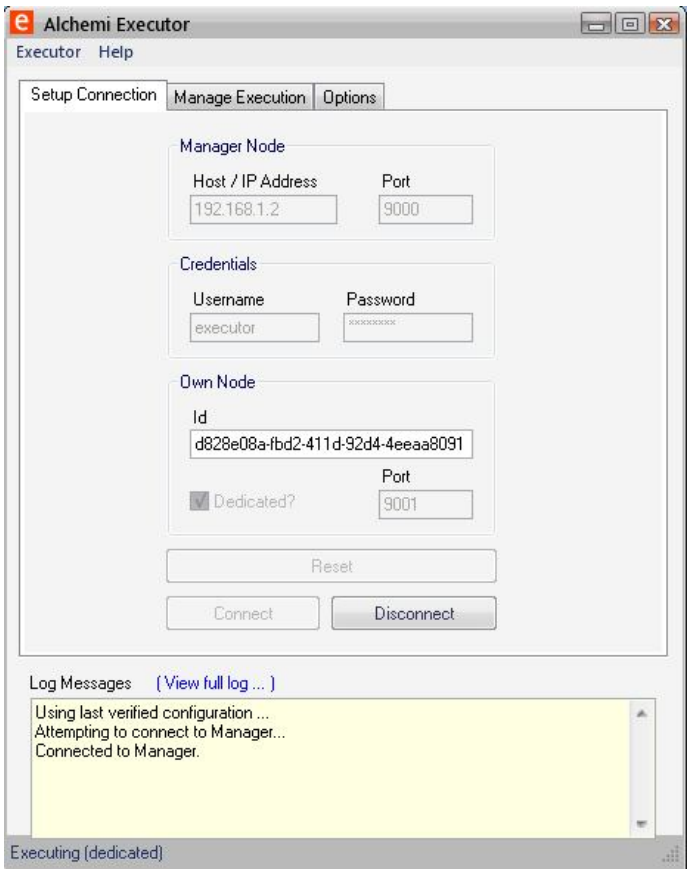

Figura 24 - Alchemi *Executor.*

Nos *Executor´s* foi necessário configurar também:

- O endereço e porto do *Manager*, ao qual o *Executor* se vai ligar.
- Se a execução vai ser dedicada ou não-dedicada. Foi escolhida a execução dedicada, desta maneira, o executor está sempre a executar *threads*.

Para iniciar a execução dos *Executor´s* basta pressionar o botão de *Start* da Figura 24.

Assim, como no *Manager,* os erros e resultados do *Executor* são registados num ficheiro chamado "alchemi-executor.log".

Com o sistema *Grid* criado testaram-se as suas funcionalidades, através do *Software Development Kit* (SDK), fornecido pelo Alchemi. Este é constituído por:

- Alchemi *Console*
- Alchemi.Core.dll
- Exemplos

Alchemi Console

A Alchemi *Console* é uma ferramenta *Grid* de administração e controlo. Esta fornece informação acerca das características do sistema tais como, a capacidade que está a ser utilizada, o número de *Executor´s* e as *threads* que ainda não foram executadas, recorrendo a gráficos e tabelas para visualizar a informação pretendida, Figura 25.

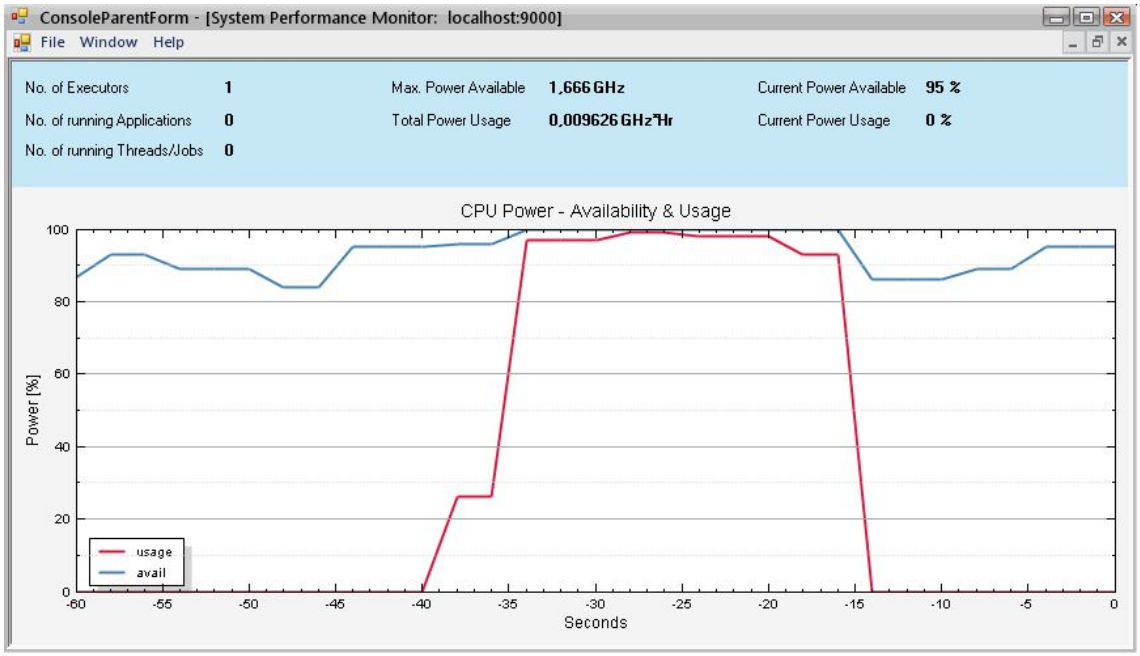

Figura 25 - Alchemi *Console*.

Alchemi.Core.dll

A Alchemi.Core.dll é uma biblioteca, disponibilizada pelo Alchemi, para criar aplicações *Grid*.

Exemplos

Os exemplos, consistem em pequenas aplicações *Grid*, ideais para testar e verificar o funcionamento da *Grid*, assim como através do *tuturial* é possível perceber como se deve programar para implementar uma aplicação *Grid* para ser executada no Alchemi.

9.2 Código das aplicações desenvolvidas

9.2.1 ExtPDFtoTXT

```
using System;
using System.Text;
using org.pdfbox.util;
using org.pdfbox.pdmodel;
using System.IO;
using Alchemi.Core.Owner;
using java.util;

namespace ExtPDFtoTXT
{
    [Serializable]
    public class PDFToSendGT : GThread
    {
        public static GApplication App = new GApplication();

        private int pageNumber;

        private String result;
        private string WSURL;

        public static string FileDestination;

        public static string FileDestinationPath;

        private byte[] doc;

        private DateTime di;
        private static DateTime dtInit = DateTime.Now;

        public String getResult()
        {
            return this.result;
        }

        public PDFToSendGT()
        {
        }

        public  DateTime getdi()
        {
            return this.di;
        }

        public PDFToSendGT(string filepath)
        {
            FileInfo f = new FileInfo(filepath);
            FileDestination = f.Name;
        }
        public PDFToSendGT(int page, PDDocument document)
        {
            this.pageNumber = page;
            this.doc = PDF2Array.savePdfToByte(document);
        }
```

```
        public PDFToSendGT(int page, PDDocument document, string
filepath,DateTime di)
        {
            //FileInfo f = new FileInfo(filepath);
            FileDestination = filepath;

            this.pageNumber = page;
            this.doc = PDF2Array.savePdfToByte(document);
            this.di = di;
        }

        public PDFToSendGT(int page, PDDocument document, string
filepath, DateTime di, string WSURL)
        {

            if (WSURL.Length == 0)
            {
                this.WSURL =
"http://localhost/ExtPDFtoTXT/ExtPDFtoTXTWS.asmx";
            }
            else
            {
                this.WSURL = WSURL;
            }

            //FileInfo f = new FileInfo(filepath);
            FileDestination = filepath;

            this.pageNumber = page;
            this.doc = PDF2Array.savePdfToByte(document);
            this.di = di;
        }

        public override void Start()
        {

            ExtPDFtoTXTWS.ExtPDFtoTXTWS exppdftxtWS = new
ExtPDFtoTXT.ExtPDFtoTXTWS.ExtPDFtoTXTWS();
            exppdftxtWS.Url = this.WSURL;
            // coloca o resultado de leitura da pagina da currente thread
            this.result = exppdftxtWS.ReturnTxtOFByte(this.doc);
        }

        private static void Init()
        {
            Init("localhost");
        }
        private static void Init(string managerIP)
        {
            App.Connection = new GConnection(managerIP, 9000, "user",
"user");
            // grid thread needs to

            if (!App.Manifest.Contains(new
ModuleDependency(typeof(PDFToSendGT).Module)))
            {
                App.Manifest.Add(new
ModuleDependency(typeof(PDFToSendGT).Module));
```

```
            }

            App.ThreadFinish += new GThreadFinish(App_ThreadFinish);

            App.ApplicationFinish += new
GApplicationFinish(App_ApplicationFinish);

        }

        static void App_ApplicationFinish()
        {
            //apos finalizada as threads
            TextWriter twl = new StreamWriter(FileDestination +
"_LOG.txt", true);
            TextWriter tw = new StreamWriter(FileDestination +
"_Result.txt", true);
            foreach (GThread f in App.Threads)
            {
                //percorre as threads para guardar no ficheiro de
resultado o texto de cada pagina
                PDFToSendGT pdf = (PDFToSendGT)f;
                tw.WriteLine("Pagina "+ pdf.getPageNumber().ToString());
                tw.WriteLine(pdf.getResult());
            }
            twl.WriteLine();

            TimeSpan ti = DateTime.Now.Subtract(dtInit);

            twl.WriteLine(" FINISH " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff") + " Com o tempo de (s) " + ti.Hours.ToString().PadLeft(2,
'0') + ":" + ti.Minutes.ToString().PadLeft(2, '0') + ":" +
ti.Seconds.ToString().PadLeft(2, '0') + "," +
ti.Milliseconds.ToString().PadLeft(3, '0'));
            twl.Close();
            tw.Close();
            App.Stop();

        }

        private static void App_ThreadFinish(GThread thread)
        {

            PDFToSendGT pdf = (PDFToSendGT)thread;

        TextWriter twl = new StreamWriter(FileDestination + "_LOG.txt" ,
true);

            TimeSpan ti = DateTime.Now.Subtract(pdf.di);

            twl.WriteLine("Page " + pdf.getPageNumber().ToString() + "-"
+ DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss:fff") + " Com o tempo de (s)
" + ti.Hours.ToString().PadLeft(2, '0') + ":" +
ti.Minutes.ToString().PadLeft(2, '0') + ":" +
ti.Seconds.ToString().PadLeft(2, '0') + "," +
ti.Milliseconds.ToString().PadLeft(3, '0'));
            twl.Close();
        }
        private long id;
```

```
        private List pdfToSend;

        public PDFToSendGT(long id,PDDocument document)
        {
            Splitter splitter = new Splitter();
            this.setId(id);
            this.doc = PDF2Array.savePdfToByte(document);
            this.pageNumber = Convert.ToInt32(id);
        }

        public void setId(long id)
        {
            this.id = id;
         }

        public long getId()
        {
            return this.id;
        }

        public void setPageNumber(int pageNumber)
        {
            this.pageNumber = pageNumber;
        }

        public int getPageNumber()
        {
            return this.pageNumber;
        }

        public void setPdfToSend(List ppd)
        {
            this.pdfToSend = ppd;
        }

        public List getPdfToSend()
        {
            return this.pdfToSend;
        }

        public void AppStart(byte[] file, string guidExection, string
directoryPath, string WSURL, string managerIP)
        {
            //inicia o documento pdf para percorrer as paginas
           PDDocument doc = PDF2Array.loadPdfFromByte(file);
           FileDestinationPath = directoryPath;
           FileDestination = directoryPath + guidExection ;
           TextWriter twl = new StreamWriter(FileDestination +
"_LOG.txt");

           twl.WriteLine("INICIO  " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff"));
           twl.Close();
           PDFTextStripper pdfStripper = new PDFTextStripper();
           Splitter teste = new Splitter();
            List testLisT = teste.split(doc);
            for (int i = 0; i < testLisT.size(); i++)
            {
                //para cada pagina inicia uma thread de leitura de texto
```

```
                App.Threads.Add(new PDFToSendGT(i + 1,
(PDDocument)testLisT.get(i), FileDestination, DateTime.Now, WSURL));
            }
            //inicia as settings do alchemi manager
            Init(managerIP);
            //inicia o processo das threads
            App.Start();
        }
    }
}
```

9.2.2 ExtPDFtoTXTWS

```
using System;
using System.Collections.Generic;
using System.Web;
using System.Web.Services;
using org.pdfbox.util;
using ExtPDFtoTXT;
using java.lang;
using org.pdfbox.pdmodel;
using System.IO;
using System.Web.Hosting;

namespace ExtPDFtoTXTWS
{
    /// <summary>
    /// Summary description for Service1
    /// </summary>
    [WebService(Namespace = "http://ExtPDFtoTXTWS/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    public class ExtPDFtoTXTWS : System.Web.Services.WebService
    {

        [WebMethod]
        public string SetFileExecuting(byte[] pdfFile)
        {
            //inicia a chave de execucao
            string guidExecuting = System.Guid.NewGuid().ToString();
            DateTime di = DateTime.Now;
            PDFToSendGT p = new PDFToSendGT();
            //pasta de execucao
            string directoryPath =
HostingEnvironment.ApplicationPhysicalPath+ @"PDFtoTXT\";
            //inicia processo
            p.AppStart(pdfFile, guidExecuting, directoryPath,
this.Context.Request.Url.ToString().Substring(0,
this.Context.Request.Url.ToString().LastIndexOf('/')),
UtilsWS.GetManagerIP());

            return guidExecuting;
        }
        [WebMethod]
        public string MySetFileExecuting(string pdfFile)
        {

            return SetFileExecuting(File.ReadAllBytes(pdfFile));
```

```
        }

        [WebMethod]
        public string ReturnTxtResultOfGuid(string guid)
        {
            string Text = "False";

            if (File.Exists(HostingEnvironment.ApplicationPhysicalPath +
@"PDFtoTXT\" + guid+"_result.txt"))
            {
                //caso exista o ficheiro de resultado da chave de
execucao e devolvido o texto deste
                Text =
File.ReadAllText(HostingEnvironment.ApplicationPhysicalPath +
@"PDFtoTXT\" + guid + "_result.txt");
            }
            return Text;
        }

        [WebMethod]
        public string ReturnTxtOFByte(byte[] doc)
        {
            PDFTextStripper pdfStripper = new PDFTextStripper();
            return pdfStripper.getText(PDF2Array.loadPdfFromByte(doc));
        }
    }
}
```

9.2.3 FrameRemover

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Drawing;
using Alchemi.Core.Owner;
using System.IO;

namespace ExtPDFtoTXT
{

    [Serializable]
    public class FrameRemoverGT : GThread
    {
        public static GApplication App = new GApplication();

        private Bitmap image;

        public Image frameless;

        public Image getFrameless()
        {
            return this.frameless;
        }

        public String where;

        public String getWhere()
```

```
        {
            return this.where;
        }
        public static string FilePathExecution;
        public static DateTime dtInitExecution;
        public static string guidExecution;

        private DateTime init;

        public FrameRemoverGT()
        {
         }

        public FrameRemoverGT(String img, String where,DateTime init)
        {
            this.image = new Bitmap(img);
            this.where = where;
            this.init = init;
        }

        public FrameRemoverGT(String img, String where, DateTime init,
string filePathExecution)
        {
            FilePathExecution = filePathExecution;
            this.image = new Bitmap(img);
            this.where = where;
            this.init = init;
        }

        public override void Start()
        {
            //faz o processo da currente thread

            int x = this.findX(image);
            int y = this.findY(image);

            int lastx = this.findXLast(image);
            int lasty = this.findYLast(image);
            //coloca a imagem final no resultado
            this.frameless = ((Image)this.remove(image, x, lastx, y,
lasty));
        }

        private static void Init()
        {
            Init("localhost");
        }
        private static void Init(string managerIP)
        {
            // specify connection properties

            App.Connection = new GConnection(managerIP, 9000, "user",
"user");

            if (!App.Manifest.Contains(new
ModuleDependency(typeof(FrameRemoverGT).Module)))
            {
                App.Manifest.Add(new
ModuleDependency(typeof(FrameRemoverGT).Module));
```

```
            }
            // subscribe to ThreadFinish event
            App.ThreadFinish += new GThreadFinish(App_ThreadFinish);
            App.ApplicationFinish += new
GApplicationFinish(App_ApplicationFinish);
        }

        static void App_ApplicationFinish()
        {
            //apos todas as threads terem finalizado
            //cria a pasta final da execucao
            if (!Directory.Exists(FilePathExecution + @"\FINAL"))
            {
                Directory.CreateDirectory(FilePathExecution + @"\FINAL");
            }
            foreach (GThread f in App.Threads)
            {
                //percorre as threads da aplicacao e guarda o resultado
na pasta final
                FrameRemoverGT fr = (FrameRemoverGT)f;
                fr.getFrameless().Save(fr.getWhere());
            }
            TextWriter twl = new StreamWriter(FilePathExecution +
@"\LOG.txt", true);
            TimeSpan ti = DateTime.Now.Subtract(dtInitExecution);

            twl.WriteLine(" FINISH " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff") + " Com o tempo de (s) " + ti.Hours.ToString().PadLeft(2,
'0') + ":" + ti.Minutes.ToString().PadLeft(2, '0') + ":" +
ti.Seconds.ToString().PadLeft(2, '0') + "," +
ti.Milliseconds.ToString().PadLeft(3, '0'));
            twl.Close();
            App.Stop();
        }

        private static void App_ThreadFinish(GThread thread)
        {
            FrameRemoverGT fr = (FrameRemoverGT)thread;
            TimeSpan ti = DateTime.Now.Subtract(fr.init);
            TextWriter twl = new StreamWriter( FilePathExecution +
@"\LOG.txt", true);
            twl.WriteLine("Imagem " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff") + " Com o tempo de (s) " + ti.Hours.ToString().PadLeft(2,
'0') + ":" + ti.Minutes.ToString().PadLeft(2, '0') + ":" +
ti.Seconds.ToString().PadLeft(2, '0') + "," +
ti.Milliseconds.ToString().PadLeft(3, '0'));
            twl.Close();

        }

        public Bitmap remove(Bitmap img, int xStart, int xEnd, int
yStart, int yEnd)
        {
            Bitmap returnValue = new Bitmap(xEnd-xStart+1,yEnd-yStart+1);
// criação da nova imagem ja com novos limites

            for (int i = xStart, xNew = 0; i < xEnd; i++, xNew++)
            {
                for (int j = yStart, yNew = 0; j < yEnd; j++, yNew++)
```

```
                {
                    returnValue.SetPixel(xNew, yNew, img.GetPixel(i, j));
// setting do pixel na coordenada xNew,yNew
                }
            }

            return returnValue;

        }

        public int findX(Bitmap img)
        {
            int returnValue = img.Height;

            Color first = new Color(),temp;

            for (int x = 0; x < img.Width; x++)
            {

                for (int y = 0; y < img.Height; y++)
                {
                    temp = img.GetPixel(x, y);

                    if(first.IsEmpty)
                    {
                        first = temp;
                    }
                    if (!first.Equals(temp) && returnValue > x)
                    {
                        returnValue = x - 1;
                        if (returnValue < 0)
                        {
                            returnValue = 0;
                        }
                    }
                }
            }


            return returnValue;
        }

        public int findY(Bitmap img)
        {
            int returnValue = img.Height;

            Color first = new Color(), temp;

            for (int x = 0; x < img.Width; x++)
            {

                for (int y = 0; y < img.Height; y++)
                {
                    temp = img.GetPixel(x, y);

                    if (first.IsEmpty)
                    {
                        first = temp;
                    }
```

```
                if (!first.Equals(temp) && returnValue > y)
                {
                    returnValue = y - 1;
                    if (returnValue < 0)
                    {
                        returnValue = 0;
                    }
                }
            }
        }
        return returnValue;
    }

    public int findXLast(Bitmap img)
    {
        int returnValue = 0;

        Color first = new Color(), temp;

        for (int x =img.Width-1 ; x >= 0; x--)
        {

            for (int y = img.Height-1 ; y >= 0; y--)
            {
                temp = img.GetPixel(x, y);

                if (first.IsEmpty)
                {
                    first = temp;
                }
                if (!first.Equals(temp) && returnValue < x)
                {
                    returnValue = x + 1;
                }
            }
        }
        return returnValue;
    }

    public int findYLast(Bitmap img)
    {
        int returnValue = 0;

        Color first = new Color(), temp;

        for (int x = img.Width-1; x >= 0; x--)
        {

            for (int y = img.Height-1 ; y >= 0 ; y--)
            {
                temp = img.GetPixel(x, y);

                if (first.IsEmpty)
                {
                    first = temp;
                }
                if (!first.Equals(temp) && returnValue < y)
                {
                    returnValue = y + 1;
```

```
                    }
                }
            }
            return returnValue;
        }

        public void AppStart(string directoryPath,string managerIP)
        {
            FilePathExecution = directoryPath;
            DateTime init = DateTime.Now;
            dtInitExecution = init;
            TextWriter twl = new StreamWriter(directoryPath
+@"\LOG.txt");
            twl.WriteLine("INICIO  " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff"));
            twl.Close();

                DirectoryInfo di = new
DirectoryInfo(directoryPath+@"\Source");
                FileInfo[] fis = di.GetFiles();

                foreach (FileInfo fi in fis)
                {
                    //para cada ficheiro da pasta instancia uma thread
                    App.Threads.Add(new FrameRemoverGT(fi.FullName,
directoryPath +@"\FINAL\"+ fi.Name, init , directoryPath));
                }
            //inicia as settings do alchemi manager
            Init(managerIP);
            //inicia o processo das threads
            App.Start();
        }
    }
}
```

9.2.4 FrameRemoverWS

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.IO;
using ICSharpCode.SharpZipLib.Zip;
using System.Web.Hosting;

namespace ExtPDFtoTXTWS
{
    /// <summary>
    /// Summary description for FrameRemoverWS
    /// </summary>
    [WebService(Namespace = "http://FrameRemoverWS/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [ToolboxItem(false)]
    public class FrameRemoverWS : System.Web.Services.WebService
    {
```

```
        [WebMethod]
        public string SetExecutionFrameRemover(byte[] fileZIP)
        {
            //inicia a chave de execucao
            string guidExecution = System.Guid.NewGuid().ToString();
            //path para o caminho da pasta de execucao
            string directoryPathExecution =
HostingEnvironment.ApplicationPhysicalPath + @"FrameRemover\" +
guidExecution;

            UtilsWS uws = new UtilsWS();
            //extrai o ficcheiro zip para pasta source da execucao
            uws.ExtractZipToExecution(directoryPathExecution,
guidExecution, fileZIP);

            ExtPDFtoTXT.FrameRemoverGT fr = new
ExtPDFtoTXT.FrameRemoverGT();
            //inicia o processo
            fr.AppStart(directoryPathExecution, UtilsWS.GetManagerIP());

            return guidExecution;
        }

        [WebMethod]
        public string MySetExecutionFrameRemover(string path)
        {
            return SetExecutionFrameRemover(File.ReadAllBytes(path));
        }

        [WebMethod]
        public byte[] GetExecutionFrameRemover(string guidExecution)
        {
             string directoryPathExecution =
HostingEnvironment.ApplicationPhysicalPath + @"FrameRemover\" +
guidExecution + @"\FINAL";

             if (Directory.Exists(directoryPathExecution))
             {
                 //caso exista a pasta entao a execucao ja foi finalizada
                 UtilsWS uWS = new UtilsWS();
                 //compacta a pasta e devolve o ZIP
                 return uWS.GetBytesOFExecution(directoryPathExecution);
             }

            //caso o processo ainda nao tenha sido finalizado devolve
null
             return null;
        }
    }
}
```

9.2.5 ImageConvert

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Drawing;
using System.Drawing.Imaging;
using Alchemi.Core.Owner;
using System.IO;

namespace ExtPDFtoTXT
{
    [Serializable]
    public class ImageConverterGT : GThread
    {
        public static GApplication App = new GApplication();
        public static string directoryExecution;
        public static DateTime dtInitExecution = DateTime.Now;

        public Image img;

        public int hres;

        public int vres;

        public String where;

        private Image getImg()
        {
            return this.img;
        }

        private String getWhere()
        {
            return this.where;
        }

        private DateTime init;
        public ImageConverterGT()
        { }

        public ImageConverterGT(String path, String where, int hres, int
vres, ImageFormat format, DateTime init )
        {
            this.img = this.loadBitmap(path);
            this.hres = hres;
            this.vres = vres;
            this.where = where;
            this.init = init;
        }

        public override void Start()
        {
            //converte a imagem referente a currente thread
            img = this.setResolution(this.img, this.hres, this.vres);
        }
```

```
        private static void Init()
        {
            Init("localhost");
        }
        private static void Init(string managerIP)
        {
            // specify connection properties
            App.Connection = new GConnection(managerIP, 9000, "user",
"user");

            // grid thread needs to

            if (!App.Manifest.Contains(new
ModuleDependency(typeof(ImageConverterGT).Module)))
            {
                App.Manifest.Add(new
ModuleDependency(typeof(ImageConverterGT).Module));
            }

            // subscribe to ThreadFinish event

            App.ThreadFinish += new GThreadFinish(App_ThreadFinish);
            App.ApplicationFinish += new
GApplicationFinish(App_ApplicationFinish);

        }

        private static void App_ThreadFinish(GThread thread)
        {
            ImageConverterGT imc = (ImageConverterGT)thread;
            TimeSpan ti = DateTime.Now.Subtract(imc.init);
            TextWriter twl = new StreamWriter(directoryExecution
+@"\LOG.txt", true);
            twl.WriteLine(" Imagem " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff") + " Com o tempo de (s) " + ti.Hours.ToString().PadLeft(2,
'0') + ":" + ti.Minutes.ToString().PadLeft(2, '0') + ":" +
ti.Seconds.ToString().PadLeft(2, '0') + "," +
ti.Milliseconds.ToString().PadLeft(3, '0'));
            twl.Close();
        }

        static void App_ApplicationFinish()
        {
            //no final de todas as threads acabarem
            //cria a pasta final
            if (!Directory.Exists(directoryExecution + @"\FINAL"))
            {
                Directory.CreateDirectory(directoryExecution +
@"\FINAL");
            }
            foreach (GThread f in App.Threads)
            {
                //percorre todas as threads da aplicacao e guarda a
imagem convertida na pasta final
                ImageConverterGT fr = (ImageConverterGT)f;
                fr.save(((ImageConverterGT)f).getWhere(),
((ImageConverterGT)f).getImg(), ImageFormat.Png);
            }
```

```
            TextWriter twl = new StreamWriter(directoryExecution +
@"\LOG.txt", true);
            TimeSpan ti = DateTime.Now.Subtract(dtInitExecution);

            twl.WriteLine(" FINISH " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff") + " Com o tempo de (s) " + ti.Hours.ToString().PadLeft(2,
'0') + ":" + ti.Minutes.ToString().PadLeft(2, '0') + ":" +
ti.Seconds.ToString().PadLeft(2, '0') + "," +
ti.Milliseconds.ToString().PadLeft(3, '0'));
            twl.Close();
            App.Stop();
        }

        public void convert(Image img, String toWhere, int xDpi, int
yDpi, System.Drawing.Imaging.ImageFormat format)
        {
            img = this.setResolution(img, xDpi, yDpi);
            this.save(toWhere, img, format);
        }

        public void convert(String startImage, String toWhere, int xDpi,
int yDpi, System.Drawing.Imaging.ImageFormat format)
        {
            Image img = this.loadBitmap(startImage);
            img = this.setResolution(img, xDpi, yDpi);
            this.save(toWhere, img, format);
        }
        private Image loadBitmap(String path)
        {

            return  Image.FromFile(path);
        }
        /**
         * gives a new BitMap with resolution set
         **/
        private Image setResolution(Image input,int xDpi,int yDpi)
        {
            Image returnValue = input;
            ((Bitmap)returnValue).SetResolution(xDpi, yDpi);
            returnValue =(Image) this.ResizeBitmap(((Bitmap)returnValue),
xDpi, yDpi);

            return returnValue;
        }

        private Bitmap ResizeBitmap(Bitmap b, int nWidth, int nHeight)
        {
            Bitmap result = new Bitmap(nWidth, nHeight);
            using (Graphics g = Graphics.FromImage((Image)result))
                g.DrawImage(b, 0, 0, nWidth, nHeight);
            return result;
        }
        /*
         * Saving to a new format
         */
        private void save(String path, Image
input,System.Drawing.Imaging.ImageFormat format)
        {
            input.Save(path, format);
```

```
        }

        public void AppStart(string DirectoryExecution, int Hresolution,
int Vresolution, ImageFormat imgF, string managerIP)
        {
            directoryExecution = DirectoryExecution;
            TextWriter twl = new StreamWriter(directoryExecution +
@"\LOG.txt");
            twl.WriteLine("INICIO  " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff"));
            twl.Close();

            DateTime init = DateTime.Now;
            DirectoryInfo di = new DirectoryInfo(directoryExecution
+@"\SOURCE");
            FileInfo[] fis = di.GetFiles();
            foreach (FileInfo fi in fis)
            {
                //para cada imagem da pasta instancia uma thread
                App.Threads.Add(new ImageConverterGT(directoryExecution +
@"\SOURCE\" + fi.Name, directoryExecution + @"\Final\" + fi.Name +"."+
GetExtension(imgF), Hresolution , Vresolution , imgF, init));
            }
            //inicializa variaveis do alchemi manager
            Init(managerIP);
            //ordena o manageer iniciar as threads
            App.Start();

        }

        private string GetExtension(ImageFormat imgf)
        {
            if (imgf == ImageFormat.Png)
                return "png";
            if (imgf == ImageFormat.Jpeg)
                return "jpeg";
            if (imgf == ImageFormat.Gif)
                return "gif";
            if (imgf == ImageFormat.Icon)
                return "ico";
            if (imgf == ImageFormat.MemoryBmp)
                return "bmp";
            if (imgf == ImageFormat.Png)
                return "png";
            return "png";
        }
    }
}
```

9.2.6 ImageConverterWS

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
```

```
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.Web.Hosting;
using System.Drawing.Imaging;
using System.IO;

namespace ExtPDFtoTXTWS
{
    /// <summary>
    /// Summary description for ImageConverterWS
    /// </summary>
    [WebService(Namespace = "http://ImageConverterWS/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [ToolboxItem(false)]
    public class ImageConverterWS : System.Web.Services.WebService
    {

        [WebMethod]
        public string SetExecutionImageConverter(byte[] fileZIP ,int
Hresolution, int Vresolution, string imageFormat)//ImageFormat imgF)
        {
            //incia variavel de execucao
            string guidExecution = System.Guid.NewGuid().ToString();
            //path para a directoria a usar na execucao
            string directoryPathExecution =
HostingEnvironment.ApplicationPhysicalPath + @"ImageConverter\" +
guidExecution;

            UtilsWS uws = new UtilsWS();
            //extrai ficheiro zip (fileZIp) para a pasta source da pasta
de execucao
            uws.ExtractZipToExecution(directoryPathExecution,
guidExecution, fileZIP);

            ExtPDFtoTXT.ImageConverterGT imgC = new
ExtPDFtoTXT.ImageConverterGT();
            //inicia o processo de conversao
            imgC.AppStart(directoryPathExecution, Hresolution,
Vresolution, GetImageFormat(imageFormat), UtilsWS.GetManagerIP());

            //devolve a chave de execucao para ser depois ser devolvido o
resultado
            return guidExecution;
        }

        public ImageFormat GetImageFormat(string format)
        {
            switch (format.ToLower())
            {
                case ("jpg"): return ImageFormat.Jpeg; break;
                case ("png"): return ImageFormat.Png; break;
                case ("jpeg"): return ImageFormat.Jpeg; break;
                case ("gif"): return ImageFormat.Gif; break;
                case ("ico"): return ImageFormat.Icon; break;
                case ("bmp"): return ImageFormat.MemoryBmp; break;

                default: return ImageFormat.Png;
            }
```

```
        }

        [WebMethod]
        public string MySetExecutionImageConverter(string path, int
Hresolution, int Vresolution , string extensao)
        {
            return
SetExecutionImageConverter(System.IO.File.ReadAllBytes(path),
Hresolution, Vresolution, extensao);
        }

        [WebMethod]
        public byte[] GetExecutionImageConverter(string guidExecution)
        {
            string directoryPathExecution =
HostingEnvironment.ApplicationPhysicalPath + @"ImageConverter\" +
guidExecution + @"\FINAL";
            byte[] bresult = null;
            if (Directory.Exists(directoryPathExecution))
            {
                //caso exista a pasta entao a execucao ja foi finalizada

                UtilsWS uWS = new UtilsWS();
                //compacta a pasta e devolve o ZIP
                return uWS.GetBytesOFExecution(directoryPathExecution);
            }
            return bresult;
        }
    }
}
```

9.2.7 OCR

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Drawing;
using System.Threading;
using System.Collections;
using System.IO;
using Alchemi.Core.Owner;
using OCR.OCRWS;

namespace ExtPDFtoTXT
{
    [Serializable]
    public class OCRGT : GThread
    {
        public static GApplication App = new GApplication();
        public static DateTime dtinit = DateTime.Now;

        public static string PathExecution;
        private System.Drawing.Bitmap image;
        private String lang;
        private String temp;
        private int position;
        private string WSURL;
```

```
        private int getPosition()
        {
            return this.position;
        }

        private String getTemp()
        {
            return this.temp;
        }

        public OCRGT()
        { }

        private OCRGT(String path, String langPath, String lang, int
position, string WSURL, string pathExecution)
        {
            PathExecution = pathExecution;
            if (WSURL.Length == 0)
            {
                this.WSURL = "http://localhost/ExtPDFtoTXT/OCRWS.asmx";
            }
            else
            {
                this.WSURL = WSURL;
            }
            this.lang = lang;
            this.Path = path;
            this.rootPath = langPath;
            this.position = position;

        }

        public override void Start()
        {
            OCRWS o = new OCRWS();
            o.Url = this.WSURL;
            //guarda resultado do texto lido na imagem
            this.temp = o.ReturnTXTOfImage(this.rootPath,
this.lang,this.Path);//, ()image);

         //   this.temp =
        }
        private static void Init()
        {
            Init("localhost");
        }
        private static void Init(string managerIP)
        {
            // specify connection properties
            App.Connection = new GConnection(managerIP, 9000, "user",
"user");

            // grid thread needs to
            if (!App.Manifest.Contains(new
ModuleDependency(typeof(OCRGT).Module)))
            {
                App.Manifest.Add(new
ModuleDependency(typeof(OCRGT).Module));
            }
```

```
            // subscribe to ThreadFinish event
            App.ThreadFinish += new GThreadFinish(App_ThreadFinish);
            App.ApplicationFinish += new
GApplicationFinish(App_ApplicationFinish);

        }

        private static void App_ThreadFinish(GThread thread)
        {
            TextWriter twl = new StreamWriter(PathExecution +
@"\LOG.txt",true);
            TimeSpan ti = DateTime.Now.Subtract(dtinit);
            twl.WriteLine(" Imagem " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff") + " Com o tempo de (s) " + ti.Hours.ToString().PadLeft(2,
'0') + ":" + ti.Minutes.ToString().PadLeft(2, '0') + ":" +
ti.Seconds.ToString().PadLeft(2, '0') + "," +
ti.Milliseconds.ToString().PadLeft(3, '0'));
            twl.Close();
               }

        static void App_ApplicationFinish()
        {
            //apos todas as threads terem finalizado
            TextWriter tw = new StreamWriter(PathExecution +
@"\Result.txt");
            TextWriter twl = new StreamWriter(PathExecution +
@"\LOG.txt",true);
            TimeSpan ti = DateTime.Now.Subtract(dtinit);

            foreach (GThread f in App.Threads)
            {
                //para cada thread guarda o resultado no ficheiro de
resultados
                tw.WriteLine(((OCRGT)f).getPosition().ToString());
                tw.WriteLine(((OCRGT)f).getTemp());
            }
            tw.Close();
            twl.WriteLine(" FINISH " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff") + " Com o tempo de (s) " + ti.Hours + ":" + ti.Minutes +
":" + ti.Seconds + "," + ti.Milliseconds);
            twl.Close();
            App.Stop();
        }

       private String Path;
       private String rootPath;
       public OCRGT(String rootLocation)
       {
           this.rootPath = rootLocation;
       }

       public void setRootPath(String root)
       {
           this.rootPath = root;
       }

        public void AppStart(string RootPath, string guidExecution,
string lang, string WSURL,string ManagerIP)
```

```
        {
            PathExecution = RootPath + @"\" + guidExecution;
            TextWriter twl = new StreamWriter(PathExecution+
@"\LOG.txt");
            twl.WriteLine("INICIO  " + DateTime.Now.ToString("yyyy-MM-dd
HH:mm:ss:fff"));
            twl.Close();

            dtinit = DateTime.Now;
            DirectoryInfo di = new DirectoryInfo(PathExecution +
@"\SOURCE");
            FileInfo[] fis = di.GetFiles();

            for (int i = 0; i < fis.Length; i++ )
            {
                //para cada imagem na pasta source da pasta de execucao
cria uma nova thread
                FileInfo fi = fis[i];
                App.Threads.Add(new OCRGT(fi.FullName, RootPath +
@"\tessdata", lang, i, WSURL, PathExecution));
            }
            //inicia as settings do alchemi manager
            Init(ManagerIP);
            //inicia o processo das threads
            App.Start();
        }
    }
}
```

9.2.8 OCRWS

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.Collections.Generic;
using System.Drawing;
using System.Web.Hosting;
using System.IO;
using System.Configuration;

namespace ExtPDFtoTXTWS
{
    /// <summary>
    /// Summary description for OCRWS
    /// </summary>
    [WebService(Namespace = "http://OCRWS/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [ToolboxItem(false)]
    public class OCRWS : System.Web.Services.WebService
    {
        [WebMethod]
        public string SetExecution(byte[] fileZIP , string lang )
        {
            //inicializa a chave de execucao
            string guidExecution = System.Guid.NewGuid().ToString();
```

```
            //path para a pasta de execucao
            string directoryPathExecution =
HostingEnvironment.ApplicationPhysicalPath + @"OCR" ;

            UtilsWS uws = new UtilsWS();
            //extrai imagens do ficheiro zip para a pasta source da pasta
de execucao
uws.ExtractZipToExecution(directoryPathExecution+@"\"+guidExecution,
guidExecution, fileZIP);

            ExtPDFtoTXT.OCRGT O = new ExtPDFtoTXT.OCRGT();
            //inicia o processo
            O.AppStart(directoryPathExecution, guidExecution, lang,
this.Context.Request.Url.ToString().Substring(0,
this.Context.Request.Url.ToString().LastIndexOf('/')),UtilsWS.GetManagerI
P());
            return guidExecution;
        }

        [WebMethod]
        public string MySetExecution(string path, string lang)
        {
            return SetExecution(File.ReadAllBytes(path), lang);
        }

        [WebMethod]
        public string ReturnTXTOfImage(string rootPath , string lang
,string path)//, Bitmap image)
        {
            string txt = "";
            tessnet2.Tesseract ocr = new tessnet2.Tesseract();
            ocr.SetRootPath(rootPath, lang); // tessdata debug directory
is in c:\temp
            ocr.Init(lang, false);
            Bitmap b = new Bitmap(path);
            List<tessnet2.Word> result = ocr.DoOCR( b, Rectangle.Empty);

            foreach (tessnet2.Word word in result)
            {
                txt = txt + " " + word;
            }
            return txt;
        }

        [WebMethod]
        public string GetExecution(string guid)
        {
            string Text = "False";
            if (File.Exists(HostingEnvironment.ApplicationPhysicalPath +
@"OCR\" + guid + @"\result.txt"))
            {
                Text =
File.ReadAllText(HostingEnvironment.ApplicationPhysicalPath + @"OCR\" +
guid + @"\result.txt");
            }
            return Text;
        }
    }
}
```