

Retrieval on the Grid

Results from the European Project GRACE (Grid Search and Categorization Engine)

WERNER STEPHAN

University of Stuttgart, Germany
Director of Libraries

This paper is based on the internal GRACE position paper - *Information Retrieval on the Grid*. I want to thank especially the main author of this paper Nahum Korda from GL 2006 and Frank Scholze who represented Stuttgart University Library in the project.

Introduction

Internet computing and Grid-technologies promise to change the way we tackle complex problems. They will enable large-scale aggregation and sharing of computational, data and other resources across institutional and geographical boundaries.

Internet computing is just a special case of something much more powerful: the ability for communities to share resources as they tackle common goals. Business today is increasingly international and multidisciplinary. It is not unusual for corporations to span states, countries and continents. It is also not unusual for corporations to bundle together a variety of industries and to collect information and generate expertise in various areas of business, technology and science. E-mail and the World Wide Web provide basic mechanisms that allow such groups to work together. But what if they could link their data, computers and other resources into a single virtual office? So-called Grid-technologies seek to make this possible by providing the protocols, services and software development kits needed to enable flexible, controlled resource sharing on a large scale.

The World Wide Web has facilitated unprecedented ways of speedy global information sharing. The Grid-technology will build on this by allowing facilitating the global sharing of not just information but of tangible assets to be used at a distance. Very large databases - literary terabytes and petabytes of information - that now are geographically confined will become Grid-sharable. This is why - in addition to the computational Grid technology recent efforts are directed into developing data-Grid infrastructures¹.

Project GRACE

GRACE² – Grid Retrieval and Categorization Engine – is an information retrieval technology. Semantic Information Retrieval involves computationally intensive tasks that require extensive computational and storage resources. GRACE undertook the task of investigating whether the current Grid technologies can offer a practical solution to this demand for extensive computational and storage resources. GRACE is the first practical GRID application, designed as a comprehensive retrieval system, tailored specifically to the needs of researchers in any field. It is a unique solution which introduces the concept of knowledge domains, federated search and distributed processing to information retrieval. It collects, categorizes and presents information to the researcher through a simple and user friendly

¹ The most outstanding example is the European [Data Grid](#) project (EDG) and its successor EGEE (Enabling Grids for E-science in Europe) headed by CERN and funded by the European Commission.

² <http://www.grace-ist.org/>

interface. The knowledge domains are defined within loadable ontologies in the form of domain specific Thesauri.

GRACE is based on the ability of distributed systems such as the Grid to offer controlled and authorized sharing of resources – computational, storage, human. This is precisely what the concept of the knowledge discovery makes possible in the domain of the information retrieval (IR). It offers appealing and relevant content (documentation, knowledge basis, etc.), but also allows dynamic document publishing and storage. The solution combines a document management system found on many Intranets with relevant content sources.

An important result has been gaining the experience of which parts of the process of the information retrieval workflow could benefit from being executed on the Grid. During the implementation it became evident that the grid middleware used as the platform provides a response time that is not suitable for the interaction with the end-user. Accordingly, the implementation took the direction of using the grid for computationally heavy pre-processing tasks.

Information retrieval on the whole involves many computationally intensive tasks that require extensive computational and storage resources. These computationally intensive tasks are typically performed as some kind of “text crunching” that may include various aspects of natural language processing, and are designed to transform the original document text into indices optimized for efficient querying. However, the “text crunching” performed by GRACE is even more complex and resource demanding. This is due to the advanced natural language processing functionalities offered by GRACE: categorization, named entity extraction, and concept indexing. Consequently, GRACE undertook the task of investigating whether the current Grid technologies can offer a practical solution to this demand for extensive computational and storage resources. Accordingly, GRACE combines three innovative technologies:

1. Federated search technology to access multiple, distributed content sources in parallel,
2. Natural language processing technology for highly advanced text indexing, and
3. Grid technology for execution of computationally intensive indexing tasks through flexible, just-in-time resource sharing.

The GRACE Toolkit prototype was integrated with Gilda testbed of the Large Hadron Collider (LHC) Computer Group (LCG) that maintains currently the largest data grid worldwide. LCG data grid uses a version of the European DataGrid (EDG) middleware. Gilda is available to other European projects through the courtesy of Istituto Nazionale di Fisica Nucleare (INFN - The Italian National Institute for Nuclear Physics) that is responsible for its maintenance. The project duration is a 30 months and it is led by Telecom Italia, one of the largest telecommunications operators in Europe.

GRID understanding

Grid is currently one of the most promising emerging technologies. It is designed to allow performing computationally intensive tasks with only limited resources by using additional computational resources that are temporarily made available on the Grid by members of a “virtual organization”. In a “virtual organization” all members contribute their limited resources for the benefits of all partners. In this respect Grid is actually based on the timesharing of computational resources within a “virtual organization”. This makes Grid in particular useful for collaboration between geographically wide-spread partners who can use all computational resources available on the Grid during their work hours while other partners don't work.

A special case of this scenario is the Data-GRID. Data-GRIDs are designed to share not only the computational resources, but also the storage space and the content. It is hereby

assumed that the data stored at different places on the GRID, will be used by all partners in a “virtual organization”.

Data Grid-technologies address the problem of efficient storage of petabytes of data (typically generated by scientific experimental instrumentation) by distributing them across a Grid-network. Although due to their enormous size these data cannot be stored in any single central location, they are still required to be highly accessible. Data Grid-technologies attempt to coherently manage these petabytes of distributed data in order to enable their speedy and efficient manipulation.

Requirements from an Information Retrieval Application to run on the GRID

Without going to much in details I will focus on some minimal Requirements which we (i.e. the project partners) believe must be satisfied, in order to ensure that Grid solutions indeed can improve information retrieval.

1. Resources pay-off: the resources contributed to a virtual organisation by any single partner must be alone largely insufficient for the required processing task, since otherwise there are certainly more efficient ways than the Grid to perform the required task.
2. Data-centered processing: Data GRID design assumes that relatively lightweight application is required in order to process massive data input, so that it is more efficient to run the processing application at the location where the data is available, rather than to transport data over the network.
3. Scheduled processing: Grid is suitable for the application that do not require interactive response time, but can rather be scheduled, and run whenever the required resources are available.
4. Trivial parallelism: it doesn't make much sense to run an extensive task on a single Grid work node. The real time saving is accomplished by Grid when breaking down such an extensive task into numerous short tasks that run in parallel on many Grid work nodes.

Information Retrieval

However promising the Grid technology seems to be, the question is whether it might be used for information retrieval applications. Information retrieval involves many computationally intensive tasks that require extensive resources. Under information retrieval I understand retrieval of unstructured textual information, usually stored in various document formats. These computationally intensive tasks are typically performed as some kind of “text crunching” that may include various aspects of natural language processing, and are designed to transform the original document text into indices optimized for efficient querying. The resulting indices are also of significant size, and may be of the same order of magnitude as the original processed text, or even higher. This combination of the computationally intensive tasks with requirement for extensive storage space makes the Grid, and in particular the Data-GRID attractive.

Requirements from GRID to support an information retrieval application

Still the question remains: what can we expect from the Grid in order to use it efficiently for the information retrieval? The best approach to presenting these expectations is by analyzing a typical lifecycle of a Grid job. Here are the major phases:

1. Grid job submission, certification and resources allocation
2. Upload of the computing application and the data
3. Queuing in the local queue

4. Initiation of the computing application
5. Processing
6. Monitoring
7. Download of the results

It is evident that phases 2,3, and 4 could be omitted by running the processing application as a service on the targeted work node. Although running a service on the resources that belong to another member for the virtual organization seems to be primarily an administrative issue, it may also influence certification and the efficiency of resources allocation. The allocation of resources must guarantee that the required service is installed and indeed running on the targeted work node prior to launching the Grid job.

The certification on the Grid is already awkward, since it is repeatedly performed at every step of the Grid job submission. It would be a general expectation from the emerging Grid technologies to simplify the certification process significantly, i.e. to make it more similar to the Web Services certification. On the other hand, the members of the relevant virtual community will be forced to handle a stricter but simplified certification procedure. Another issue related to the phase two is the upload of the data that are to be processed. The design of Data Grids allows these data to be uploaded in advance, and then used by the computing applications as needed. However, this may be not always desirable for the information retrieval.

In the context of information retrieval this data is typically the original text of a document. After the text crunching that is performed on the data, the original document text is of little importance, and does not need to be stored on the GRID any longer. This may suggest to prefer the upload of data as a part of the Grid job submission. For the scheduled Grid jobs the resulting time overhead during the Grid job submission is in any case of little importance. Nevertheless, it may be more efficient to separate completely these two tasks, and to pre-load the data into a temporarily storage on the GRID. In this case the failing GRID jobs would not need to reload the data with every additional resubmission. Pre-loading of the data would also be more suitable if the processing application is run as a service that is then invoked without any additional uploads. It would, however, require an efficient system of data management that insures that the remaining data residues are eventually removed, and that the storage space can be reclaimed.

Although this issue seems to be more of concern to the efficient design of an application that runs on the GRID, it would be preferable if such storage management functionality would be offered directly by the Data GRID middleware, and handled seamlessly by merely replacing the original data with the processed data. If the processing application is run as a service this would merely require invoking the desired service while pointing to the data stored on the GRID, which would be then automatically replaced by the processing results upon the successful completion of the GRID job. This scenario would completely eliminate the phase 7 of the GRID job submission.

The current GRID job failure rate does not seem to be of much concern for the GRID users. Nevertheless, it is of critical importance to timely detect such failures in order to handle them. For this, an interactive monitoring ability is required from the GRID. This would significantly improve the Phase 6 of the GRID job submission. Therefore, the greatest hopes from the emerging WSRF (Web Service Resource Framework) technology are that it will allow interactive messaging.

Besides monitoring, interactive messaging can improve significantly the allocation of resources by allowing their interactive inspection and selection. This could improve the phase 1 of the GRID job submission, and lower the overhead in time required. In future it is possible to envision even WS-resources orchestration that may be extremely interesting for the information retrieval. For example, various text crunching tasks could be exposed as

services on various WS-resources, and different combinations of these tasks could be invoked in order to achieve results satisfying different needs.

The final requirement from the Data GRID (that is not related to the GRID job submission) is to simplify the system installation and configuration. Installation and configuration of current Data GRIDS seem to be extremely difficult, most probably due to the fact that they consist of different layers developed by various producers.

For example, a GRID testbed like GILDA developed and maintained by INFN uses Globus Toolkit 2 (GT2) as the basic middleware. On top of GT2 runs European Data Grid 2 (EDG2) that provides basic Data Grid functionalities, and on top of EDG2 there is still a Large Hadron Collider (LHC) Computing Group 2 (LCG2) installation that uses a specific configuration of EDG2 functionalities. It is obvious that this kind of configuration is extremely difficult to manage and maintain, and that a simpler middleware should be offered for the Data GRIDs.

Architectural Design for Information Retrieval on the Data GRID

In previous chapters I discussed the information retrieval scenario in which the GRID is used for text crunching performed during the indexing. This is indeed the most computationally intensive phase of the information retrieval. On the following pages I will first outline an architecture for the information retrieval application utilizing the Data GRID based on the discussion so far and then show the proposal for an improved design for *distributed* information retrieval on the GRID, and discuss the conditions under which even the index querying may be performed on the GRID.

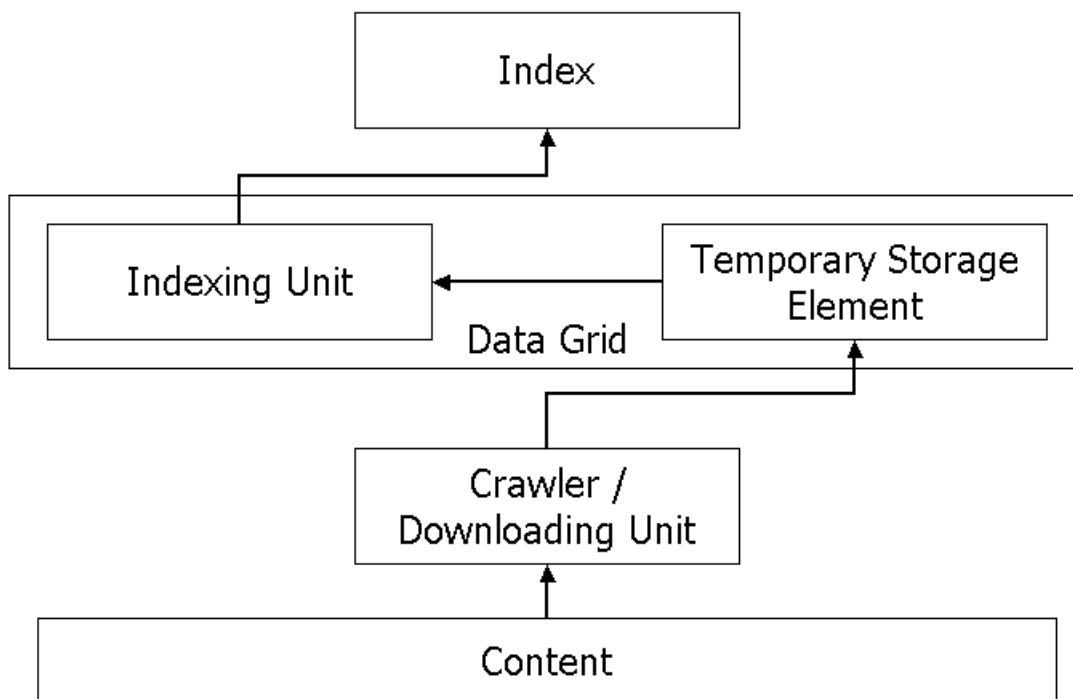


Figure 1 The basic design of an Information Retrieval system using the Grid

The figure depicts schematically the initially proposed architectural design for an information retrieval system utilizing the GRID.

The content is any corpus of documents that needs to be indexed in order to allow retrieving documents by querying the index. This corpus of documents may be web pages on the WWW, or documents stored in a document management system, or documents stored in

any other document repository (such as digital subscription sources) that can be interfaced by the system.

The module responsible for interfacing the content, and systematically downloading it for further processing is the Crawler/Downloading Unit. The downloaded documents are then stored in the Temporary Storage Element on the Grid. This particular aspect of design follows the recommendations from the discussion above. Pre-uploading the data for future processing significantly simplifies the Grid job submission, and minimizes network traffic during the processing phase.

The documents stored in the Temporary Storage Element are then processed by the Indexing Unit that operates on the Grid. This unit performs any kind of required text crunching, which may include extremely computationally intensive operations. Following the recommendations from the discussion above, the Indexing Unit should run as a service thus minimizing the time overhead associated with the upload and initiation of the processing application. It is also hereby assumed that the Indexing Unit maximizes the parallel processing, in order to harness the power of the Grid to the full extent.

The resulting Index is finally retrieved, and stored outside the Grid. This design aims at overcoming the long response time expected when submitting Grid jobs. Accordingly, the index would be queried directly, completely avoiding any involvement of the Grid middleware.

This proposed architecture approximately matches the recommendations of the Grid Information Retrieval (GIR) workgroup of the Global Grid Forum (GGF)³. It harnesses the Grid as a powerful processor for computationally intensive tasks. Nevertheless, this architectural proposal does not maximize the full power of the Data Grid designed to share data stored on the Grid by multiple and potentially various processing applications. In this architectural proposal that data shared on the Grid are the original document texts that can be then processed by various indexing applications. The data that are to be shared in the Information Retrieval are not that much the original document texts, as discussed above, but rather the **index** that should be made available on the Grid to be shared by various querying applications.

In the following chapter we shall argue that this is a crucial requirement for the Information Retrieval, and present an improved architectural design that accommodates this requirement.

Distributed Information Retrieval on the Grid

The information needs of knowledge workers require retrieval from multiple content sources. No single content source can fully satisfy the information needs that are typically encountered in in-depth research and analysis. This need is driven by at least three motives:

1. The need to retrieve information from multiple highly specialized content sources – often offered on the subscription base,
2. The need to integrate internal content sources, such as internal documentation, and
3. The emergence of the so-called “Deep Web” that resides in various databases and document repositories, which are not accessible through the major web-based search engines, but can be queried on demand.

³ N. Nassar, K. Gamiel, J. Morris, "Grid Information Retrieval Architecture", http://www.gridir.org/papers/Grid_Information_Retrieval_Architecture.pdf.

There is evidently a strong need to integrate all of these content sources into a single platform, and create a single access point to all of them. This motivated the introduction of the so-called “meta-search” engines that can access several content sources in parallel, and then integrate the search results into a uniform presentation. The meta-search abilities are nowadays a standard functionality offered by all major knowledge platforms⁴.

This requirement for integration of multiple content sources seem to agree with the general design of the Data Grids: the Data Grid can be envisioned as the virtualization of all the data sources required by the members of a Virtual Organization. Similarly, in the context of the Information Retrieval the Data Grid should serve as the virtualization of all the content sources required by the members of the knowledge-oriented Virtual Organization.

There is, however, one major obstacle in utilizing the Data Grids for the meta-search: many of the content sources that need to be integrated do not allow direct access to the original text except by querying them on demand. This would make in principle pre-indexing of these content sources impossible. The typical solution on this problem is post-retrieval re-indexing of the retrieved documents that was discussed above, and became de facto standard for a variety of meta-search applications.⁵ It is evident from the above discussion that Grid is unsuitable for the post-retrieval indexing due to its response time that greatly exceeds the accepted Information Retrieval standards.

In order to overcome this major obstacle, and allow Data Grid to serve as the virtualization of multiple content sources we shall introduce the concept of Knowledge Domain. Knowledge Domain is a multifaceted representation of the knowledge in a particular domain. It is multifaceted since it consists of several essential aspects:

1. Multiple content sources that are relevant for a particular domain of knowledge, and technically accessible in a digital format,
2. A system of concepts and their corresponding relations that is stored in a digital format, and typically called ontology,
3. Virtual Organization in which members share their computational and storage resources based on (a) clearly defined common interest in a particular domain of knowledge, on (b) agreed conceptualization of this domain of knowledge, and (c) strong consensus regarding the way in which their resources are to be shared.

We shall briefly discuss each of these aspects.

Multiple Content Sources

Knowledge Domain is not merely a list of content sources regarded by the members of a Virtual Organisation as relevant for their interest in knowledge, but it also contains the technical information necessary to access these content sources, and retrieve documents from them. In this aspect, Knowledge Domain serves as an interoperability layer between the users and the information stored in multiple formats and locations.

Ontologies

Ontologies play the central role for the ability of the Knowledge Domains to overcome the obstacle of integrating multiple content sources by utilizing the Data Grid. Ontology can be

⁴ Such as Northern Light, Overture's FAST and AltaVista, Copernic, Convera's RetrievalWare (previously Excalibur), Verity's Ultraseek, and Autonomy

⁵ Such as Vivisimo, WebBrain, KartOO, Grokker, TDNet, and Leximancer

envisioned as an index in which every concept points to the set of documents that use the concept. A simple index generated by the full-text indexing associated merely every single word that occurs in a document with that document⁶. Ontologies, on the other hand, allow not merely single words, but also compound phrases to be associated with a document, and, depending on the complexity of the ontology, this may also include synonyms. In addition, not every single word is automatically indexed, but rather only the terminology bearing critical importance for a domain of knowledge is selected during the indexing. Accordingly, ontologies could be comprehended as concept-based indices as opposed to the simple term-based indices.

Restricted terminology, such as that offered by ontologies, is typically referred to as “controlled vocabulary”, and is extensively used for manual keyword indexing of documents. If performed by a machine in an unsupervised manner, this kind of indexing is often referred to as “automatic classification”, and became a standard functionality of the commercial knowledge platforms. An additional advantage of this method is that the resulting index can be exposed not only for querying, but also for browsing, just like manually created directories.

In the scenario in which the original document text is not accessible except by querying on demand ontology concepts can be utilized to systematically query the content sources. Unlike standard meta-search systems that wait for the user to submit a query, and then distribute it to the multiple content sources, Knowledge Domain can anticipate user’s queries, and retrieve the required documents in advance, by utilizing the ontology concepts by which these documents will be eventually indexed. The concepts from ontology can be used as the pre-set queries distributed to the multiple content sources in order to retrieve relevant documents.

In this manner Knowledge Domains allow documents to be pre-indexed despite the fact that their original text is not directly accessible, and that they originated from various content sources. Unlike the web crawlers that follow the links from one document to another, Knowledge Domains systematically, but selectively query already indexed content sources by only the terminology relevant for a domain of knowledge. This approach enables a complete virtualization of multiple content sources by creating a unified index of all of them. The user factually interacts only with a single index, but nevertheless accesses multiple content sources.

Shared Interest in Knowledge

We emphasized above the importance of the common interests shared by the members of a Virtual Organization. This is even more critical in the context of the Knowledge Domain. We already pointed out that there must be a strong agreement among the members of a Virtual Organization regarding what content sources are relevant for their common interests in knowledge. This agreement must be extended into the common conceptualization of the domain of knowledge.

Every ontology mirrors a particular conceptualization of reality. It is hard to imagine any aspect of human reality that could not be conceptualized in significant different ways, and allow significantly different interpretations. Accordingly, there must be a clear consensus regarding the particular conceptualization among the members of a Virtual Organization that is, consequently, reflected in the adopted ontologies. This agreement guarantees that the documents retrieved by the Knowledge Domain are indeed relevant for the common interest in knowledge, and that they satisfy the common information needs.

⁶ In practice, some often used words, typically called “stopwords”, are excluded during the full-text indexing.

It is also of critical importance that a consensus is reached within a Virtual Organization regarding the use of the shared resources. We discussed above the importance of the administrative consent to allow text-crunching applications to run as services on the shared resources. This may be further extended to the scheduling and prioritization of Grid jobs that can be established as a commonly agreed overall resource sharing policy.

A strong consensus regarding the resources sharing could also lead to a simplified certification policy allowing a kind of a secure “private” Grid. The concept of the Knowledge Domain grounds such resources sharing policies in the common interest in knowledge creating a strong motivational base even for Grid users from heterogeneous backgrounds.

Improved Architectural Design

The concept of the Knowledge Domain enables us now to formulate an improved architectural proposal for the utilization of the Data Grids for the distributed Information Retrieval, which is schematically depicted in **Figure 2**.

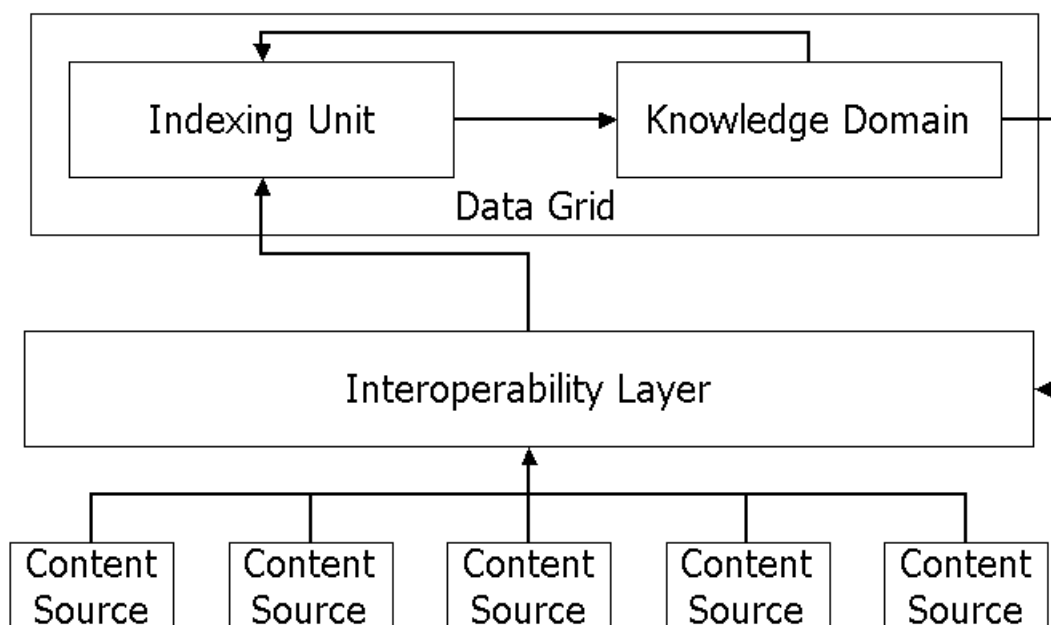


Figure 2: Architectural design for the distributed Information Retrieval on the Data Grid

Compared to **Figure 1**, there are two major differences: the content is not any more a single corpus of documents, but rather multiple, distributed content sources, and the index, which is as ontology an integral element of the *Knowledge Domain*, is kept on the Grid. Both of these improvements are enabled by the concept of the Knowledge Domain.

In the previously proposed architectural design the pre-indexing was not possible for the content sources whose content was not accessible except by querying. These content sources necessitated the post-retrieval processing. Accordingly, the previous architectural design was suitable only for a limited class of content that is expected to be targeted by the Information Retrieval. Actually, the entire previously proposed architecture could be envisioned only as a single content source in the improved design. While in the previous design the integration of the multiple content sources would be required during the post-retrieval processing, and face a series of complications related to the use of the Grid, as

discussed above, the improved design allows this integration of multiple content sources to be performed during the pre-indexing phase.

This shift of the integration of heterogeneous content to the pre-indexing is clearly enabled by the use of ontology concepts as pre-set queries used to query various content sources in advance. This is accomplished by the *Interoperability Layer*, which is merely an access to the multiple content sources. However, the *Knowledge Domain* provides both technical configuration information regarding how to access various content sources, and the queries that are routed to them. The use on ontology concepts as pre-set queries allows access to the content sources whose content is inaccessible except by querying, and completely eliminated the need for the post-retrieval processing. This is the first major contribution of the concept of the Knowledge Domain to the efficient use of the Grid for the Information Retrieval.

The second major contribution of the Knowledge Domain to the efficient use of the Grid for the Information Retrieval is in the fact that it allows the architectural design in which the index is stored, and queried on the Grid. The previously proposed architectural design uses the Grid only for the text crunching, while the index querying is performed outside the Grid. In the improved design the *Indexing Unit* indexes the retrieved documents on the Grid as before, but it is again the *Knowledge Domain* that provides the ontologies required for indexing.

These ontologies are then populated by the pointers to the associated documents, and serve as an index to the underlying content sources. Accordingly, instead of interfacing a separate index for each target content sources, the end user interacts with a single index for all of them.

The interaction with a single index is significantly simpler than routing a query to multiple indices. Even without the post-retrieval processing required in the previous design, query routing to the multiple indices is alone complex enough, and significantly influences the response time of the system. It also requires reformulation of the original query to the specific query syntaxes supported by the various content sources, parsing of the lists containing the search results, and downloading of the actual documents. In the improved architectural design all this processing overhead is performed during the pre-processing.

Query routing also necessitates the unification of the retrieved results in a unified structure prior to their presentation. As discussed above, this is a kind of processing that does not support trivial parallelism, and is thus not well suitable for the Grid. Although not a computationally intensive processing task, unification of retrieved results still causes an additional time overhead, and is better omitted.

Through the use of ontologies this unification is automatically achieved when populating them during the pre-indexing. Accordingly, it becomes sufficient to query the ontologies encapsulated in a Knowledge Domain, in order to retrieve at once all relevant results. Querying ontologies in a Knowledge Domain can be designed as a single Grid job that invokes a single querying mechanism installed and run as a service somewhere on the Grid, as we shall discuss this in more detail in the following chapter.

It is evident that the *Knowledge Domain* plays a very central role in this architectural design. At every step the *Knowledge Domain* provides the necessary information, and it in the end stores the resulting index. From the user's point of view, the *Knowledge Domain* is thus an ultimate virtualization of the underlying, distributed content sources, and serves as the single access point to all of them in parallel. The ultimate end-user does not even need to know which are these distributed content sources, and what are the ontologies used for indexing. – For the end-user, a Knowledge Domain is simply an efficient access to the knowledge.

Knowledge Domain on the Grid

In order to access multiple, distributed content sources with the first proposed architecture it would be necessary to keep an index for each of them. It would be also necessary to perform some kind of post-retrieval processing in order to integrate content sources that are inaccessible except by querying them. The concept of Knowledge Domain efficiently solves both of these problems. By eliminating the need for post-retrieval processing, and completely transferring the weight to pre-indexing of the information, Knowledge Domain efficiently harnesses the advantages of the Grid to the full extent. This is why ontologies are of critical importance for the Information Retrieval on the Grid, and must be, consequently, an integral part of the architectural design.

Nevertheless, ontologies are also used for querying. Keeping them on the Grid necessitates the best possible response time in order to ensure that the retrieval of the search results matches the current Information Retrieval standards. In the previous section we discussed the general conditions that must be satisfied to ensure such response time. In the following chapters we shall summarize them, and apply them to the specific scenario of querying ontologies stored on the Grid.

The most essential condition is to run the querying mechanism as a service on the Grid. This condition omits the downloading and the initiation of such mechanism – two operations that can add significant time overhead to the submission of a Grid job. The concept of the Knowledge Domain that implies strong consensus among the members of a Virtual Organization can efficiently solve the administrative aspects necessary for this condition.

The strong consensus among the members of a Virtual Organization could also allow simplification of the certification process, and thus further improve the procedure of the Grid job submission. How this simplification of certification could be accomplished is still an open research topic. It would be also required to research the scenario in which the certification mechanism could be used for billing according to the use the Information Retrieval system on the Grid.

The final obstacle to the rapid Grid job submission is the allocations of the Grid work node on which the querying Grid job would be launched. Fixing the number of Grid work nodes on which this particular service will run could efficiently solve this. Instead of submitting the querying Grid job to the Resource Broker module of the Data Grid, it would be submitted directly to the Grid work node on which the querying mechanism runs, and thus completely omit the resource allocation.

An even better solution could be offered through the use of web services-resources interface. A user could simply invoke a WS-resource for querying, and the WS-resource would then seamlessly interact with the work node running the querying service.

Conclusion

The original view of GRACE was that it would be an interactive search and retrieval engine. During the course of the project much was learned by the consortium of the operating characteristics of the Grid as it is implemented in this instance. The view that the Grid will be the successor to the web as the internet infrastructure is a long way from realization and the limitations of the present iteration became apparent during the project.

Where GRACE originally envisaged a real-time interaction between the user and the data sources it became apparent that the grid response times did not allow for this. Grid is optimized for batch submission of multiple jobs and the GRACE architecture and operation were redesigned to cope with this characteristic. For this reason GRACE became a retrieval

and categorization technology with pre-processing and indexing of queries. By this method it became possible to give the end user a rapid response to the initial query, using pre-indexed and documents, whilst the search and categorization was submitted and in process in the background.

This paper outlined a possible architecture of an Information Retrieval system based on the concept of Knowledge Domains that aims at efficiently overcoming the obstacles of present Grid implementations, and thus allowing the Information Retrieval to fully harness the power of Grid, without compromising its efficiency.

References

Maurizio Cecchi, "A Contribution on Knowledge Management on Grid"

<http://www.grace-ist.org/docs/GGF-KM%20on%20Grid.pdf>

Jawed Siddiqi, "Requirements for Knowledge Discovery within the Grid Space"

<http://www.grace-ist.org/docs/GGF-knowledge%20discovery%20reqs.pdf>

Nahum Korda, "GRACE: Lessons Learned"

<http://www.grace-ist.org/docs/GGF-Lessons%20learned.pdf>

In: *GGF12 - The Twelfth Global Grid Forum* September 20-23, 2004 Brussels, Belgium

Frank Scholze, Glenn Haya, Jens Vigen, Petra Prazak, "Project GRACE - A grid based search tool for the global digital library" In: *7th International Conference on Electronic Theses and Dissertations*, June 3-5, 2004, University of Kentucky, Lexington, KY USA

<http://www.uky.edu/ETD/ETD2004/scholze/etd2004.ppt>

Glenn Haya, Frank Scholze, Jens Vigen, "GRACE - Developing a Grid-Based Search and Categorization Tool" In: *HEP Libraries Webzine* Issue 8 / October 2003

<http://library.cern.ch/HEPLW/8/papers/1/>

Jawed Siddiqi, Babak Akhgar and Mehrdad Naderi "Towards a Grid Enabled Knowledge Management Services" In: *Proceedings of UK e-Science All Hands Meeting 2003*, 2-4th September, Nottingham, UK

<http://www.nesc.ac.uk/events/ahm2003/AHMCD/pdf/073.pdf>