分散 RDF データベース応用のためのグリッド基盤に基づくミドルウェアの実現

Design and Implementation of OGSA based Grid Database Middleware for Distributed RDF Database Applications

小島 功[1] 木本 正裕[2]

Isao KOJIMA Masahiro KIMOTO

グリッドのアーキテクチャに基き，分散 RDF データ応用を構築するためのミドルウェア，OGSA-DAI-RDF の開発について述べる．これは，OGSA-DAI と呼ばれるミドルウェアの拡張で，SPARQL 検索などの RDF 処理を提供する．第 3 者転送を含む分散処理を容易に記述・実行できるようにした点が特徴であり，実現したソフトの性能評価も行ったので報告する．

In this paper, we present a service-based database middleware called OGSA-DAI-RDF, which is an extension of the OGSA-DAI middleware in order to support RDF databases. The software provides various RDF data processing activities including W3C SPARQL query interface. Since OGSA-DAI-RDF is based on the OGSA architecture, it is easy to construct distributed RDF database application which includes the third party transfer. Performance evaluation including the reasoning function shows the usefulness of the system.

1. まえがき

グリッド[1]で知られる分散環境の目標の一つは，広域を含む分散，並列の環境で，大量のデータと大規模な計算を結び付けた応用を効果的に処理することである．グリッドの普及と発展に伴って，意味処理や推論を含んだ高度応用をグリッド上で支援する，セマンティック・グリッド[2]の分野が注目されている．ここでは，大規模なグリッド応用へのセマンティックWeb技術の適用や，セマンティックWeb応用の広域化・大規模化に伴うグリッドによる支援などが課題となっている．

さて，セマンティックWebの分野では，RDF(Resource Description Framework)[3]で表現されたデータを扱うことが不可欠であり，セマンティック・グリッドの応用でも，分散・大規模なRDFデータをグリッド上で扱うデータベース環境が必要である．

RDFのデータベースシステムは多く知られており，HTTP経由の遠隔アクセスを支援する[4][5]．しかし，これらはグリッドのミドルウェア[6]とその機能，例えばGSI(Grid Security Infrastructure)と呼ばれる認証基盤を支援せず，データベース処理とグリッド上での計算処理との連携ができない．また，クライアント・サーバ型のアクセスモデルに基づくため，第3者転送（サーバでの検索結果をクライアントでない第3のサイトに送る）が支援できず，サイト間での中間結果の相互転送といった分散問い合わせ処理に関わる基本機能の実現が容易でない．

[1] 正会員 産業技術総合研究所 kojima@ni.aist.go.jp
[2] 非会員 ビジネスサーチテクノロジ kimoto@bsearchtech.com

そこで本稿では，分散RDFデータ応用の構築支援を目的とし，グリッドの標準アーキテクチャOGSA(Open Grid Services Architecture)[7]に基づく，OGSA-DAI-RDFと呼ばれるミドルウェアを開発したので，その評価とあわせ報告する．

2. OGSA-DAI-RDF とそのアプローチ

2.1 背景と要求

著者らはグリッド上のデータベースシステムの研究開発の一つとして，Federated SPARQL[8]の研究を行っている．これは，分散した RDF データベースを連携させ，仮想的に一つのデータベースを実現するもので，SPARQL の分散問い合わせ処理・最適化などの新しい問題がある．応用としては、広域でボトムアップ的に作成されたメタデータに基づく知識基盤の構築や，セマンティックSOA(Service Oriented Architecture)[9]におけるレジストリの分散化などが想定されている．この問題解決の支援のためには，分散問い合わせ処理を含めた分散 RDF データ応用を容易に構築できる，以下の機能を有するミドルウェアが必要である．

1) Web サービスに基づき，RDF の標準問い合わせ言語である SPARQL[10]による問い合わせを支援する機能で，各サイトのデータベース管理ソフトトウェアの相違を吸収できるもの．
2) 問い合わせ処理を含む分散 RDF 処理において，各サーバで必要な処理やその手順を容易に記述できる機能で，第 3 者転送を含むサーバ間データ転送や同期を支援するもの．

この要求に対し本稿では，OGSA-DAI(OGSA Data Access and Integration[11])というグリッドのデータベースミドルウェアを， RDF のデータ形式や問い合わせ言語 SPARQLを扱うよう拡張することで，これらの機能を実現した．

2.2 OGSA-DAIによるアプローチとその特徴

OGSA-DAIは，英国e-Scienceプロジェクトで研究開発中のミドルウェアで，OGSAに基づき，Webサービスによる遠隔のデータベースのアクセスを提供する．現在，OracleやPostgreSQL，MySQLなどの関係データベースと，eXist，XindiceなどのXMLデータベースが支援されており，次のような基本的特徴を有する．

1) グリッドのセキュリティ基盤の支援と第3者転送の実現:

クライアントが処理を依頼するサーバから，さらに第3のサイトへ処理を呼び出す場合，単純な認証ではなりすましの問題がある．グリッドにおける標準セキュリティ基盤であるGSIは，証明書のdelegationの仕組みを有し，安全な認証の伝搬に基づいた第3のサイトへの処理の依頼が可能である．OGSA-DAIはGSIを支援するデータベースミドルウェアで，単一のアカウントでの計算とデータベース処理の連携や，検索結果を他のサイトにFTPで転送するなど第3者転送を安全に実現できる．

2) Activity Framework:

分散データ処理を行う場合，例えば１つのサイトで1)SPARQLでローカルな検索をし，2)他のサイトから送られるデータと結合し，3) 結果を圧縮し，4)HTTPやFTP等でさらに次のサイトに送る．という一連の処理を指示し，実行させる必要があるが，同一サイト内の処理をWebサービス同士の連携で実現するのは効果的ではない．OGSA-DAIは，SQLやXPathの問い合わせや結果の圧縮，加工，FTPでのデータ転送など，データベースの周辺処理を含めた機能を，Activityという独自の単位で実現し,1つのサービスに対する1つの要求は，これらActivityの連携したワークフローとして実行させる．例を3節で述べるが，連携した複数の操作を1つのサービス要求の中で記述できるので，分散データ処理応用の記述を単純にすると共に，サービス連携のオーバーヘッドを軽減している．OGSA-DAI上の分散SQLシステムであるOGSA-DQP（Distributed Query Processing)[12]は，この枠組

みを有効に利用して実装されている.

3) 標準化活動との連携:

グリッドの場合，他のデータベースソフトなど異種のシステムとの相互接続性の提供が不可欠で，インターフェイスの開発だけでなく，標準に基づく相互接続性を提供することが重要である．グリッドの分野では，OGF（Open Grid Forum）というコンソーシアムにおいてデータベースアクセスの標準（WS-DAI）が議論され，OGSA-DAIはその参照実装の1つとして検討されている．

以上**1),2)**,のように，OGSA-DAIの提供する枠組みは，分散SPARQL問い合わせ処理を含めた分散RDFデータ応用を実現するために有益で，これをRDFに拡張することは意義が高い．また，**3)**の点でも，WS-DAI規格でRDFを扱うよう拡張[13]することが検討されており，相互互換性の点でも有益である．

2.3 比較

現在，グリッド上で大規模なRDFデータベースを扱うソフトウェア基盤は存在しない．また，SPARQLではサイト同士の情報の結合が暗黙的に記述されるため，その分散処理はSQLの分散問い合わせ処理に比べ複雑で，研究開発もあまり知られていない．代表的なものにDARQ(Distributed ARQ)[14]がある．これは複数のJenaサーバ上で分散SPARQLの環境を構築するものである．OGSA-DAI-RDFは分散SPARQLも含めた分散RDFデータ処理応用を容易に構築するためのミドルウェアで，分散SPARQLそのものを実現するソフトウェアではないが，比較すると，

1) 特定のシステムに依存しない．
2) サーバ間の第3者転送など，クライアント・サーバ型のアクセスに限らないデータ処理が可能で，中間結果の相互転送を利用した処理など，問い合わせ最適化の余地が大きい．
3) 各サイトで実行すべき処理をActivityのワークフローで記述できるので，分散処理の記述，実装が容易である．

という特徴がある．

3. OGSA-DAI-RDF の設計と開発

3.1 システムの概要

概念図を図1に示す．基本的には，OGSA-DAIが支援する関係データベース,XMLデータベースに加え，RDFデータベース型を拡張し，次の種類のRDF処理のActivityを実現したものである．

1) **SPARQLの問い合わせ処理(SPARQLQueryStatement):** W3Cの標準言語SPARQLに基づく問い合わせ処理機能
2) **推論処理(ontologyReasonerActivity):** RDFデータ（トリプル）の集合（グラフ）に対する，RDFSやOWLの推論機能．
3) **グラフ管理(GraphManagementActivity):** RDFデータセット（RDFのグラフの集合）の管理機能（リストや削除，追加）
4) **トリプル管理(TripleManagementActivity):**RDFデータ（トリプル）の基本操作機能．トリプルの更新や追加など．

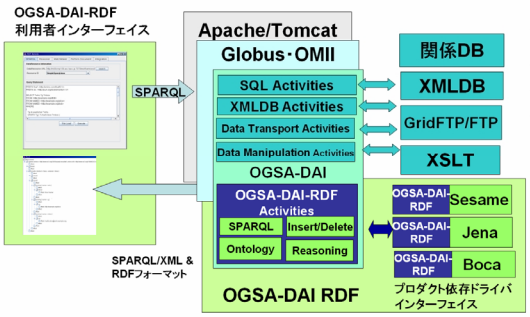

図1. OGSA-DAI-RDFの基本構成
Fig.1. The Architecture of OGSA-DAI-RDF

他にも，オントロジの操作Activity等を開発中である．

各ActivityはXML形式のインターフェイスを持ち，Webサービスであ るOGSA-DAIを経由して呼ばれる．1)のSPARQLQueryStatementActivityを実行するXML記述を図2に示す．<query>で囲まれた部分がSPARQL文である．なお，対応するJavaのAPIを提供して，応用のプログラミングを容易にしている．

```
<xml version="1.0" encoding="UTF-8" ?>
<perform xmlns="http://ogsadai.org.uk/namespaces/2005/10/types">
  <documentation>SPARQL Query Sample</documentation>
 - <sparqlQueryStatement name="myActivity2">
 - <query-request>
   <query>
   PREFIX foaf: <http://xmlns.com/foaf/0.1/>
   PREFIX dc: <http://purl.org/dc/elements/1.1/>
   SELECT ?who ?g ?mbox
   FROM <http://example.org/dft.ttl>
   FROM NAMED <http://example.org/alice>
   FROM NAMED <http://example.org/bob>
   WHERE { ?g dc:publisher ?who . GRAPH ?g { ?x foaf:mbox ?mbox } }
   </query>
  <default-graph-uri>http://example.org/dft.ttl</default-graph-uri>
  <named-graph-uri>http://example.org/bob</named-graph-uri>
  <named-graph-uri>http://example.org/alice</named-graph-uri>
</query-request>
  <query-result name="SelectOutput" />
  </sparqlQueryStatement>
  </perform>
```

図2 SPARQL問い合わせActivityを実行するXML
Fig.2 An XML request to invoke a SPARQL Activity.

OGSA-DAI-RDFは，OGSA-DAIの関係データベースやXMLに対し，RDF固有の問題とその解決のため，次の特徴を有する．

1)中間インターフェイス(ドライバ)層の設定: RDFのデータベースはシステムごとに仕様やインターフェイスが大きく異なる．そこで，ソフトウェアの相違を吸収する中間インターフェイス層を実現した．この入出力インターフェイスは，W3Cの標準プロトコルとデータフォーマットに準拠する．これにより，将来他のRDFデータベースシステムを支援する場合の開発を容易にした．

2)推論Activityの実装: RDFの問い合わせ処理では，オントロジに基づいた検索など，推論(Reasoning)を伴う検索が不可欠である．しかし，SPARQLは推論の実現方法を規定しないので，推論を伴う検索はデータベースの実装により異なる．そこで，SPARQLの問い合わせと推論の処理とを別のActivityとして実装し，推論を伴う問い合わせは，推論で導出されたグラフに対する問い合わせ処理として実現した．標準の言語と実装依存性の高い推論処理を分離することで，より多くのシステムに対応する．一方，この方法はActivity間でグラフ全体が渡されるため，性能の問題が発生しやすく，4節で評価する．

3)ストリーム入出力の支援: SPARQLの関連規格[10]では，SPARQLの検索はURLで指定されたグラフの集合に対して行われるが，本ミドルウェアではActivityの入出力となるストリームデータに対しても，問い合わせ言語による検索が可能なように拡張して実装した．この結果，**2)**の推論Activityからストリーム的に出力されるグラフに対する問い合わせ処理が可能になった．また，他のサイトからの検索結果と，ローカルに指定されるグラフに対して，これらをマージしたデータ集合への問い合わせ処理が可能となり，分散問い合わせ処理の記述が容易になった．

3.2 ソフトウェアの構成と実現.

OGSA-DAI-RDF は分散 RDF 応用の構築のためのミドルウェアであるが，図 3 に示すGUIを有し，タブで区別された次の機能を利用者に直接提供する．

1) SPARQL の対話的な入力と実行機能．
2) 推論付きの SPARQL の入力と実行機能．推論は RDFS と OWL を支援する．
3) グラフ管理や操作，トリプルの操作・挿入など RDF データベースの基本操作機能．
4) 分散ワークフローの作成と実行機能．

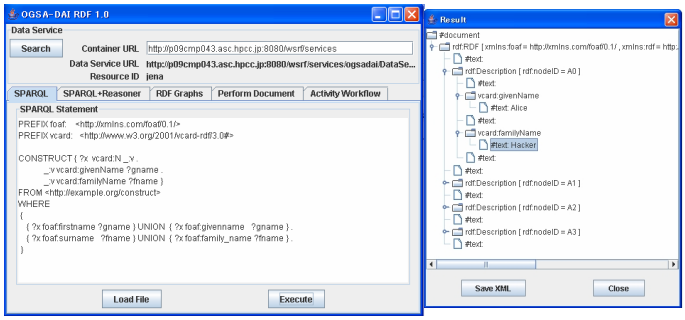

図 3　OGSA-DAI-RDF の SPARQL 問い合わせと結果表示

Figure.3　A Screenshot of the SPARQL Query Interface

3.3 OGSA-DAI-RDFにおけるActivityワークフロー

ここでは，OGSA-DAI-RDF において各サイトおよびサイト間で行われる処理(Activity のワークフロー)がどのように記述されるかを示す．例として先に述べた，

1)　あるサイトで SPARQL による検索を行い，
2)　結果を加工して
3)　データを他のサイトに送る，

処理を考える．これは，OGSA-DAI-RDF では図 4 のような1つのXML表現で記述される．Activity の間のデータ転送は入出力の名前(図で赤文字)で結びつけられており，このXMLを送れば，該当する Activity が実行され，データが相互に転送される．

```
<perform> <documentation> SPARQL Query Sample </documentation>

 <sparqlQueryStatement name="myActivity">
   <queryRequest rdb="/var/ogsadai/repository/dblp.dat">
   PREFIX dblp:&lt;http://dblp.uni-trier.de/rdf#&gt;
    SELECT ?title WHERE {
    ?contributor dblp:title ?title.
    ?contributor dblp:year "1997"  }
   </queryRequest>
   <queryResponse name="SesameTestOutput"/>
  </sparqlQueryStatement>

  <xslTransform name="transform">
    <inputXSLT from="deliverXSLTOutput"/>
    <inputXML from="SesameTestOutput"/>
    <output name="transformedOutput2"/>
  </xslTransform>

 <deliverFromFile name="deliverXSLT">
    <fromFile>/opt/ogsadai-wsrf-2.1/integration/demo/sparql_to_rdf.xsl</fromFile>
    <toLocal name="deliverXSLTOutput"/>
  </deliverFromFile>

   <deliverToGDT name="delivery">
    <fromLocal from="transformedOutput"/>
    <toGDT streamId="SESSIONID:inputStream2" mode="block"
    resourceName="jenatest">http://localhost:9090/wsrf/services/ogsadai/DataService
    </toGDT>
  </deliverToGDT>
</perform>
```

SPARQL問い合わせ

結果をXSLTで加工

加工結果のファイルを取得

他のサイトに転送

図 4. 分散RDF処理のためのXML要求の例.

Fig.4. An XML request to perform Activity workflows.

同様に，データを受ける側のサイトでも，

1)　データをFTPなどで受信して中間ファイルに格納し
2)　他の結果と合わせて統合する．

といった処理がやはり1つのXMLで記述できる．処理の受け側と送り側に送られる2つのXMLの間でセッションを共有(図で茶文字)できるので，これらの要求を各サービスに別個に投げても，サイト間で連携した分散処理が可能である．

Activity ワークフローは，GUI に基づくエディタを使って記述することもできる．図 5 に画面例を示す．図の左側のリストがサービス上での処理ワークフローを構成する Activity 群であり，この例では以下の4つの処理によるワークフローを定義している．

1)　**deliverfromURL0** および，**2)deliverfromURL1**：RDF のデータと推論に用いるスキーマを，それぞれ URL0，URL1 内で記述された 2 つの別のサイトから HTTP を使って取得．
3)　**ontologyReasoner0:**　取得したデータに推論をかけ，結果のグラフを生成，出力．
4)　**SparqlQuery0:**　3)で出力されたグラフを受信して問い合わせを実行．

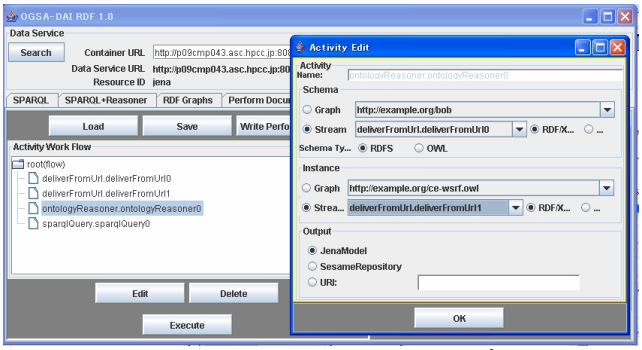

図 5. OGSA-DAI-RDF のワークフローエディタ.

Fig5. Activity Workflow Editor of OGSA-DAI-RDF

図の右側の入力指定画面は，**3)**の Reasoner の Activity の編集画面であり，2つのストリーム入力からRDFS に基づく推論を行い，結果を Jena のモデル形式で出力することを指定している．構築したワークフローはそのまま実行(Execute)できる．

問い合わせ処理を含めた多様なデータ処理を容易に連携できる GUI 環境を提供することで，RDF にかかわる分散データ処理を容易に実現可能としている．

4. 性能評価

4.1 概要

開発したOGSA-DAI-RDFの性能を評価した．本ミドルウェアは，OGSA-DAIから呼ばれるサブシステムなので，OGSA-DAIそのものの処理性能[15]は改良できない．そこで本稿では，1) クライアントからのネットワーク越しのRDF問い合わせの全体応答時間が，実用的な性能を満足すること，2) 3.2節に示したような，DAI-RDF固有の特徴である，Reasonerを別Activityとして実装した方法が実用性を持つこと，3) 大規模なデータ応用に対しての実用性があること，に注目し，脚注の環境[3]で評価を行った．いずれもWebサービスに基づく遠隔呼び出しに基づく評価である．

4.2 問い合わせ応答時間の評価：SPARQL検索処理

OGSA-DAIの関係データベースのベンチマーク[15]に基づき，SQLの代わりにSPARQLを発行し，Jenaと，Sesameの提供する2つの実装(MemoryとNative)について評価した．検索データはWordNetを用い，検索結果の個数に応じたクライアントでの応答時間(10回の平均，秒)を評価した．図6のように，おおむね検索結果の個数に応じた応答時間増となった．Sesameのメモリ上の実装(Server)の結果から，ネットワーク通信のオーバーヘッドは小さいと判断できる．

次に，Jenaを用いて問い合わせ処理の内訳時間を評価した．表1のように，ミドルウェア内のデータ加工に最も時間がかかり，Jenaシステム単体での検索時間は小さいことが示された．これは，Jenaの検索結果がmodelと言われる独自形式で，最終的な答を標準のXML形式に変換するとき，Jenaの提供するAPIを用いて処理した結果である．改良は今後の課題である．SesameではXML形式が直接出力できるので，このような加工時間は発生しない．

3　AMD　Opteron 2.0Ghz x 2way，主記憶6GB，ディスク500GB，SuSE linux9. Java1.5,ヒープ512MB, OGSA-DAI2.2, GlobusToolkit4, Tomcat5

全体として，応答時間が接続先のデータベースシステムの実装に大きく依存する結果となり，ミドルウェアのオーバーヘッドは応答時間に対し主要な要素とならないことが示された．

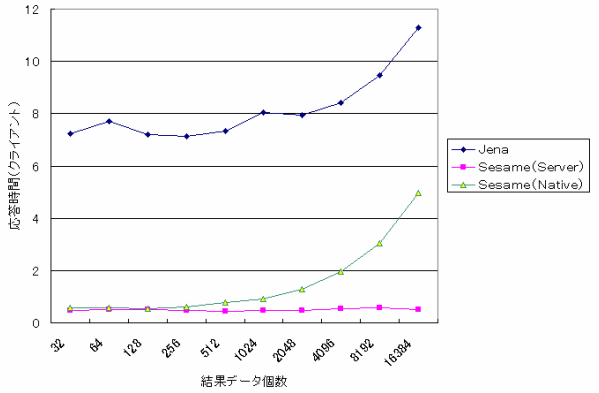

図6 Javaクライアントからの応答時間(秒)
Fig.6. Response Time Evaluation from Java Client

表1 Jenaにおける処理の内訳
Table 1. Detailed Processing Analysis using Jena

結果の個数	クライアント応答時間(全体)	DB単体の実行時間	データ加工・変換	呼び出し準備その他	Activity処理全体	OGSA-DAIのオーバヘッド
32	7.2474	0.0014	6.4983	0.0462	6.5462	0.7012
64	7.7051	0.0012	7.0637	0.024	7.089	0.6962
128	7.1994	0.0017	6.9647	0.0236	6.9904	0.209
256	7.1442	0.0019	6.6118	0.022	6.6359	0.5893
512	7.3276	0.0024	6.6354	0.0268	6.6648	0.6626
1024	8.0345	0.0015	6.6079	0.0695	6.679	1.3555
2048	7.9485	0.0014	6.9375	0.0227	6.9618	0.9867
8192	8.4111	0.002	6.12	0.024	6.146	2.2651
8192	9.4505	0.0015	6.781	0.0256	6.8083	2.6422
16384	11.2763	0.0016	6.9483	0.0218	6.9717	4.3045

4.3 RDF固有の処理：ReasonerとActivityの連携

本システムではRDF固有の要求から，3.2節に示したように，ontologyReasonerActivityを定義し，推論と問い合わせ処理を別個に行うよう実現した．この方法は，推論を行ったグラフ全体を次の問い合わせ処理に送り，そこで検索を実行する必要があるため，性能の低下の可能性がある．そこで，4.2と同様の評価方法で，推論(Reasoner)とそれに対する問い合わせ(SPARQL)との，2つのActivityを連携させた処理の時間を評価した．

表2.推論付き問い合わせ処理時間(秒)
Table 2. Performance of SPARQL with Reasoner (seconds)

		Reasoner			SPARQL Activity			
	クライアントからの応答時間	前処理	実行時間	後処理	前処理	DBサーバ実行時間	データ加工	後処理
Jena	1.0435	0.051	0.397	0	0.0553	0.0059	0.049	0
Sesame (Memory)	1.6494	0.063	0.763	0	0.054	0.0537		0
Sesame (Native)	1.6053	0.116	0.773	0	0.0448	0.0442		0

データはOWLで，論文[16]のものを用いた．比較のため，Sesameにより推論結果をディスクに書き込み，そのURLに対してSPARQLを実行した場合を示す(Jenaではサーバの制限から，対応する操作が実現できない)．結果は表2の通りで，2つのActivity(ReasoningとSPARQL)の連携に伴うデータの転送(前処理)はほとんど無視でき，URLを渡すSesameの場合より推論の実行時間も含め高速となり，ストリームに基づく処理の有効性が示された．なお，ここではmodel形式でデータ転送を行うため，先に問題となったデータ変換は，2つのActivity間では発生しない．

4.4 大規模評価：LUBM(Lehigh University BenchMark)

LUBM[17]は，RDFデータベースシステムのベンチマークとして提案されたもので，サンプルデータと14個の問い合わせからなる．基本的にはSesameやJenaなどのDBサーバ本体に対するベンチマークであるが，参考のために評価した．JenaとそのReasonerを用い，クライアントでの応答時間を測定したものを表3に示す．[17]と比較してタイムアウトが発生する場合が4個から3個に減少する結果が得られ，実用性の指標としては十分と考えられる．

表 3. LUBMベンチマークの結果(秒，Jena，Javaクライアント)
Table 3. The result of LUBM benchmark (seconds)

Q1	Q2	Q3	Q4	Q5	Q6	Q7
4.0097	timeout	6.946	4.848	16.8041	6.3892	timeout
Q8	Q9	Q10	Q11	Q12	Q13	Q14
109.5456	timeout	6.1108	3.4489	3.1776	8.9399	6.0145

5. まとめと今後の課題

OGSA-DAI-RDFの開発と評価について述べた．OGSA-DAIをRDFに拡張した点が特徴であるが，分散RDFデータベース応用の実現のための基盤として十分な実用性を有していると考える．ソフトウェアは http://www.dbgrid.org/を通じ公開中で，引き続きこの環境上で分散SPARQL処理の研究開発を行い，広域にわたる分散知識基盤の構築に貢献する予定である．

謝辞：日頃から有益な議論を頂く，産総研グリッド研究センターのSaid Mirza Pahlevi,的野晃整，Steven Lyndenの各氏に深謝します．

[文献]

[1] I.Foster, C.Kesselman, “The GRID 2”, Morgan Kaufmann, (2003).
[2] Semantic Grid, http://www.semanticgrid.org/
[3] Manola.F, Miller,E, RDF Primer, W3C Recommendation, http://www.w3.org/TR/rdf-primer/ (2004).
[4] Jena, http://jena.sourceforge.net/
[5] Sesame, http://www.openrdf.org/
[6] Globus Toolkit, http://www.globus.org/
[7] OGF OGSA Working Group http://forge.ofg.org/
[8] http://www.w3.org/2007/05/SPARQLfed/
[9] S.Sekiguchi “AIST-SOA for Building Service Oriented e-Infrastructure”, CCGRID (2006).
[10] SPARQL, http://www.w3.org/TR/rdf-sparql-query/
[11] OGSA-DAI, http://www.ogsadai.org.uk/
[12] S.Lynden et al, “Design and Implementation of OGSA-DQP, A Service based Query Processor” in Submission
[13] I.Kojma and S.Mirza, “WS-DAI RDFS Querying Specification”, http://forge.ogf.org/ (2006).
[14] DARQ, http://darq.sourceforge.net/
[15] B.Dobrzelecki et al. “Profiling OGSA-DAI Performance for Common Use Patterns”, Uk All-Hands Meeting, (2006).
[16] S.Mirza and I.Kojima, "S-MDS:A Semantic Information Service for Advanced Resource Discovery and Monitoring in WS-Resource Framework",3rd Semantic Grid Workshop, (2006).
[17] Y. Guo, Z. Pan, and J. Heflin. LUBM: A Benchmark for OWL Knowledge Base Systems”. Journal of Web Semantics 3(2), (2005).

小島 功 Isao KOJIMA
産業技術総合研究所グリッド研究センターデータグリッドチーム長．データグリッドの研究に従事．情報処理学会,ACM,IEEE 各会員. OGF(Open Grid Forum)メンバ．
木本 正裕 Masahiro KIMOTO
株式会社ビジネスサーチテクノロジ先端技術開発センター