

# Dynamic Data Integration Using Web Services

Fujun Zhu<sup>b</sup>, Mark Turner<sup>a</sup>, Ioannis Kotsiopoulos<sup>c</sup>, Keith Bennett<sup>b</sup>, Michelle Russell<sup>d</sup>,  
David Budgen<sup>a</sup>, Pearl Brereton<sup>a</sup>, John Keane<sup>c</sup>, Paul Layzell<sup>c</sup>, Michael Rigby<sup>d</sup> and Jie Xu<sup>e</sup>  
<sup>a</sup>*Department of Computing & Mathematics, Keele University, Staffordshire ST5 5BG, UK*  
<sup>b</sup>*Department of Computer Science, University of Durham, Durham DH1 3LE, UK*  
<sup>c</sup>*Department of Computation, UMIST, Manchester M60 1QD, UK*  
<sup>d</sup>*Centre for Health Planning & Management, Keele University, Staffordshire ST5 5BG, UK*  
<sup>e</sup>*School of Computing, University of Leeds, Leeds LS2 9JT, UK*  
[fujun.zhu@durham.ac.uk](mailto:fujun.zhu@durham.ac.uk)

## Abstract

*We address the problem of large-scale data integration, where the data sources are unknown at design time, are from autonomous organisations, and may evolve. Experiments are described involving a demonstrator system in the field of health services data integration within the UK. Current web services technology has been used extensively and largely successfully in these distributed prototype systems. The work shows that web services provide a good infrastructure layer, but integration demands a higher level “broker” architectural layer; the paper identifies eight specific requirements for such an architecture that have emerged from the experiments, derived from an analysis of shortcomings which are collectively due to the static nature of the initial prototype. The way in which these are being met in the current version in order to achieve a more dynamic integration is described.*

## 1. Introduction

Effective communication within and between organisations is critical to their success. Current business requires a single global integrated enterprise system, which can provide the latest, unified information to both employees and business partners and meet business relationship changes such as organisation re-engineering, enterprise emergence etc.

However, traditional data integration approaches such as federated schema systems [01, 02, 03] and data warehouses [04, 02] fail to meet the requirements of a constantly changing environment (user requirement changes, business environment change and technical environment change etc.). In addition, successive maintenance operations on software systems tend to make the software system increasingly hard and expensive to maintain [05, 07].

In this paper we argue that the concept *software as a service (SaaS)* [05,06] and *late binding mechanism* under investigation by the Pennine Research Group<sup>1</sup> provide a possible approach for organisations to share resources in a constantly changing environment by integrating their applications and systems. This provides an overall context for the research described here.

Therefore, a *service-oriented data integration architecture (SODIA)* has been proposed to provide a dynamically unified view of data *on demand* from various autonomous, heterogeneous and distributed data sources. Service providers publish their data sources as *data access services (a kind of Web service, which allows for the provision of a data intensive service)*, which may then be discovered, bound at the time they are needed and disengaged after use. Hence the changes such as organization structures, backend data sources, data structures and semantics could be managed dynamically and potentially reduce the maintenance cost. To achieve this, those data access services should be semantically described and discovered.

We have built the first prototype in the EPSRC-funded IBHIS (Integration Broker for Heterogeneous Information Sources) project. This project references the UK’s National Health Service as an example of a large complex service based environment. Current web services [08] technology has been used extensively and largely successfully in these distributed prototype systems. The work shows that web services provide a good infrastructure layer, but integration demands a higher level “broker” architectural layer; the paper identifies eight specific requirements for such an architecture that have emerged from the experiments, derived from an analysis of shortcomings which are collectively due to the static nature of the initial prototype.

---

<sup>1</sup> The Pennine Group is the name of a collaborative team of software engineering researchers from the Universities of Keele, UMIST and Durham; it was formed in 1996. See [www.service-oriented.com](http://www.service-oriented.com)

The paper is organized as follows. We start section 2 by describing an example application; Section 3 discusses related work; and Section 4 introduces our *SODIA* conceptual model. In Section 5, we introduce our first prototype in which we combined federated schema [01] and web services [08]. Section 6 discusses the lessons we learned from the first prototype. Section 7 describes our current research on semantic interoperability, access control and change management in the second prototype to achieve more dynamic data integration. Finally, we conclude in Section 8.

## 2. Application scenario

### 2.1 The healthcare domain

UK Health and Social Care is a domain where autonomous agents are expected to deliver seamless care within a large, complex, and changing organizational context. However, the current NHS too often still resembles an archipelago of isolated information islands. Information systems in the NHS are distributed, heterogeneous, and autonomously administrated. Information is independently entered at different sites, which bear responsibility for the security and privacy of this data. Also the organisational structure and the data sources frequently change. Data sources are:

- 1) *Autonomous*: [01] the data sources exhibit design and implementation autonomy, communication autonomy, execution autonomy
- 2) *Heterogeneous*: Heterogeneity can appear in nine ways such as: naming, relational structure, value, semantic, data model, timing, system, syntax, transaction and security. [09, 10].
- 3) *Distributed*: the individual data sources are distributed across different organizations and sites rather than situated on the same host.
- 4) *The need for change because of business drivers*. For example, NHS authority policy changes will lead to change in the individual information systems and thus also within the data sources: the data structure, the security mechanisms etc.

These issues require that data sources be *late bound* rather than *statically bound (early bound)* to the integrated system. Late binding at run time to the data sources would allow the user to acquire a unified integrated view of the data *on demand* and would also allow the individual data sources to evolve.

### 2.2 An example

Let us consider a case in an accident and emergency department in UK Health Care. Mr Keane fainted when he travelled from Manchester to Durham. He had previously lived in Leeds, Newcastle and London before moving to Manchester. Therefore, his treatment history

data are stored in different systems at different sites. To treat him properly, the doctor in this department needs to know all the information about Mr. Keane. He needs to acquire an integrated view of the data about Mr. Keane from autonomous, heterogenous and distributed data sources and these data sources may change before integration, for example, the security policy could change at one of the sites.

## 3. Related work

**Federated Database Systems**: these comprise an integrated collection of fully-featured autonomous databases, in which the component administrators maintain control over their local systems, but cooperate with the federation by supporting global operations [01, 02]. The key limitation of this approach is that applications have to explicitly specify the data sources in a federated query. This means that the applications must be changed when new data sources are added. Each data source must also be explicitly registered to the federated DBMS, along with its wrapper. [03].

**Data warehousing**: a warehouse is a centralized repository of information extracted from multiple data sources. It is a static copy of the integration data and cannot be updated [03]. This centralized approach cannot meet a changing environment [04] for example, an environment in which information changes rapidly, information sources change rapidly, vast amounts of data come from large numbers of data sources or clients have unpredictable needs.

## 4. Service-Oriented Data Integration

### 4.1 Service-based software

System change is inevitable, expensive and very hard [07]. Both Bennett [05, 06] and Ghezzi [11] suggest that traditional static bound (early bound) supply-side systems cannot meet the needs of continually changing business environments. Therefore, the concept “Software as a Service (SaaS) was proposed by the Pennine Research Group, in which *services* [05, 12] are composed out of smaller ones (and so on recursively), procured and paid for on demand. This solution offers the potential for users to create, compose and assemble a service by bringing together a number of suppliers to meet their needs at a specific point in time. The approach proposed by providers such as IBM [13] and Microsoft focuses primarily on technical solutions. The Pennine approach uses a market led approach. Vendors offering the required services will be rewarded; those not doing so will be punished in terms of financial reward.

The central technical issue for this solution is very late binding (and disengagement), at the point of the execution of a system.

The general scheme for service-based software [05,06,13] is: publish, find and bind. We need to be able to find and bind services at the moment of need, and then unbind (disengage) as soon as the service is finished. Current work in *Web services* [08] does bring binding closer to execution, allowing an application or user to find and execute information services, such as *airline reservations and credit card validations*. Data handling and underlying protocols are automated through current web services technologies such as WSDL (Web Service Description Language), UDDI (Universal Discovery, Description and Integration), DAML-S [15], and SOAP (Simple Object Access Protocol)[14]. However, the key limitation of current approaches is that they lack fully dynamic discovery and binding processes in a large-scale, open, and ever-changing computational environment[16]. Current web services standards provide very limited support in mechanizing service recognition, service configuration and combination (i.e. realizing complex workflows and business logics with web services), service comparison and automated negotiation [17]. Furthermore we have found that system integration demands a higher level “broker” architectural layer.

## 4.2 Service-Oriented Data Integration Architecture (SODIA)

In order to provide the end-user with a unified view of data on demand from autonomous heterogeneous data sources, we propose a *Service-Oriented Data Integration Architecture (SODIA)* (Figure 1), which is based on the idea of *SaaS*[05, 06] and *dynamic (late) binding*.[05, 18] It is this architecture that we plan to employ within the final prototype IBHIS broker.

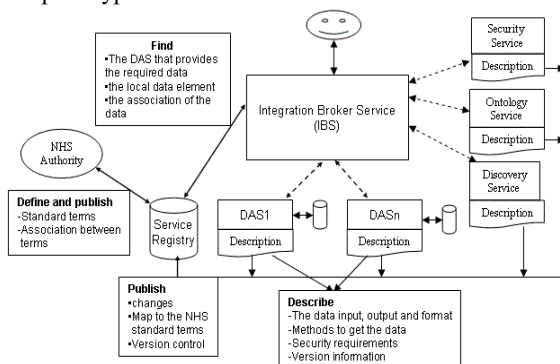


Figure 1. Service-oriented data integration architecture

An explanation of the architecture is as follows:

- 1) We assume that all the elements such as business content (vocabulary), constant (measure, postcode etc), datatypes and intent are defined or described with an ontology-based approach by the NHS authority within the NHS domain. This ensures that

all data elements are semantically correct and can be interpreted correctly by service requesters and providers.

- 2) Service providers (including data service providers) implement services and describe them using service description languages such as WSDL and DAML-S[15]. All services may be implemented using different languages on different platforms.
- 3) Service providers publish the service description file into a service registry, such as UDDI, and map the local data elements to the standard terms defined by NHS authority.
- 4) Services are then located using the service registry and the service implementation is invoked via SOAP. When an end user wants to acquire integrated data they look for the UDDI registry and find a suitable *Integration Broker Service (IBS)* that satisfies the requirements. When the IBS is invoked, it dynamically finds the sub services (data and functional services), and then binds to them appropriately.
- 5) The discovery service is used to discover and bind to service implementations at run time. When there is more than one service providing the same function, it can be used to choose one service based on the user’s requirements.
- 6) The ontology service is used to provide semantic transformation for different data items.
- 7) The security service is used to authenticate the user and control the data items a given user can access.
- 8) The Integration Broker Service (IBS) is a composed service, which integrates different data access services (DASs) and functional services such as the Discovery Service (DS), security service, and the ontology service in order to provide the end-user with an integrated uniform view of the data. The IBS could also compose a negotiation service to perform negotiation with the available data sources.
- 9) A *Data access service (DAS)* is a variation on the typical (Web) service as it is more data intensive and is designed to expose data as a service. Data service providers implement the DAS, which may query multiple, heterogeneous data sources. Alternatively, different DASs may query the same data source but produce different data outputs. The data service providers in the NHS domain must describe the data input, output and data format, the methods to acquire the data, the security requirements to use this service, and data service version.
- 10) When data service providers publish the DAS description file into the registry, they publish the DAS metadata and ontology.
- 11) For change management, DAS providers may provide users with different versions of the DAS. The service consumer can decide which one better meets

their requirements, or choose to bind to the latest version by default.

- 12) There is no integrated schema; data is integrated on the fly.

The next section describes the form of the *initial* prototype broker as completed in the summer of 2003, including details of the technologies used and the methods we have employed to expose a number of sample healthcare data sources as services.

## 5. The first prototype

The aim of the IBHIS project is to develop a series of prototype information brokers of increasing complexity. Each subsequent version of the prototype will realise more of the ongoing SaaS research, such as dynamic composition and binding, gradually extending the degree to which the broker is based around the SODIA architecture, as illustrated by Figure 1.

The following section describes the implementation of the first prototype, which incorporates an agreed sub-set of the service-oriented elements.

### 5.1 System architecture

The underlying architecture of the first prototype is based around an amalgamation of the available web service protocols with the concepts of a Federated Database System (FDBS). A detailed breakdown of the IBHIS architecture in relation to FDBS can be found in [18,29].

By its very nature, a major part of the IBHIS project is concerned with the seamless accessing and integration of many forms of complex data. This is different from a great deal of other service-based projects as by and large they are concerned with exposing *software* as a set of services and not *data*. Therefore, one of our research questions was how to realise traditional database structures within a service-oriented architecture, especially given the limitations of the available protocols. It was with this in mind that we chose to employ a traditionally database-oriented concept – that of the federated schema – but realise it within a service-based implementation. We use web services to enable the provision of data on demand whilst keeping the underlying data sources autonomous. This architecture allows us to provide *data* as a service (DaaS) as opposed to *software*.

### 5.2 Service elements within IBHIS

Figure 2 illustrates how, conceptually, the first IBHIS prototype is made up of several component ‘services’.

**Access Rules Service (ARS):** The ARS is responsible for the user authentication and data access authorisation. Within the Health and Social Care domain there are a

number of defined roles each with their own data access rights. Much research has been conducted into applying Role Based Access Control (RBAC) within the Health domain [19], with the conclusion that RBAC alone is too inflexible for health care applications. Therefore, a more complex set of access rules were developed in conjunction with a collaborative health service user (Solihull Primary Care Trust), incorporating individual user rights with roles.

**Federated Schema Service (FSS):** The FSS maintains the Federated Schema and all of the mappings between the local Export Schemas and the global Federated Schema. The FSS is consulted by the Federated Query Service during query decomposition. The Federated Schema and the corresponding mappings to the Export Schemas are created during broker set-up.

**Federated Query Service (FQS) :** The FQS contains two sub-modules: 1)The Query Decomposer decomposes the Federated Query into a set of local queries; this is done in consultation with the FSS. 2)The Query Integrator receives the set of local results from the Data Access Services and integrates them into a Federated Record. The FQS sends the Federated Query and the Federated Record to the Audit Service.

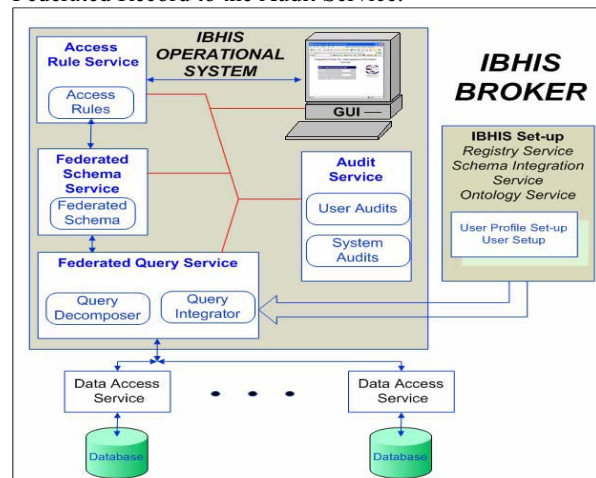


Figure 2. Conceptual architecture of Prototype 1

**Audit Service (AS) :**The AS will contain two sub-modules that will keep track of every action of IBHIS that needs to be recreated or audited in the future, including user (login date and time, IP address and sequence of queries executed) and system audit details (data source registration and user setup).

**Data Access Services (DAS):** In the first prototype, DASs are conceptually the same as described in the SODIA architecture (section 4). The description of a DAS must allow the consumer to discover:

- The data input, output and its format
- The domain and functionality related to the data
- The security requirements for using the service

- Other non-functional characteristics, including quality of service, and cost

The service providers then publish the description file into the service registry so that a consuming service, in our case the FQS, may discover them. Essentially, the DAS is used to provide a transparent, unified way to access distributed, autonomous heterogeneous data sources.

Table 1, below, illustrates how the conceptual services implemented in the first prototype can be mapped onto the SODIA architecture (Figure 1).

SODIA	Prototype 1	Notes
IBS (integration broker service)	FQS	The FQS is used for integrating data only, but IBS is a broker service that will call other services if necessary.
Data Access Service (DAS)	DAS	Similar, but in prototype 1 we do not use semantic descriptions.
Security Service	ARS	ARS is imposed at the beginning of the session rather than on a per-use basis of the DAS.
Ontology Service	FSS	In prototype 1, it is not an individual service, it is part of the FSS
Discovery Service	FSS	Included within the FSS
Service registry	FSS	Implemented as a mappings database in prototype 1
	Audit service	Not implemented in prototype 1

Table 1. Mapping prototype 1 to the SODIA

### 5.3 The implementation

We employed industry commercial developments to seed our first prototype development. Three distributed databases (two IBM DB2 and one MySQL) in three sites. (Durham, UMIST and Keele) using real health care data (but with no confidentiality issues). IBM Application Developer 5.1, Websphere Application server, java and J2EE were used as development tools. For message style, RPC was used rather than document style.

## 6. Lessons learned

### 1) Integration architecture

In the first prototype, federated schema were employed, built on three data access services at three sites. This approach to integration is static because of the nature of the FDBMS architecture. The mapping between local schemas and the global integrated schema was done manually, which was time consuming, error-prone and not

scalable. In the second prototype, very late binding will be employed in which the available data exposed by DASs will be semantically described, published to a semantic service registry, discovered and bound at run time. Data will be mapped and integrated dynamically. The table below provides a summary of the key differences between these architectural forms.

Issue	Web Service Federation	Service-Oriented Architecture
Structure	Federation resembling NHS forms	Loosely-coupled architecture enabling dynamic integration
Metadata	Database schemas only	Uses domain-specific ontologies
Semantics	Resolved by administrator	Machine-readable presentation
Scalability	Fixed number of databases	Dynamically extensible
Mappings	Created off-line, static mechanism	Decentralized and dealt with by the service provider

Table 2. Web Service Federation vs. SODIA

### 2) Web services protocols

We employed standard web service protocols in the first prototype. The use of SOAP was successful. RPC messaging was used to provide efficient performance. However, the nature of RPC required that the client side and the server side are tightly coupled and was not adaptable to change. For example, when using RPC messaging, if a service interface changes the client side must be changed accordingly. In addition, the RPC messaging requires that there must be a proxy at the client side for each service. For a large complex integration system, the problem of managing and maintaining thousands of proxies would render the system unusable. Therefore, wherever possible a general interface should be designed. This is the approach being taken for the development of our second prototype.

### 3) Web services vs. Service-oriented approach

Bennett[05, 06] and Ghezzi[11] suggest that the service-oriented approach and late binding mechanisms have the potential to ease large scale system development and reduce the maintenance effort. However, due to limitations of the FDBMS approach and RPC messaging in the first prototype it was necessary to change the federated query service (FQS) whenever new services were added to the system. In the second prototype, we are employing document-style messaging and building an integration engine service to compose the different DASs dynamically.

### 4) Service development environment

Websphere Application Developer 5.1 provided a successful IDE to create Java classes, and wrap those classes into Web services. A number of wizards were

available to help the developer to generate the WSDL files, client proxies and sample client code. All those tools ease the development of single web services. However, it does not support the generation of semantic service descriptions. Therefore, there is no guarantee that the terms and conditions of the service could be interpreted correctly, and as a result there is also no guarantee that the requested service be discovered and executed correctly.

#### 5) Service Registry UDDI

We used a relational database as a virtual service registry in the first prototype because even the recently released version 3.0 of UDDI does not allow us to specify sufficient non-functional information, such as output data sets or service security requirements. UDDI is designed primarily for human use rather than for a computer. Furthermore, it does not support rich semantics. Therefore, an adequate service semantic description could not be published into the current UDDI registry and services could not be discovered dynamically because of the lack of support for semantics. In the second prototype system we are working on extending the current UDDI registry through the use of ontologies and a semantic description language, such as DAML-S.

#### 6) Ontology

Ontologies [20] describe the meaning of terms used in a particular domain, based on semantic relationships observed among these terms. They are the key for semantic service description and dynamic service discovery. In the first prototype, we used an integrated schema as the global ontology. Domain ontology will be used in the second prototype to describe services, terms and conditions semantically thus enable the services to be dynamically discovered and bound.

#### 7) Service description

WSDL can be used to describe the service functions, methods and bindings. XML schema could also be embedded into WSDL to describe the returned data sets. However, it is not sufficient to describe the non-functional attributes such as security, versioning, quality of service, and cost. WSDL also lacks support for semantically describing the *terms and conditions* [05] and the functionality of the service. DAML-S [15] provides a semantic description of the service to some extent, but does not support the key non-functional attributes such as security. WS-Policy[21] allows for the description of the security policy. In the second prototype, DAML-S and WS-Policy will be employed for our services description. The key challenge is to find a unified approach to describe both the functional attributes and non-functional attributes of the service.

#### 8) Access control

RBAC [22] along with identities was chosen as the conceptual model for the first prototype as the health and

social care domain is familiar with the concept of roles and it appears to closely mirror the professional hierarchy. The existence of a federated schema allowed for the creation of a centralised policy that states which attributes from the schema each user and/or role is authorised to view. After authentication the user is assigned to a role, and the access control service then uses the policy to create a 'role-view' of the federated schema.

In terms of our plans for a service-oriented architecture, access control within the first prototype has a number of limitations. Firstly, the DASs are not autonomous and are essentially owned by IBHIS. As a result the IBHIS broker is able to bypass the access control in the local data sources and thus avoid the problems inherent in heterogeneous cross-domain access control. Also, the creation of a static, centralised policy is tailored to work with our Web service federation architecture and suffers from similar extensibility problems. In a completely dynamic autonomous environment the creation of such a policy would not be possible. Also, access control within the Healthcare domain depends upon other aspects over and above role and identity, for example time, and data content.

### 7. Proposed improvements for the second prototype

At present, we are working on the second prototype, where dynamic data integration will be supported. We are tackling the three research challenges:

#### 1) Change management

Change is inevitable. Therefore, the changes within the organization and outside should be managed.

**Internal changes (within the organisation)** refer to the changes that occur within organisations. Those changes include: data, data structure, constraints, permissions or rules, data model and semantics. To avoid the need for the external developer to generate a proxy for each data access service, we propose to employ the meta-database approach [23] whereby service providers describe the data access service semantically using the instance of the domain ontology. The meta-database approach ensures that any changes from the backend data sources will be managed within the meta-database through mappings.

**External changes (outside the organisation)** refer to the changes made by third party service providers. In this case, the service consumers have to manage the possible changes. In the IBHIS project, the Integration broker service (IBS) should have the capability to understand service descriptions, discover the services dynamically and invoke them. In the second prototype, we aim to semantically describe the service using a domain ontology and extend UDDI to act as a semantic service registry.

## 2) Access Control

The second prototype must enable efficient access control across a number of heterogeneous, distributed, autonomous data sources where little information is known before run-time. The OASIS project [24] is a certificate based implementation model that deals with many of the problems relating to distributed access control. However, it was not designed for a dynamic environment and therefore the decision was made to use OASIS but to extend it with service-based elements.

Along with the functional aspects of the service, the DAS semantic description also describes access control requirements, including details of the mappings between the conceptual model of the IBHIS security domain and that of the data providers. This is provided in the form of a service-level agreement (SLA). The description also provides a link to the data source access control policy, which is maintained autonomously by the data provider. The Security service of the SODIA architecture also includes an access control component that provides dynamic enforcement of policies across the DASs. This component dynamically locates service descriptions from the semantic registry and uses the description to locate and enforce, firstly, the mappings from the SLA and secondly, the access control policy for the DAS.

We have created a new conceptual access control model which extends RBAC and combines features from several existing models. Those features combined include dynamic healthcare 'teams' inherent in the Team-based Access Control model [25], the identity level permissions and logged override facilities of the Tees Confidentiality Model [26] and the contextual parameter enhancements of the OASIS model.

## 3) Semantic interoperability

The semantic interoperability problem has drawn the attention of the research community during the last few years. Recent progress in the field of web services and the semantic web has paved the way for dealing with semantics in a more consistent way based on widely accept standards. Information brokering could benefit from these recent advancements as we can now envisage hybrid brokering architectures built upon web services.

However, web services are relying on protocols that mainly deal with interoperability in terms of system, structural and syntactic heterogeneity but fail to address any semantic issues. Data retrieved from the invocation of web services need to be semantically understood by system as well as their relationship with other pieces of data. The introduction of domain ontologies provides the context for defining data objects while description languages on top of ontologies could be the basis for a well defined framework for dealing with data semantics in this context.

The second prototype of the IBHIS broker is based on a service oriented data integration architecture (SODIA)

which promises loose coupling among the interacting software components. SODIA aims to achieve loose coupling and interoperability by employing two constraints; firstly components are described by simple and ubiquitous interfaces and secondly descriptive messages are constrained by an extensible schema delivered through interface [28]. These constraints are realised within the SODIA by widely accepted internet protocols (HTTP, FTP, and SMTP) and de facto standards (XML, SOAP and WSDL). However, these protocols and standards fail to deal with similarities, differences and relationships between objects exposed by web services. Moreover, querying capabilities are limited and the existing registries (UDDI) can only deal with keyword searches.

The semantic interoperability problem is tackled within the second IBHIS prototype at three fronts:

**Data Description:** The DAS is a web service that exposes some metadata about the underlying data in the form of a WSDL description. However, since WSDL cannot provide semantic descriptions, there is a need to provide another layer of metadata about the exposed metadata. This time, the *semantic metadata* will use terms from the domain ontology in order to make the semantic description universally understood.

**Semantic Registry:** At the moment, W3C has adopted UDDI as the default registry for web services. This registry is based on a classification system (tModels) which currently does not support semantic descriptions. Our suggestion is to enhance UDDI, firstly to store semantic descriptions and secondly to accept semantic queries instead of keyword searches.

**Query Engine:** The role of the query engine is to formulate queries based on the domain ontology that will be sent to the semantic registry. It will also create an efficient execution plan according to resources available.

## 8. Conclusions

Communication within and between organisations is critical to their success. It becomes more important for an organization to get an integrated view of data from autonomous, heterogeneous, distributed and continually changing data sources. However, traditional integration approaches such as federated schema and data warehouse are failing to meet those changing environments.

We have illustrated how current web services technology has been used extensively and largely successfully in our first prototype system, and proposed a new service-oriented data integration architecture (SODIA) to deal with dynamic discovery of data sources. The work shows that web services provide a good infrastructure layer, but integration demands a higher level "broker" architectural layer. It also shows that the current web services technology has yet to mature [27].

The key limitation of current approaches is that they lack fully dynamic discovery and binding processes in a large-scale, open, and ever-changing computational environment [16].

The requirements for such a service-oriented data integration architecture have been described by analyzing the first IBHIS prototype. Three research challenges (change management, access control and semantic interoperability) have been described to achieve a more dynamic data integration broker for the second IBHIS prototype. This is currently under construction.

## Acknowledgements

We would like to thank all members of the IBHIS project and the Pennine Group for their valuable contributions to this paper. This work was funded by an award from the Distributed Information Management programme of the EPSRC.

## References:

- [01] A. Sheth and J. Larson. "Federated database systems for managing distributed, heterogeneous and autonomous databases". *ACM Computing Surveys*, 22, 3, pp. 183-236, 1990.
- [02] R. Jakobovits, "Integrating Autonomous Heterogeneous Information Sources", 1997
- [03] V. Raman, I. Narang, C. Crone, L. Haas, S. Malaika, T. Mukai, D. Wolfson and C. Baru "Data Access and Management Services on Grid", 2002
- [04] E. Franconi, "Introduction to Data Warehousing",
- [05] P.J.Layzell, K.H.Bennett, D.Budgen, P.Brereton, L.A.Macaulay, M.Munro, "Service-Based Software: The Future for Flexible Software" *Asia-Pacific Software Engineering Conference*, 5-8 December 2000, ISBN: 0-7695-0915-0 pp. 214-222
- [06] Bennett K. H., Gold N. E., Munro M., Xu J., Layzell P. J., Budgen D., Brereton O. P. and Mehandjiev N. "Prototype Implementations of an Architectural Model for Service-Based Flexible Software". in *Proc. Thirty-Fifth Hawaii International Conference on System Sciences (HICSS-35)*, edited by Ralph H. Sprague, Jr. p.76, 2002, Published by IEEE Computer Society, CA, ISBN 0-7695-1435-9.
- [07] M.M.Lehman, "Laws of Software Evolution Revisited", Proceedings 5th European Workshop on Software Process Technology, Nancy, France. Oct 1996, pp. 108-124.
- [08] Web Services Activity. <http://www.w3.org/2002/ws/>
- [09] H.T. El-Khatib, M.H. Williams, L.M. MacKinnon, D.H. Marwick "Using a Distributed Approach to Retrieve and Integrate Information from Heterogeneous Distributed Databases" *Computer Journal* Vol 45 No 4 (2002) pp 381-394
- [10] A.P. Sheth. "Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics". M. F. Goodchild, M. J. Egenhofer, R. Fegeas, and C. A. Kottman (eds.) *Interoperating Geographic Information Systems*, Kluwer, 1998
- [11] C. Ghezzi, "Ubiquitous, Decentralized, and Evolving Software: Challenges for Software Engineering". *ICGT 2002*: 1-5
- [12] C.Lovelock, S.Vandermerwe, B.Lewis, "Services Marketing", Prentice Hall Europe, 1996
- [13] H. Kreger, "Web Services Conceptual Architecture (WSCA 1.0)", IBM, May 2001
- [14] SOAP version 1.2. <http://www.w3.org/TR/SOAP/>
- [15] DAML-S <http://www.daml.org/services/>
- [16] E. Michael Maximilien, Munindar P. Singh, "Reputation and Endorsement for Web Services", *SIGecom Exchanges*, 3(1):24-31, Winter 2002, ACM Special Interest Group on E-Commerce.
- [17] D. Fensel and C. Bussler, "The Web Service Modeling Framework WSMF", Vrije Universiteit Amsterdam (VU) and Oracle Corporation, 2002
- [18] I. Kotsiopoulos, J. Keane, M. Turner, P.J. Layzell and F.Zhu, "IBHIS: Integration Broker for Heterogeneous Information Sources", *Proceedings of COMPSAC'04*, IEEE Computer Society Press.
- [19] J. Poole, J. Barkley, K. Brady, A. Cincotta, and W. Salamon, "Distributed Communication Methods and Role-Based Access Control for Use in Health Care Applications," *Second Annual CHIN Summit* 1995.
- [20] N. Guarino, "Formal ontology and information systems". In: Proceedings of the international conference on formal ontologies in information systems (FOIS), Trento, Italy, 6-8 June 1998, pp 3-15. IOS Press, Amsterdam
- [21] Web Services Policy Framework (WSPolicy) <http://www-106.ibm.com/developerworks/library/ws-polfam/>
- [22] R.S. Sandhu, E. Coyne, H. Feinstein, and C. Youman, "Role-Based Access Control Models", *IEEE Computer*, 29 (2), 1996, pp 38-47
- [23] W. Cheung and C. Hsu: "The Model-Assisted Global Query System for Multiple Databases in Distributed Enterprises". *ACM Trans. Inf. Syst.* 14(4): 421-470 (1996)
- [24] J. Bacon, and K. Moody, "Toward open, secure, widely distributed services", *CACM*, 45(6), June 2002, pp. 59-64
- [25] C. Georgiadis, I. Mavridis, G. Pangalos and R. Thomas, "Flexible Team-based Access Control Using Contexts", in *Proceedings of the 6th ACM Symposium on Access Control Models and Technologies (SACMAT 2001)*, ACM SIGSAC, Chantilly, VA, U.S.A, May 2001
- [26] J.J. Longstaff, M.A. Lockyer, and J. Nicholas, "The Tees Confidentiality Model: an authorisation model for identities and roles", *ACM SACMAT 2003*, Como, Italy.
- [27] M. Turner, D. Budgen and O.P. Brereton, "Turning Software into a Service", *IEEE Computer*, 36(10) October 2003
- [28] O'Reilly, "What is Service-Oriented Architecture", December, 2003
- [29] Turner M, Zhu F, Kotsiopoulos I, Russell M, Budgen D, Bennett K, Brereton OP, Keane J, Layzell PJ & Rigby MJ (2004) Using Web Service Technologies to create an Information Broker, *accepted for ICSE 2004*.