A Broker Architecture for Integrating Data Using a Web Services Environment

Subtext: real application
integration of heterogenous autonomous evolving databases

Pennine Research Group

www.service-oriented.com

Montrose
Scotland
Dundee
Glasgow
Berwick Upon Tweed
Prestwick
Hawick
M74
Londonderry
Newcastle Upon Tyne
OCEAN/SEA
Stranraer
Carlisle
Workington
Middlesbrough
Northern Ireland
Scarborough
Dundalk
N17
Leeds
Clew Bay
N5
Drogheda
Kingston Upon Hull
Dublin
Ireland
Anglesey
North Sea
Galway Bay
N8
A470
Kilkenny
Norwich
Cardigan Bay
A49
Waterford
A487
Wales
Ipswich
Cork
Great Britain
Clacton
Cardiff
London
Bristol Channel
Celtic Sea
Exeter
Torbay
Strait Of Dover
Belgium
100%

Authors

- ***Durham***: K. H. Bennett, J. Xu, F. Zhu,
- ***Keele***: O.P. Brereton, D. Budgen, , M. Turner
- ***UMIST***: J. Keane, I. Kotsiopoulos, O. Almilaji, J.C. Chen, A. Owrak, N.E. Gold, P.J. Layzell,
- Keith.bennett@durham.ac.uk

Overview

- We present results of a real experiment, using distributed data access system, using Websphere, J2EE
- First we explain hypothesis

Roadmap

- Assumptions
- Hypothesis
  - Study
  - Cost of ownership
  - Evolution and change
- Experimental system
- Results from V1
- Conclusions, next stage

Assumptions: Distinctive Domains

- Systems Domain
  - Well defined boundaries and requirements
- Business Domain
  - Emergent Organisations
    - "Organisations in a state of continual process change, never arriving, always in transition"

D. Truex, R.Baskeville and H.Klein, "Growing Systems in Emergent Organizations", Comm.ACM, Vol.42, No.8, August 1999

UMIST KEELE University of Durham

Our key hypothesis

- One or more services are configured to meet a specific set of requirements at a point in time, executed, and disengaged.
- "**Software as a Service**", or ultra-late binding
- How did we establish this?

1. Research in Evolution

- 60-80% of lifetime costs of software relate to change. Of that, most is due to requirements change
- Evolution technologies
  - Program comprehension, re-engineering, reverse engineering and design recovery
- Maintainability not solved: needs new approach; web services require change.

UMIST KEELE University University of Durham

Evolution

- In our experience, it seems very difficult to “bolt on” a good evolution or maintenance solution to an existing development method (e.g. formal methods)
- It needs to be addressed from the outset
- We want to explore ultra-late binding as a new approach to maintainability

UMIST KEELE University University of Durham

2. User study

- Necessary and sufficient
- Personalisation
- Adaptable/ self-adaptation
- Distribution and granularity
- Transparency
- CONCLUSION: cost of ownership is bad

P.Brereton, D.Budgen, K.Bennett, M.Munro, P.Layzell, L.Macaulay, D.Griffiths and C.Stannett, "The Future of Software: Defining the Research Agenda", Comm. ACM, Vol.42, No.12, December 1999

UMIST KEELE University University of Durham

The Vision

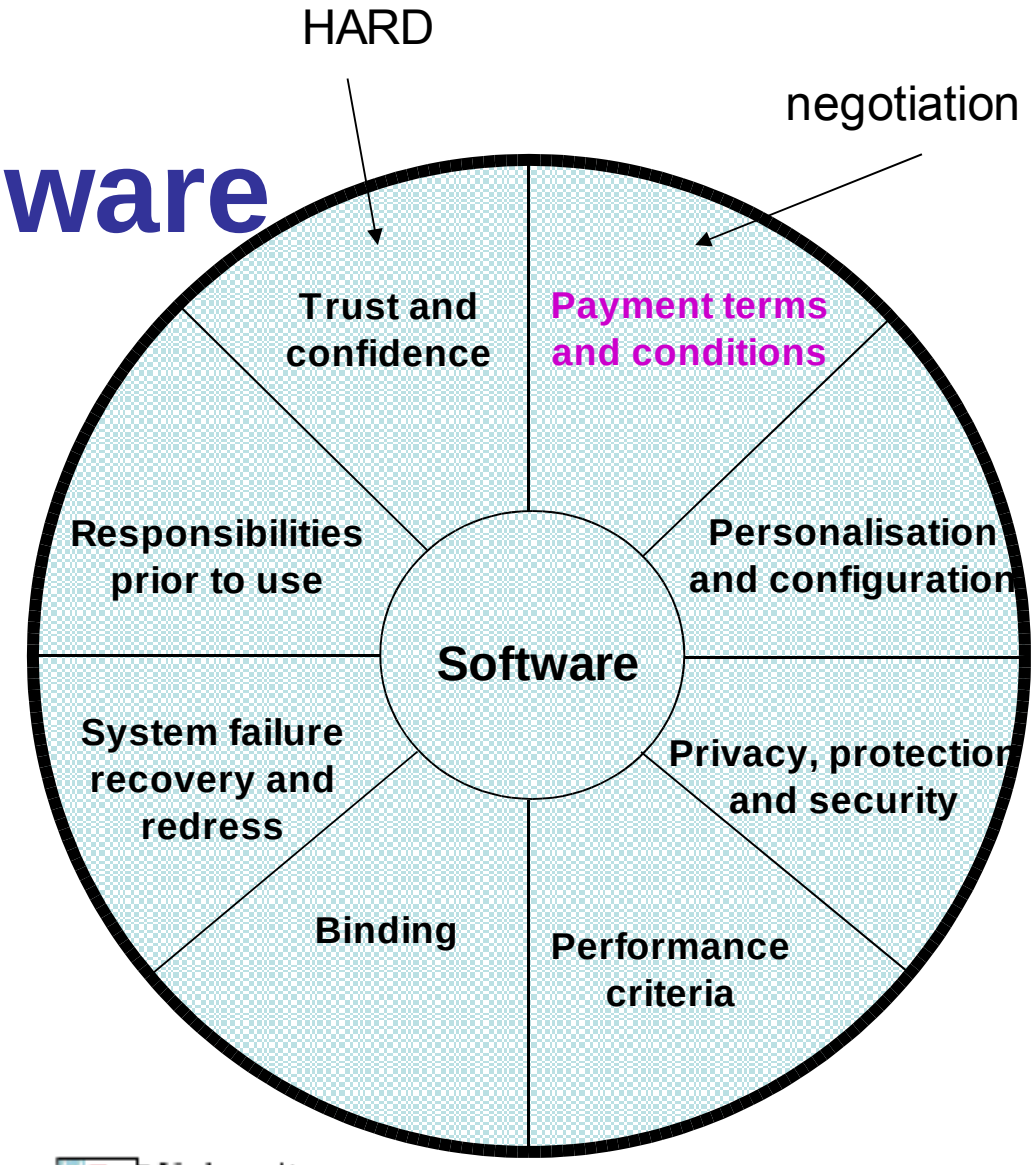
- Software moves from a **PRODUCT** to a **SERVICE**.
- A **SERVICE** is something you find, use **as and when needed** – and then discard.
- The **user** decides what services are needed, and the technology negotiates, agrees and implements their **binding**, which involves many non-technical attributes (trust, cost, redress..)

UMIST KEELE University of Durham

Ultra late binding

- Automatic binding at execute time requires non-functional attributes as well as functional attributes to be bound
- It is here that many of the challenges (not yet met by web services) apply

Serviceware



UMIST
KEELE UNIVERSITY
University of Durham

IBHIS - **I**nformation **B**roker for **H**eterogeneous **I**nformation **S**ystems

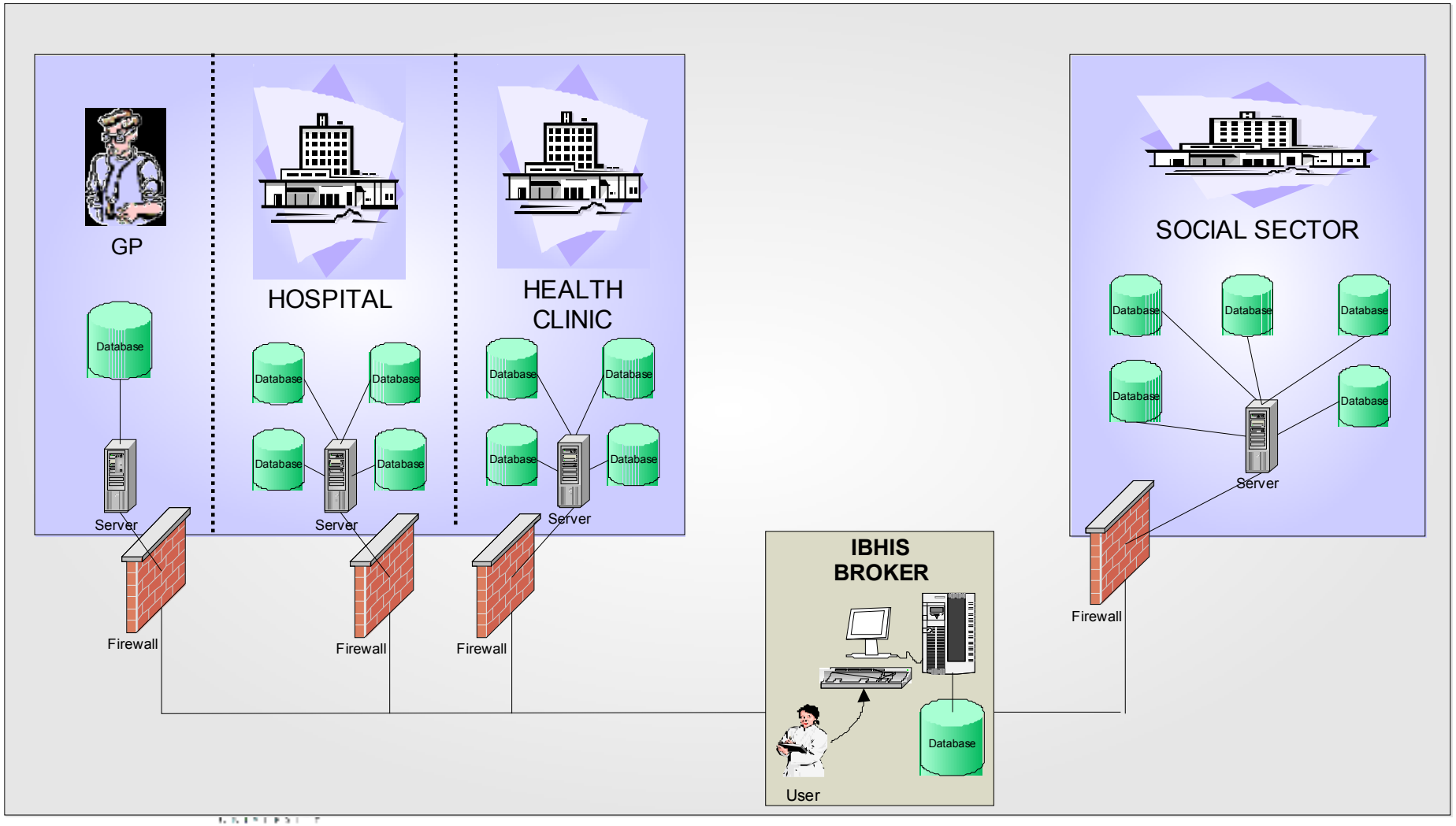
- Aim: To support decision making processes where information is drawn from a set of heterogeneous, autonomous agencies

- Domain: Health and Social Care

UMIST KEELE UNIVERSITY University of Durham

IBHIS Overview

In mental health, there are 58 different services



Example

- Patient centred care

  Disabled children with complex needs
  Looked after children
  Child protection
  Single assessment process
  Intermediate care
  Mental Health
  PRIVACY/SECURITY IS KEY

UMIST KEELE University University of Durham

Service/Broker Approach

- Strengths of this approach against fully integrated systems (currently in fashion)
  - Supports *multiple, independent* data sources
  - Handles *syntactic, semantic* and *system heterogeneity*
  - Deals with *globally distributed information*
  - A *pathway* towards discovery and access of new Information Resources with the minimum of human intervention

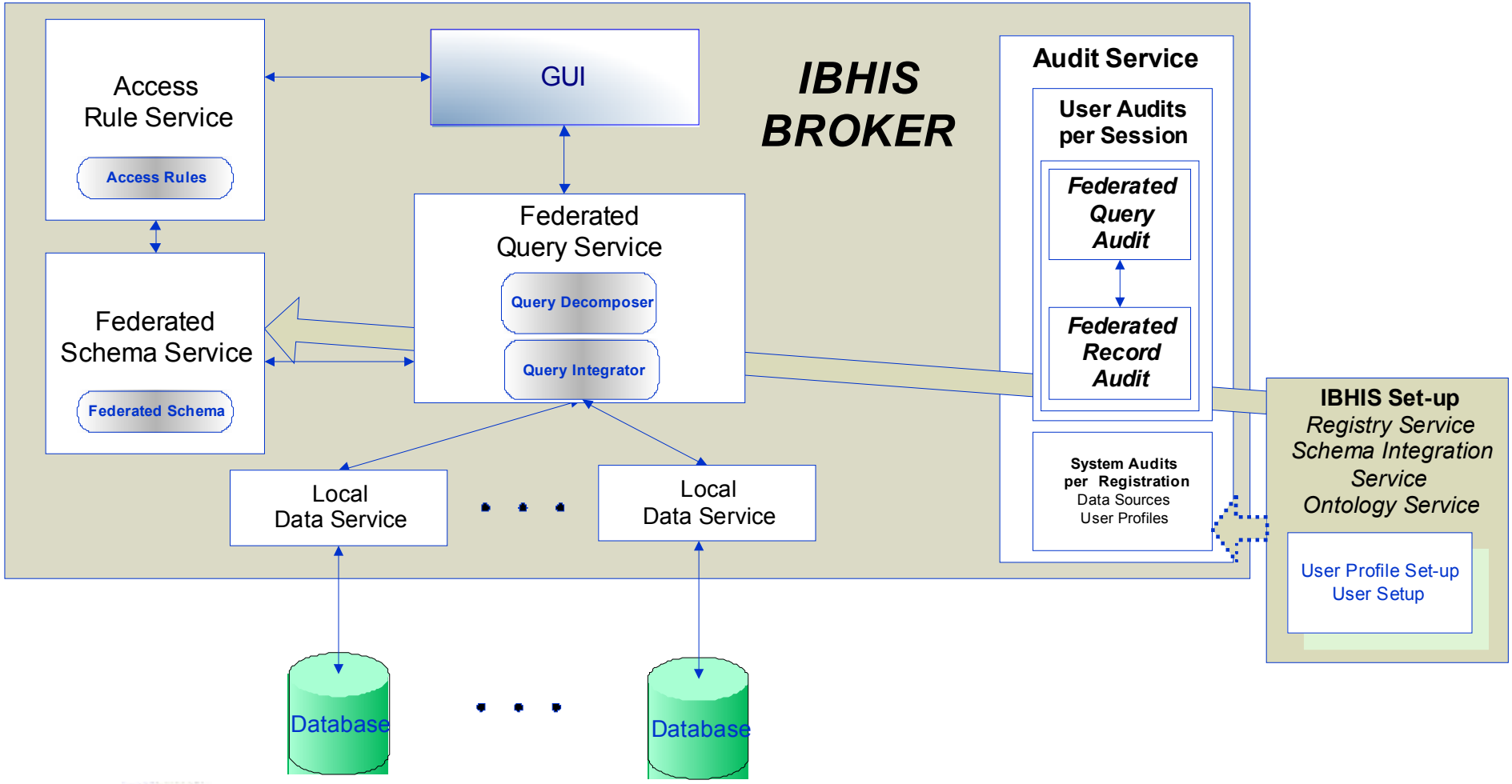
UMIST KEELE University University of Durham

Key Concepts

- Use of service-based architecture to:
  - Integrate information
  - Accommodate evolution of the broker
  - Ease adaptation to change in organisational structures and related data structure
- Exercise custodianship, authentication and ethical control of information
- Creating integrated, person-centred views of need and delivery whilst retaining professional responsibility

[Prototype Architecture]

- User queries IBHIS, IBHIS uses federated schema to interrogate ‘local’ data sources, and coalesces result
- Based on statically-bound set-up knowledge of data sources (bound in at initialisation time)
- Build federated schema from local schemas – manual
- User profile <role>, and user name
- All queries checked for access rule compliance
- All queries and results recorded with time-log
- Model allows for local data sources to themselves be federated systems or even adjoining IBHIS systems


Access
Rule Service
Access Rules
GUI
IBHIS
BROKER
Audit Service
User Audits
per Session
Federated
Query
Audit
Federated
Record
Audit
Federated
Query Service
Query Decomposer
Query Integrator
Federated
Schema Service
Federated Schema
Local
Data Service
Local
Data Service
System Audits
per Registration
Data Sources
User Profiles
IBHIS Set-up
Registry Service
Schema Integration
Service
Ontology Service
User Profile Set-up
User Setup
Database
Database

Results

- System implemented across 3 sites, using websphere, J2EE
- Used Oracle, DB2, mySQL
  - Basic patient information (Keele)
  - Treatment history (UMIST)
  - Further appointments (Durham)
- [Extensive work on domain analysis – see papers]

R1: Architecture

- V1 used a tightly coupled federated DBS, built at design time manually
- Service provided different views (schema) according to access rules
- Good for static data sources, and few of them; poor for changing schemas
- Problems really due to FDBS

R2: Service descriptions

- WSDL used for each web service – created at design time
- Adequate for V1, but not for general data sources from autonomous organisations
- We need to use an ontology approach.
- No non-functional meta data used.

UMIST KEELE University of Durham

R3: Message mechanism

- SOAP/RPC is used in prototype 1, which is tightly-coupled and signatures are static
- Document-based asynchronous message using XML should be employed, which will make integrated system easier to meet the changes of individual services owned by autonomous owner.

R4: Service registry

- We used a DB2 registry with a central manager
- Basically a statically set up directory of data items – locate and bind data sources at run time.

Web services

- Platform, language and implementation (and IDE) independence largely achieved (minor differences from IDEs).
- SOAP unifying XML data format was pivotal.
- Independence of service and service description (and thus call)

Websphere

- Good facilities for implement, test, deploy
- IDE had several time saving facilities
- Concurrent Versions (open source plug in for version control)
- Some early V5 problems

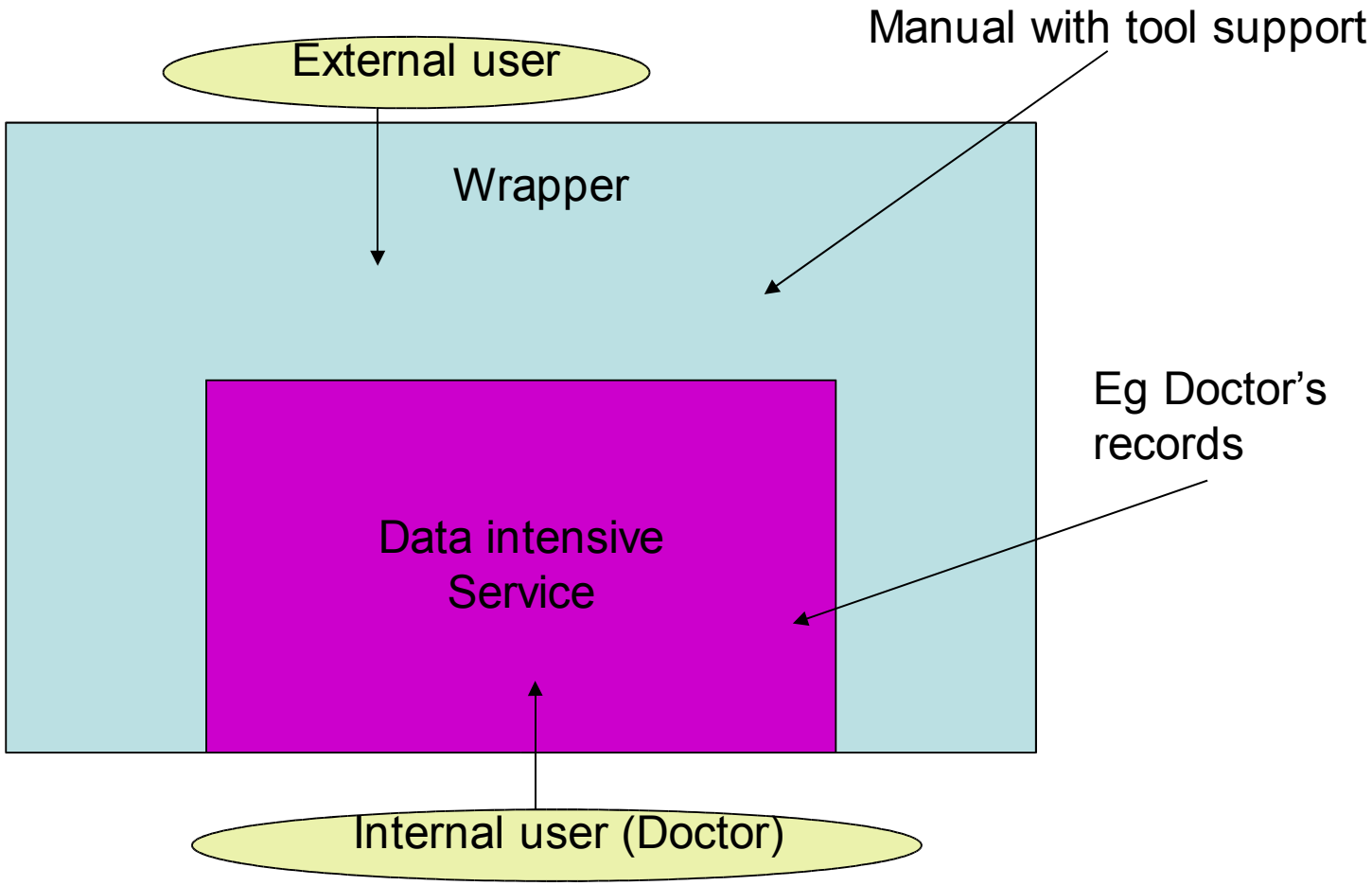
Summary

- V1 much too static
- V2 experiments with more dynamic architecture

Summary

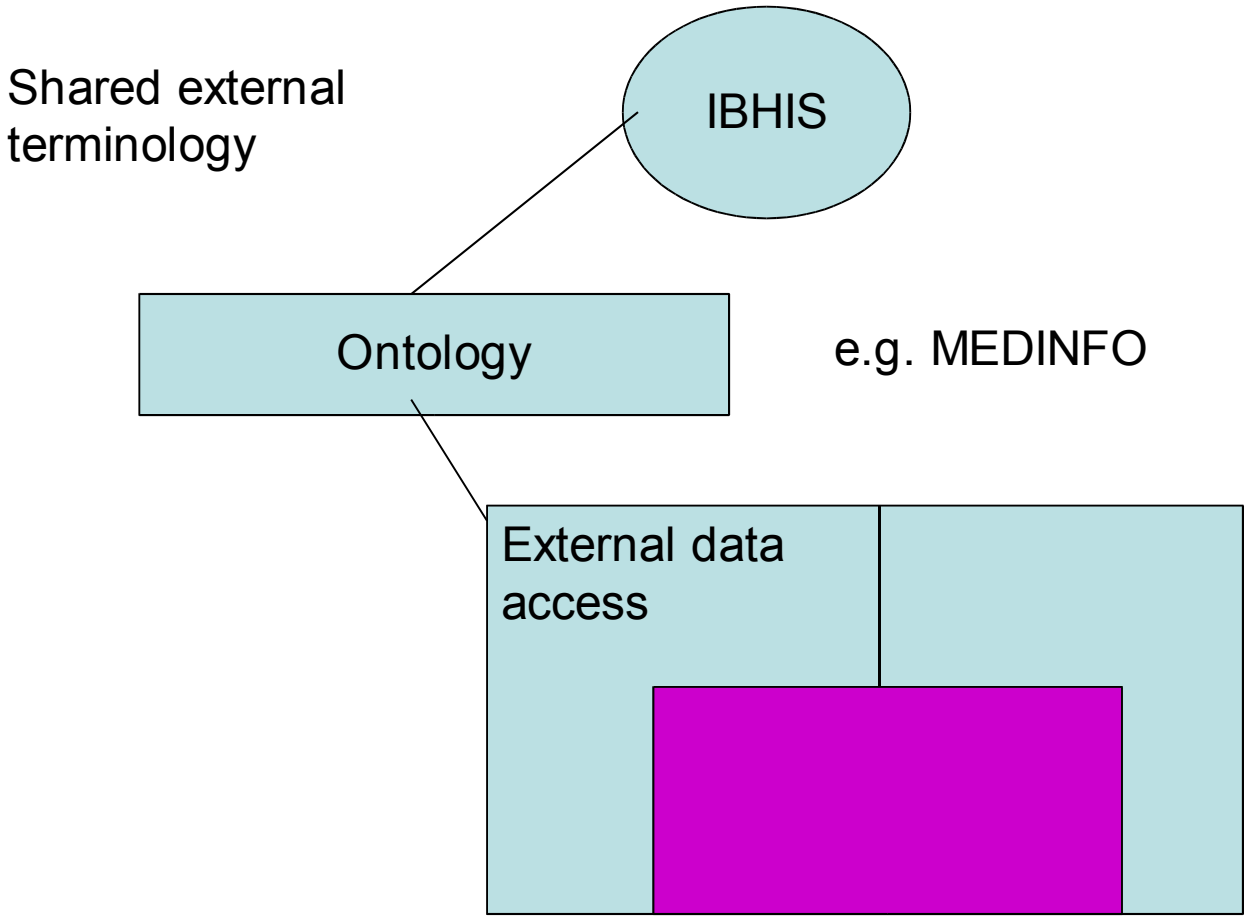
- We need much more work on a (data) web service description languages
- We need to exploit indirection and late binding to achieve evolution e.g. inclusion of new data sources.

V2 - wrap

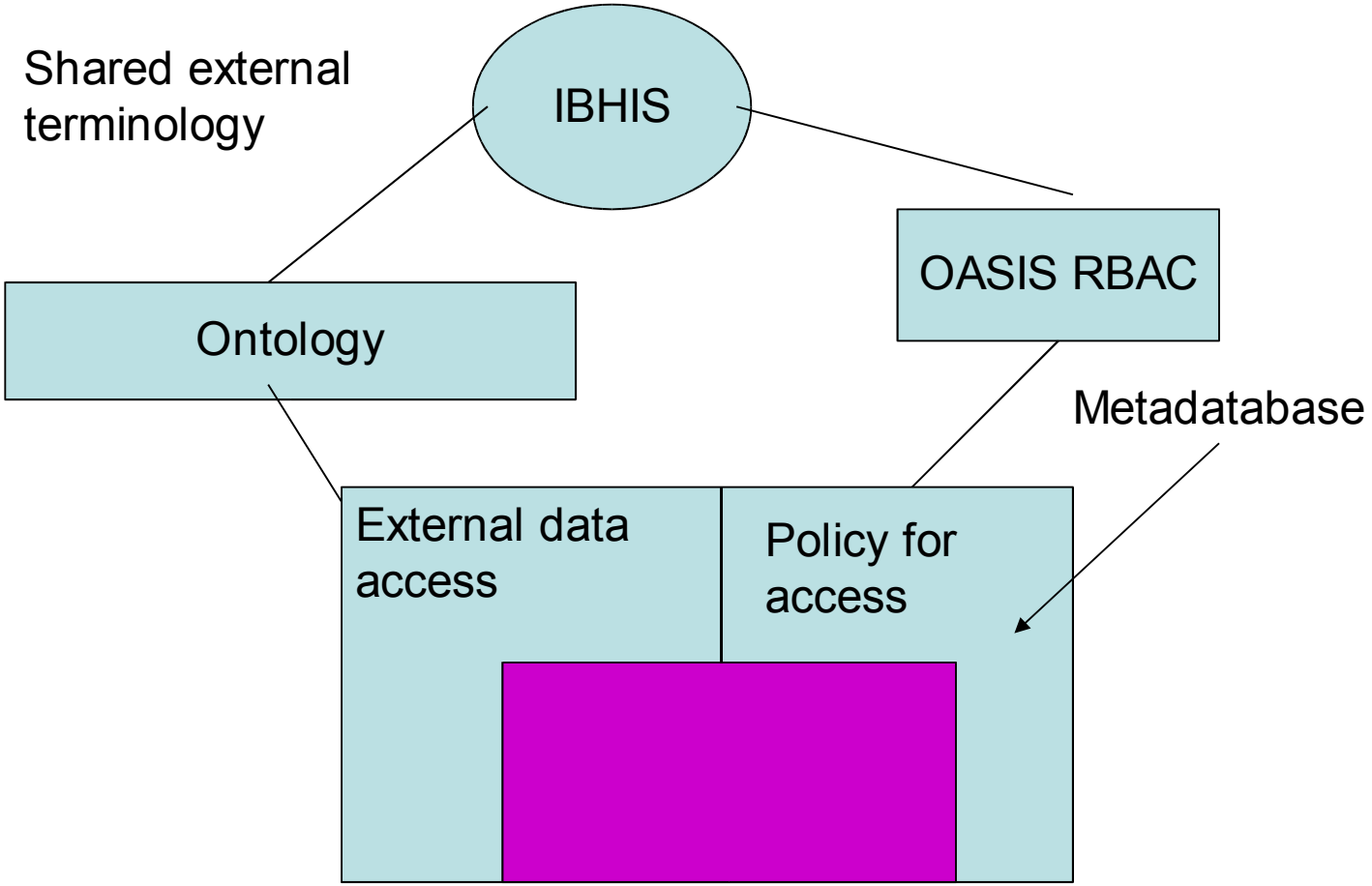


UMIST KEELE UNIVERSITY University of Durham

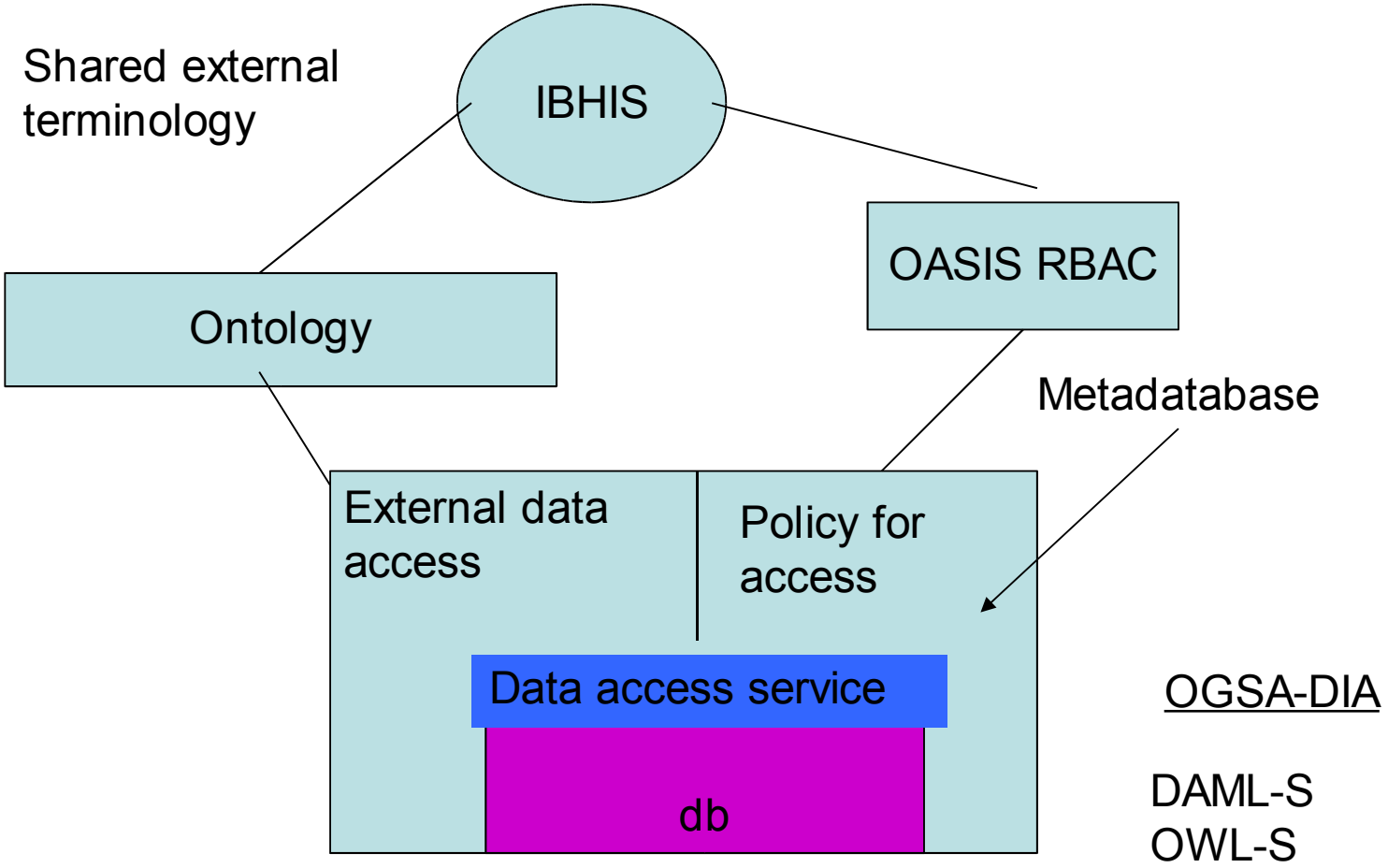
V2 integrate



V2 composition



V2 transform



Conclusion (the gap)

Service App layer (applications created “on demand” from smaller services

Service integration layer

Service transport layer (using Globus, J2EE etc)

V1 has made minor impact on middle layer
V2 is making a much bigger in-road

UMIST KEELE University of Durham

Finally

- Demand led computing is very different to supply side led
- Evolution is our real challenge – hardly solved yet: needs technical and business (market) contributions
- Evidence based engineering for results.

Acknowledgements

- To Colleagues in the Pennine Research Group
- To EPSRC for funding

Markets

- CS - likes to address very general search and match problems
- Real markets are often not open (eg kaizen)
- Supply chains
- Market will fill need if money to be made
- Compositions can be bought
- Use expertise of marketing and business school experts