ANE MIS/GIS PILOT SUPPORT PROGRAM

SYSTEM DESIGN

4/24/2006

This publication was produced for review by the United States Agency for International Development. It was prepared Booz Allen Hamilton.

Table of Contents

Table of Figures

1 Introduction

This document details the system architecture of the USAID ANE MIS/GIS Pilot Support program. From program inception, the Booz Allen team has stressed that its approach will emphasize Commercial Off The Shelf (COTS) products, integrated into a Service-oriented Architecture (SOA). Listed below are the vendors providing products that will be used in the MIS/GIS.

- BEA, including former Plumtree products
- BusinessObjects
- Cape Clear
- Chiliad
- EMC^2
- eSpatial
- ESRI
- Handysoft
- IBM
- Intergraph
- MapInfo
- MetaMatrix
- Microsoft
- Microstrategy
- Oracle
- Stellent

Reflecting the Service-oriented approach, we consider the system to be a set of interacting, loosely coupled services. The foundation of the architecture is a robust Data Model that is designed for reporting against geospatially-enabled data. Most services are implemented via a multi-tiered architecture, with a web-based client. A Portal will serve as a unified hosting environment for the interfaces of the various software packages, deployed across disparate systems. The middleware integration strategy emphasizes support of Web Services, and in particular, employs the use of an Enterprise Service Bus (ESB) as an integration platform.

The subsequent Demonstration phase of the program will implement the specified design. All software packages will be hosted in the Infoshare domain of the McLean Computing Center, more commonly referred to by its colloquial name, the SOA Sandbox.

2 Functional Service Areas

Based upon the MIS/GIS Needs Assessment, we have analyzed the pilot system requirements and identified the relevant *Functional Service Areas*. Each Functional Service Area addresses one or more Service Areas or Activities that were identified during Needs Assessment. Also, each Functional Service Area groups a set of related capabilities that the MIS/GIS should provide, and presents them as a loosely-coupled *service* to the system as a whole.

It is preferable for a service to provide an interface that supports Web Services standards, particularly Simple Object Access Protocol (SOAP) and Web Services Definition Language (WSDL). Support of Web Services standards facilitates integration, but is not strictly necessary in a Service-oriented Architecture. However, support of Web Services will weigh heavily when scoring products in the COTS evaluation process because it offers the most flexible and adaptable approach for the system, and avoids "vendor lock-out" for future COTS integration.

We have identified candidate software products, depicted in Figure 2.1, for each Functional Service Area, based on a variety of factors: first, availability of a suitable product is key. The SOA Sandbox hosts numerous products and use of a product that has already been installed, configured, and evaluated with favorable results will reduce time and effort to achieve USAID ANE's objectives. Second, vendor willingness to participate in the Pilot Support Program with no promise of the purchase of a license will also be important. These vendors were identified via the Vendor Summit, held late in 2005. The Booz Allen team engaged each vendor who showed a willingness to participate, and determined candidate software products based on vendor feedback and our own research. Finally, in one case, the Booz Allen team found that the needs of the MIS/GIS were best fulfilled by custom implementation.

Functional Service Area	Description	Service Areas/Activities Addressed	Candidate Software Products
Client Operating System	Self-explanatory	All User Interface Operations	Windows Client OS's
Document Management System	Repository for Unstructured Data	Data Collection Management	Documentum, Stellent
USAID Business Layer	Software Interface to USAID Business Data; includes User Interface and Data Access Objects	Data Collection Management, Field Mission Planning, Information Discovery, Investment Management, Knowledge Management, Program Implementation	Custom code, built with Java Server Pages, Struts and Hibernate Open Source Toolkits
Data Integration	Toolset(s) that transform mission data to USAID standard structured data	Reporting, Information Discovery	MetaMatrix, Oracle Warehouse Builder
Workflow Manager	Facilitates Workflow definition/execution	Workflow management	Bizflow, IBM BPM
User Interface Portal	Provides a Single user interface for multiple web-based applications	All User Interface Functions	Plumtree Portal, IBM WebSphere Portal
User Manager	Manages Users, Groups, and permissions	Activities with access restricted by role	Microsoft Active Directory
Phoenix Interface	Accesses Phoenix Data	Investment Management	Custom software
Search Engine	Searches Unstructured data	Information Discovery	Chiliad
Geospatial Toolset	Displays maps and features	Geospatial Visualization	ESRI suite, MapInfo, eSpatial, Intergraph

Middleware	Integration Platform, implemented by Enterprise Service Bus	All	CapeClear ESB, IBM WebSphere Process Server
Business Intelligence	Supports Report Creation, including dynamic queries and ad-hoc reports	Reporting	MicroStrategy, Business Objects
Gazetteer Service	Returns Geospatial Coordinates of supplied place name	Geospatial Visualization	NGA Gazetteer web service

Figure 2.1 MIS/GIS Functional Service Areas

Major Functional Service Areas and their interactions with other components are addressed in further detail in Section 8.

3 Data Repositories

The identified Data Repositories are given in Figure 3.1. Each repository falls under one of four categories: a custom repository, a repository maintained by a specific COTS package, a repository maintained by a host Operating System, or an external USAID repository.

Repository	Description	Associated Component
USAID Business Data	Holds all USAID data, adhering to unified structured format	USAID Business Layer, implemented in Oracle 9i
Portal Repository	Stores all Portal configuration data	Plumtree, Websphere Portal
Client File System	Self-explanatory	Windows client OS's
Document Repository	Holds document metadata for Document management	Documentum, Stellent
Workflow Repository	Repository for Workflow data	Bizflow, IBM BPM
Phoenix	USAID Financial data	Phoenix
PSIP/JAMS	Instrument data	PSIP/JAMS
Users Repository	User and Group Repository	Microsoft Active Directory
BI Repository	Holds Report and Report Template metadata for BI tools	MicroStrategy, Business Objects
Search Repository	Holds persistent search data (e.g. search agents)	Chiliad Search, Custom DB tables
Geospatial Data	Contains Base Map Data; USAID-specific feature data is contained in USAID Business Data	Base Map Data provided by Government agencies
Geospatial Repository	Holds configuration data for Geospatial Server/Middleware	ArcSDE, MapInfo, eSpatial, Intergraph

Figure 3.1 MIS/GIS Data Repositories

4 Data Model

The foundation of any data-centric application is the Data Model, and the MIS/GIS embraces this concept. The Data Model is extensive, but can be broken down into several major areas: Business Concepts, the Feature Model, Keywords, and Data Collection Forms. The data model diagrams appearing in this section visually depict the necessary relationships between data model elements, or entities. Formally, these diagrams are known as Entity-Relationship diagrams.

4.1 Rationale for Unified Data Model

In order to facilitate reporting, it is necessary to formulate a unified data model. To construct reports from disparate, unstructured data sources without defining a common data format is difficult and expensive. When extracting information from unstructured data sources, Business Intelligence and Data Integration tools benefit from being able to access the *intended* data format. Finally, the need to be able to input relevant data in a consistent fashion (identified during the Needs Assessment phase) requires the implementation of a data model in a relational database.

4.2 The Business Concepts

The Business Concepts aspect of the Data Model reflects a generic operational model of USAID missions generated from information collected and analyzed during the Needs Assessment phase. This model reconciles as best as possible the diversity of Mission approaches to operations and databases in an attempt to simplify and model the core business concepts used by the ANE Missions. The entities documented are the relevant USAID-specific Business Concepts that must be captured by the MIS/GIS. Refer to Figure 4.1 and Figure 4.2 on the following pages. The Business Concepts are intended to be compliant with the EIS Data Architecture.

At the heart of the data model are Missions, Strategic Objectives (SO), Intermediate Results Packages (IRP), Activities, Instruments, Implementing Partners, and Indicators. A Mission defines its country strategy in terms of Strategic Objectives. Strategic Objectives are composed of Intermediate Results Packages, which are in turn composed of Activities, which may, in turn, be composed of Projects. While multiple Projects can make up an Activity, we consider the Activity to be the basic unit of work. There are three reasons for this. One, not all Missions use the notion of a Project. Two, instruments apply to a single Activity, so the lowest level at which financial data may be reported is an Activity. Three, an Activity is the lowest level at which an Indicator may efficiently be applied if data is to be effectively aggregated. An Indicator may be associated with Program Components; the presence of this relationship indicates that the Indicator is a Common Indicator.

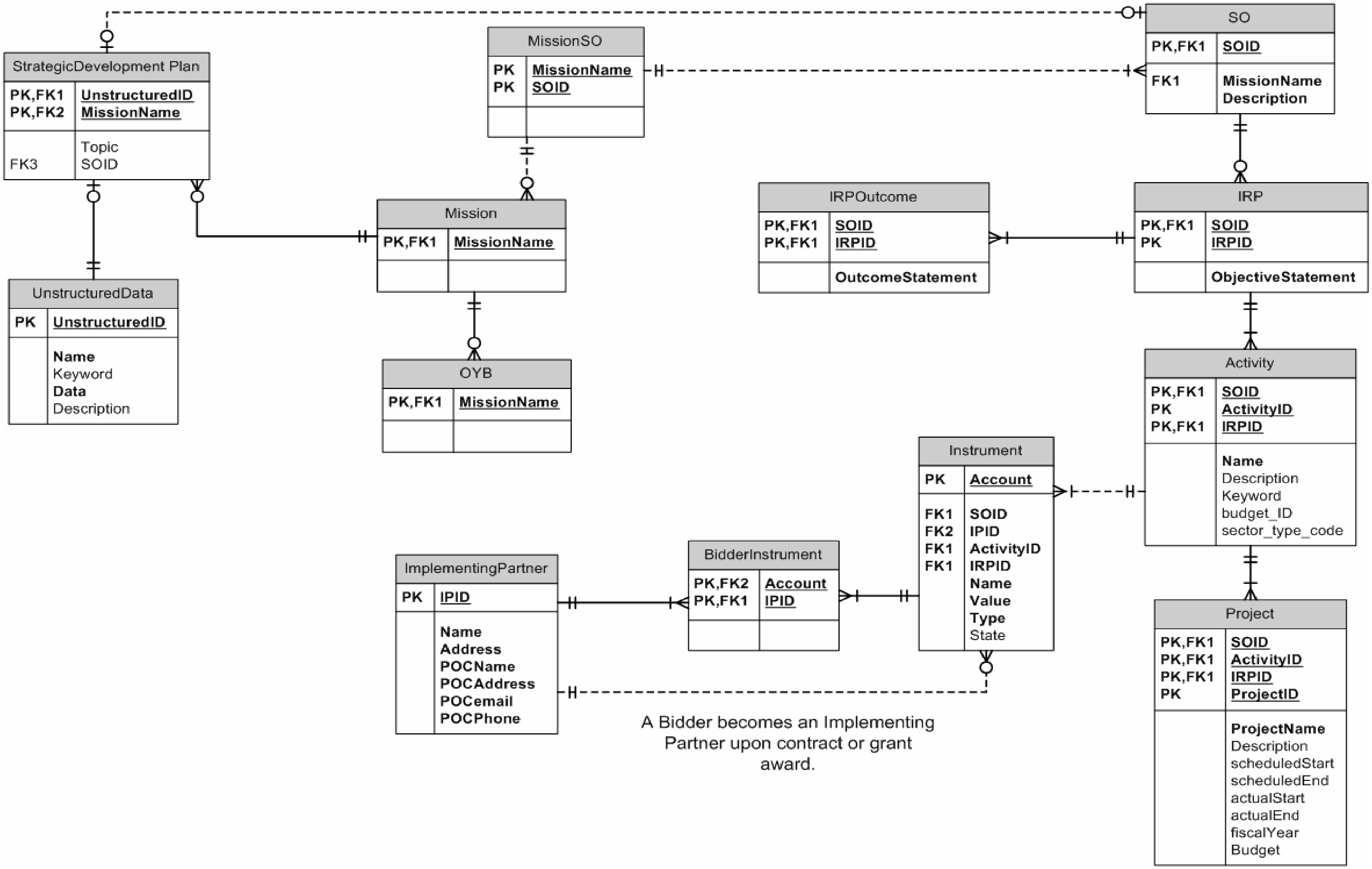

Figure 4.1 Business Concepts

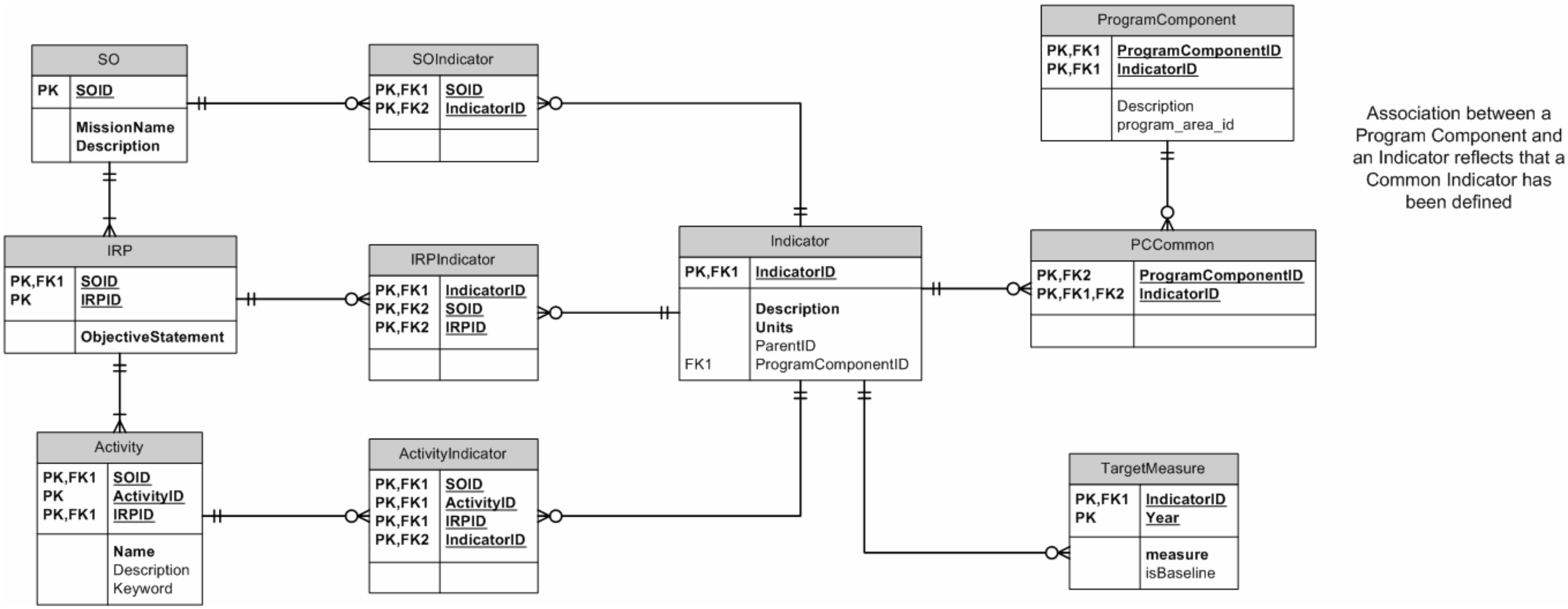

Figure 4.2 Indicators

4.3 The Feature Model

The Feature Model is depicted in Figure 4.4 and Figure 4.5 on the following pages.

4.3.1 What is a Feature?

Activities do not exist in isolation. Rather, Activities consist of a complex set of constituent elements and actions with interrelated results and impacts. For example, building three schools in Afghanistan is an example of the work breakdown of an Activity. Similarly, engaging in trade talks may be part of an Activity. In any case, it is the real-world concepts that drive the indications of performance. An example will help illustrate.

Assume we have an Activity contracted to an Implementing Partner, where the statement of work is to build three schools and train fifteen new teachers. In this case, a Performance Indicator might report the percentage increase of students going to school. Note that the same Performance Indicator could be used in many missions, for many related but distinct Activities—the Performance Indicator is not specific to the Activity. The key observation to make is that the metrics that ultimately indicate performance (e.g., number of new students attending) are specific to the core constituent elements associated with the Activity, namely, schools.

This leads to the notion of a *feature*, an abstraction of some real-world entity. A feature may be a physical structure, such as a school or a hospital. In other cases, a feature represents a more abstract concept, such as trade talks, policy dialogue, or a political boundary. A feature may or may not have an associated physical location. Schools and hospitals certainly have associated physical locations--they can be plotted on a map. In other cases, a feature is a more abstract concept that has no associated physical location, such as judicial independence.

Features provide a flexible way to associate constituent elements of a complex system, whether they are physical structures, actions, or other concepts, with USAID-specific business concepts, such as Activities and Strategic Objectives. Features can be categorized by *feature domain*, and a class of like features is referred to as a *feature type*. A feature is a specific instance of a feature type. All features that belong to a certain feature type contain the same attributes. For example, under the feature domain *education* is the feature type *school*. Each feature is a specific school, containing its particular values for the school feature type's attributes. Figure 4.3 below shows a set of instances of the school feature type—specific schools and their attribute values

School Name	Number attending	Year built	Grade level	X coord	Y coord
School1	40	2005	elementary	12354	12355
School2	1000	2001	secondary	12443	34233
School3	300	2004	elementary	45567	27655
School4	750	2002	secondary	23456	56788

Figure 4.3 Sample data for the school feature type

.

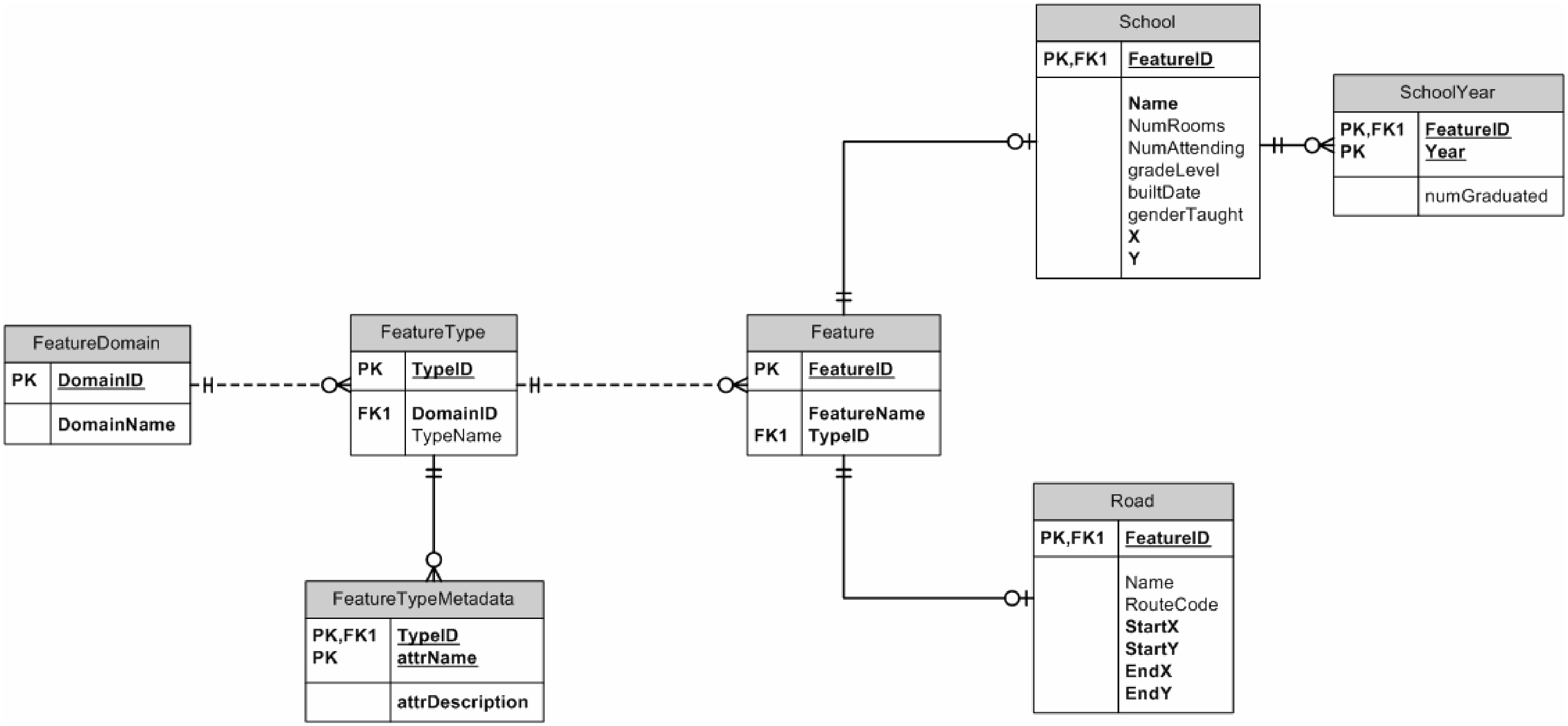

Figure 4.4 Features

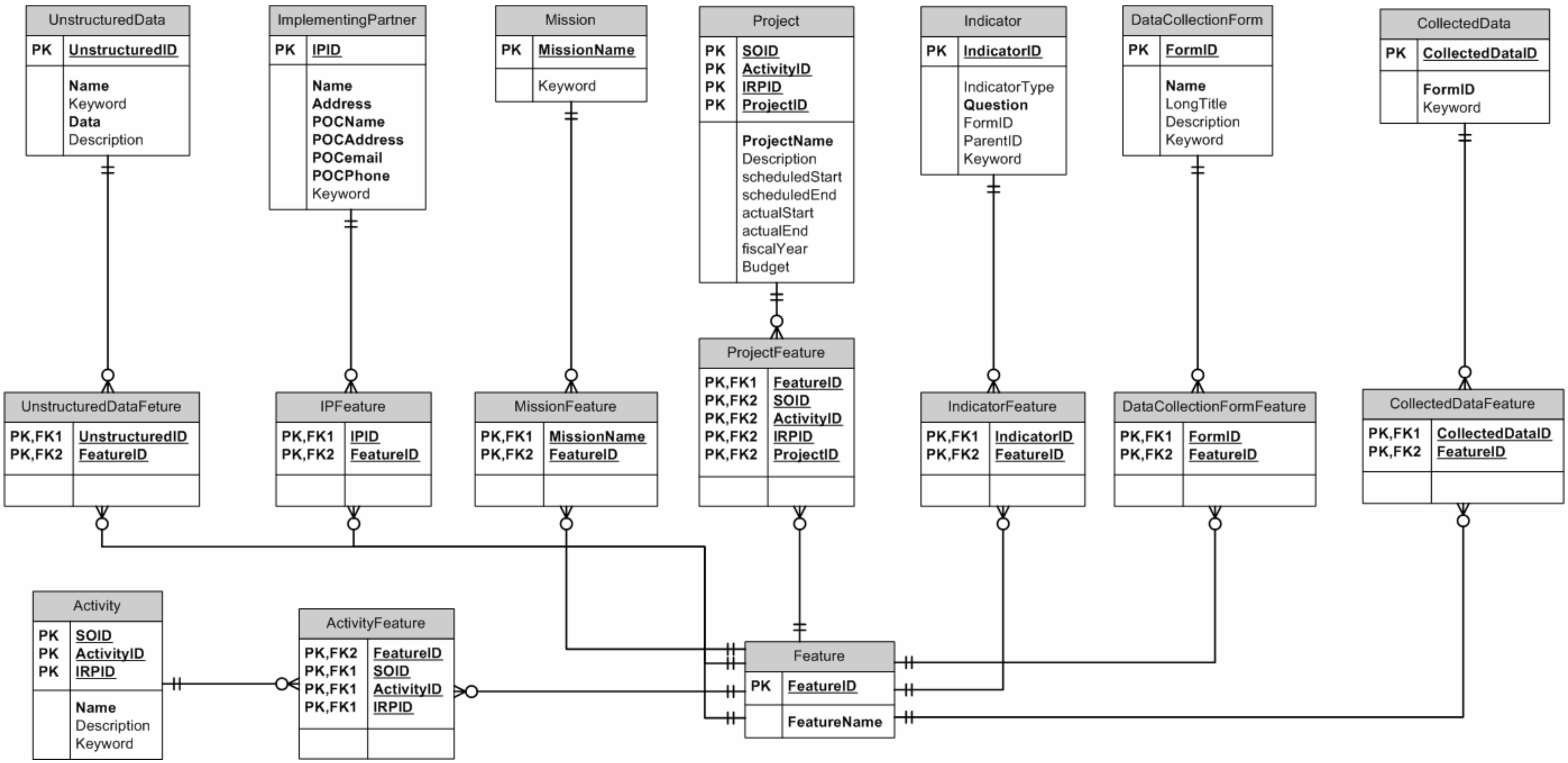

Figure 4.5 Feature Associations

4.3.2 Features and Report Generation

Below is an example element of a performance status report, giving an indicator and actual results.

I.R. 18.1 – Utility Services Enhanced

Performance Indicator: 18.1.a – Percent increase (relative to baseline) in capacity of selected wastewater utilities

Unit of Measure: Baseline (m^3/day); percent **Source:** Various contractors (please see section 3.2.1 for details)	**Fiscal Year**	**Planned (Cumulative)**	**Actual (Cumulative)**	**Met or Exceeded Target?**	
				Y	**N**
Comments: Breakdown of capacities increased per sector: • 154% increase in capacity for Water projects. • 320% increase in capacity for Wastewater projects.	1999 (Baseline)		1,828,842	N/A	
	2000	0	0	N/A	
	2001	13	6		N
	2002	15	41	Y	
	2003	53	168	Y	
	2004	**168**	**180**	**Y**	
	2005	199			
	2006	204			

Figure 4.6 Sample report element

The Indicator and its Planned values are set ahead of time, when the Intermediate Result Package is created. However, it is the metrics about features that ultimately determine the values of the *Actual* column of Figure 4.6. Most likely, some features that supply data to the actual column did not exist when the Intermediate Results Package was created. In this particular example, the feature type in question is wastewater treatment facilities, and the metric that feeds the results is capacity, in thousands (or millions) of m^3/day of wastewater treated. There might be many different features that feed the actual results, occurring within the scope of separate Activities, but with all Activities grouped under the given Intermediate Results Package. For example, one Activity might consist of building a number of new treatment facilities, while another Activity might entail upgrading equipment in existing facilities.

In more complex scenarios, the services of a Business Analyst, leveraging a Business Intelligence toolset, are invaluable, see Figure 4.7. This is especially true in cases where many different feature types are used in a single report element, or when the unit of measure for an Indicator does not coincide with the units of measure of supporting feature types. As sophisticated as some Business Intelligence tools are, they still require use by a knowledgeable user.

Figure 4.5 depicts all entities that may be associated with a feature. This allows features to be associated with other elements of the Data Model, in the same way that features are associated with Activities. As a simple example, recall that a political boundary can be a feature. Then, a Mission can be associated with a feature that represents the boundary of its host country. Effectively, the Mission record in the database has now been geospatially enabled.

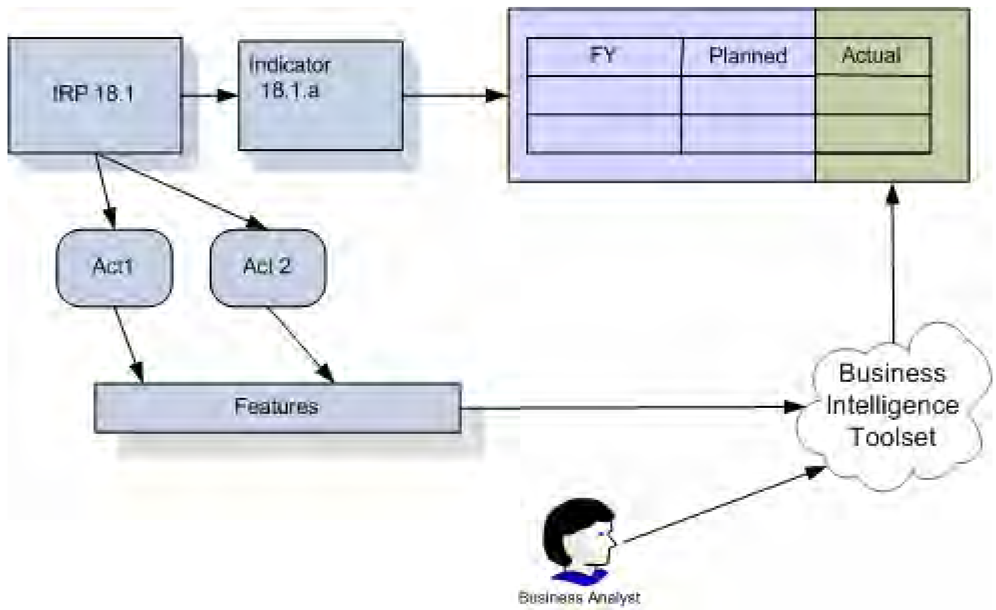

Figure 4.7 Feature data feeding reports

Finally, it is useful to note that a feature may have some temporal characteristics, in addition to its more customary, non-temporal data. The example given in the data model in Figure 4.4 shows data maintained in the *SchoolYear* table as ancillary data to *School*. In this example, the number of students who graduated in a particular school year is recorded.

4.3.3 Use of Features in Geospatial Toolsets

As mentioned, a feature may or may not have geospatial data, depending on the specific feature type. What must be mentioned at this point, however, is that *only* features will maintain geospatial data. Therefore, the use of features is essential to the concept of a spatially-enabled MIS (MIS/GIS) and to the functioning of candidate Geospatial products. The exact process to configure a Geospatial toolset to display maps that incorporate feature data is vendor product-specific.

4.3.4 Extending the Feature Model

Considering the wide variation in the types of development projects executed under the auspices of USAID, the number of possible feature types can become daunting very quickly. In order to keep the scope of the MIS/GIS pilot manageable, we have defined a set of feature types that we believe are commonly used by USAID; see Figure 4.8. The data model diagram in Figure 4.4 includes two examples of feature types, Roads and Schools. It is hoped that the pilot will serve as a test bed where commonly used feature types that are not included in our initial design can be identified rapidly.

It also must be noted that the data model will accommodate a growing set of feature types and feature domains. Adding a new feature type requires a new record to be created in the feature *type* table, while adding a new feature *domain* will require a new record to be created in the feature domain table. And lastly, the geospatial toolset must be reconfigured to support the new feature type, if it has geometry.

Feature Domain	Feature Type
Health	Hospital
	Clinic
Education	School
Utilities	Electrical Generation Plant
	Wastewater Treatment Plant
	Wells
Agriculture/Environment	Greenhouse gas emissions
	Biologically significant area
	Sustainable Timber
Communications	Radio Station
	TV Station
	Cell Phone Tower
Transportation	Road
	Rail
	Airport
	Port
Political	Boundary
	Place Name
	Law
	Democratic Organization
	Civil Society Organization
Economic	Micro-enterprise
	Enabling Policies
	Non-bank Financial Institution
Human Improvement	Trafficking in Persons
	orphanage

Figure 4.8 Feature Types Provided

4.4 Forms

One of the key requirements uncovered during the Needs Assessment phase is the capability to automate the data collection process. This is accomplished by sending out a Data Collection Form, which can be conceptualized as a survey. Each question in the survey is answered, and that data is captured in the *CollectedData* table (see Figure 4.9).

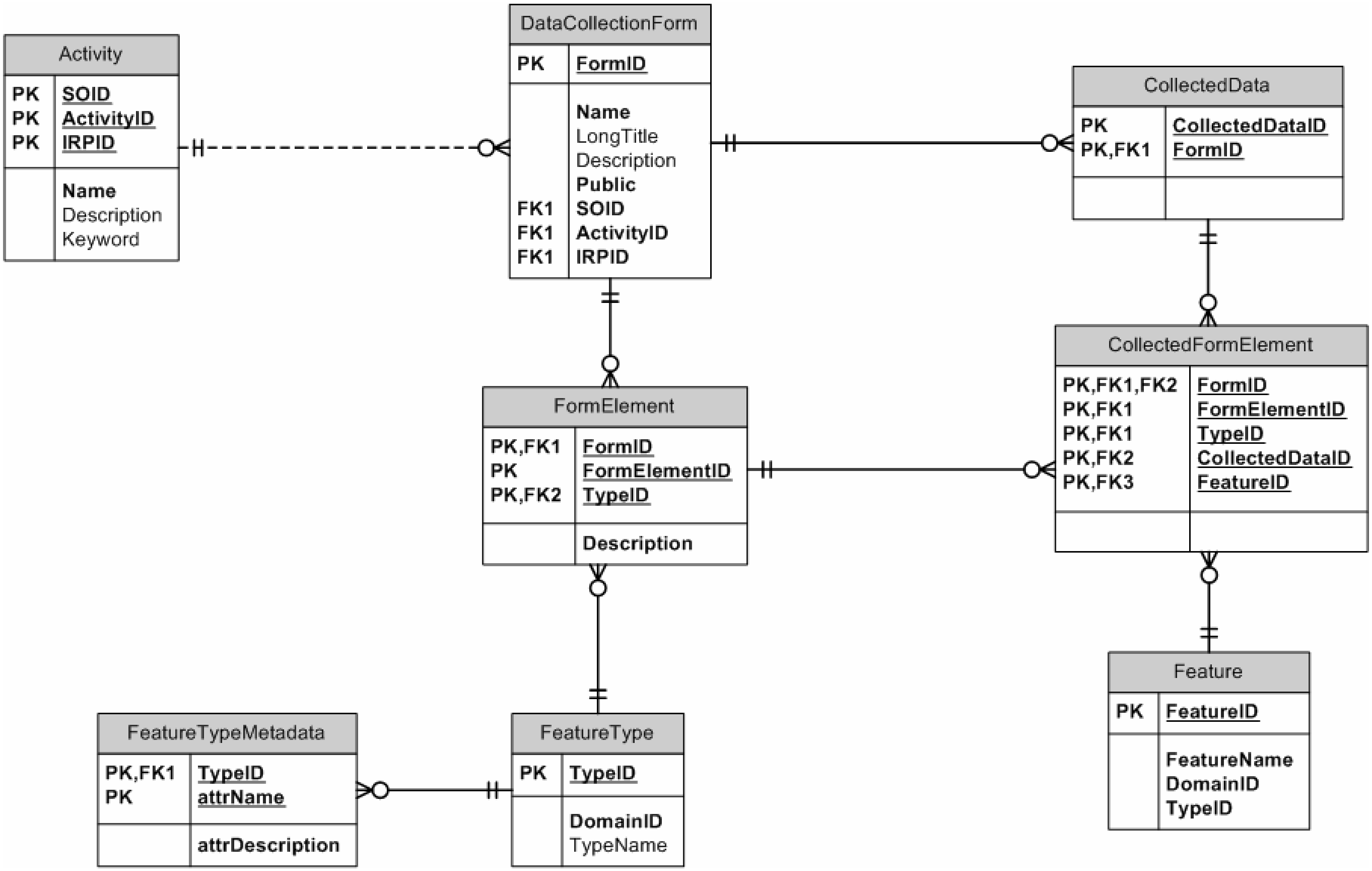

Figure 4.9 Data Collection Forms

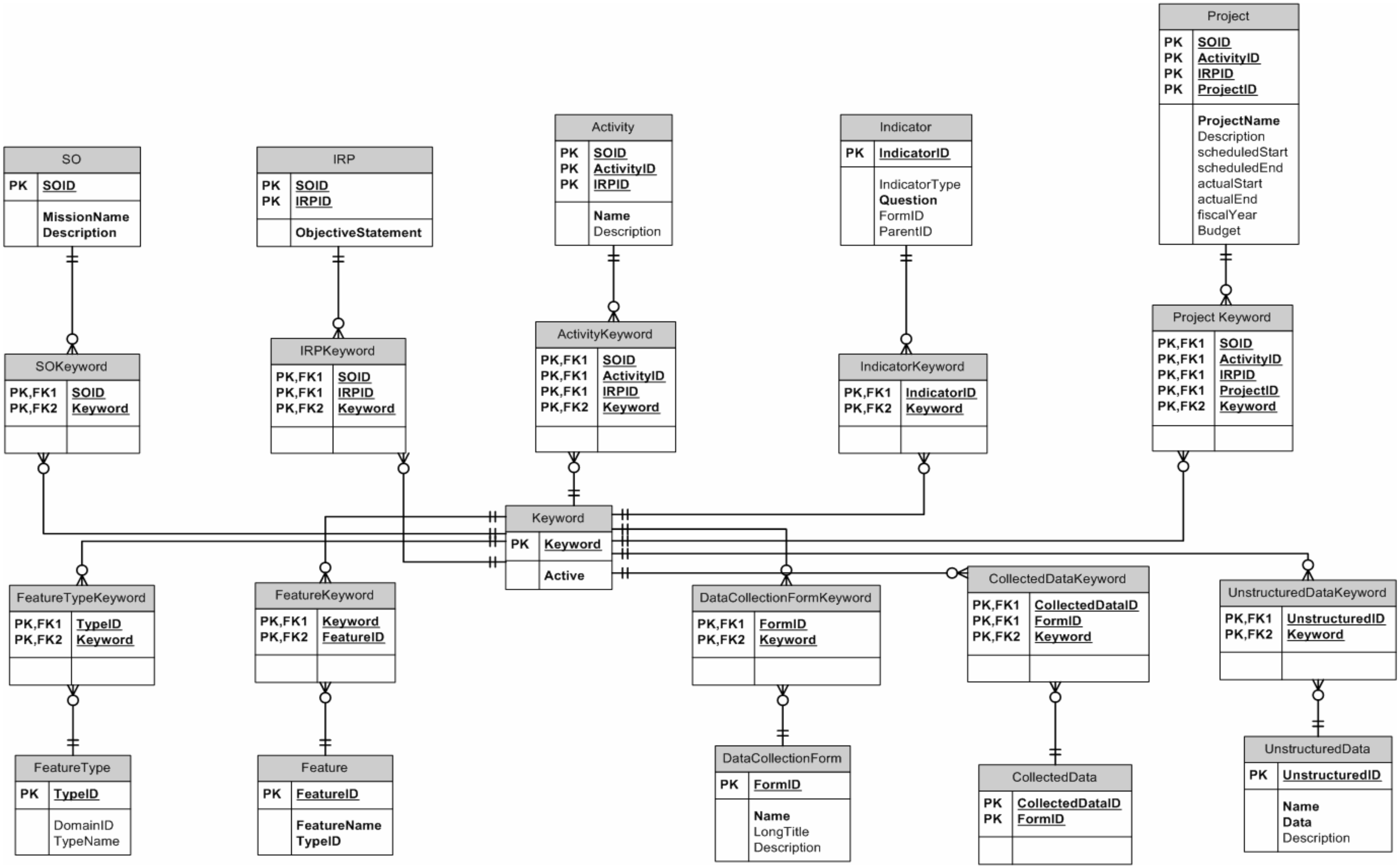

Figure 4.10 Keywords

Since features drive our performance data, a Data Collection Form becomes a set of questions about specific feature types. The germane questions that can be asked about a particular feature type are implicit in the attributes of the feature type. For example, given the attributes in the School table in Figure 4.4, a logical question about that feature type would be "How many students attend the school?" The *numAttending* attribute of the *School* table will hold answers to that question.

The difficulty of this approach is that, due to the widely varying data needs of consumers, the questions that can be asked about a particular feature type must be limited to a set of inherently core questions that won't overwhelm system capacity. This problem is solved by the *FeatureTypeMetadata* table. For each attribute of a specific feature type, there is an entry in that table, *attrDescription*, which will contain the question that can be asked about that attribute. Using this approach, we have standardized an approach to the questions that may be asked about a feature.

4.5 Keywords

Another key discovery made in the Needs Assessment process is the desire to search based on keywords. As can be seen in Figure 4.10 on the previous page, virtually every component of the data model may have multiple keywords associated with it which can be linked to other data model components. This is a simple, but powerful way to find and categorize data.

5 High Level Structural Design

Having identified the Functional Service Areas and described the Data Model, it is now important to examine the Structural Design of the MIS/GIS system. The Structural Design conceptualizes the organization of the system components and the interactions between components. The design is structured according to a system-wide, multi-tiered approach. In most cases, candidate software products of the Functional Service Areas are multi-tiered systems, so components of those systems are present in all tiers. Reference Figure 5.1.

The user interface consists of a Portal, whose function is to aggregate the web front-ends of numerous applications into a single system, from the user's perspective. The Portal packages each web interface as a Portlet; at any one time, a user's view of the Portal consists of a collection of Portlets in a web browser. The business logic of most products resides in the Middleware/Server Tier, and data conveyed in the Data Tier is made consistent at all levels.

Most Functional Service Areas have components shown in each tier, but notable exceptions are the User manager, which is packaged as part of the Operating System, and external services such as Phoenix and the Gazetteer Web Service.

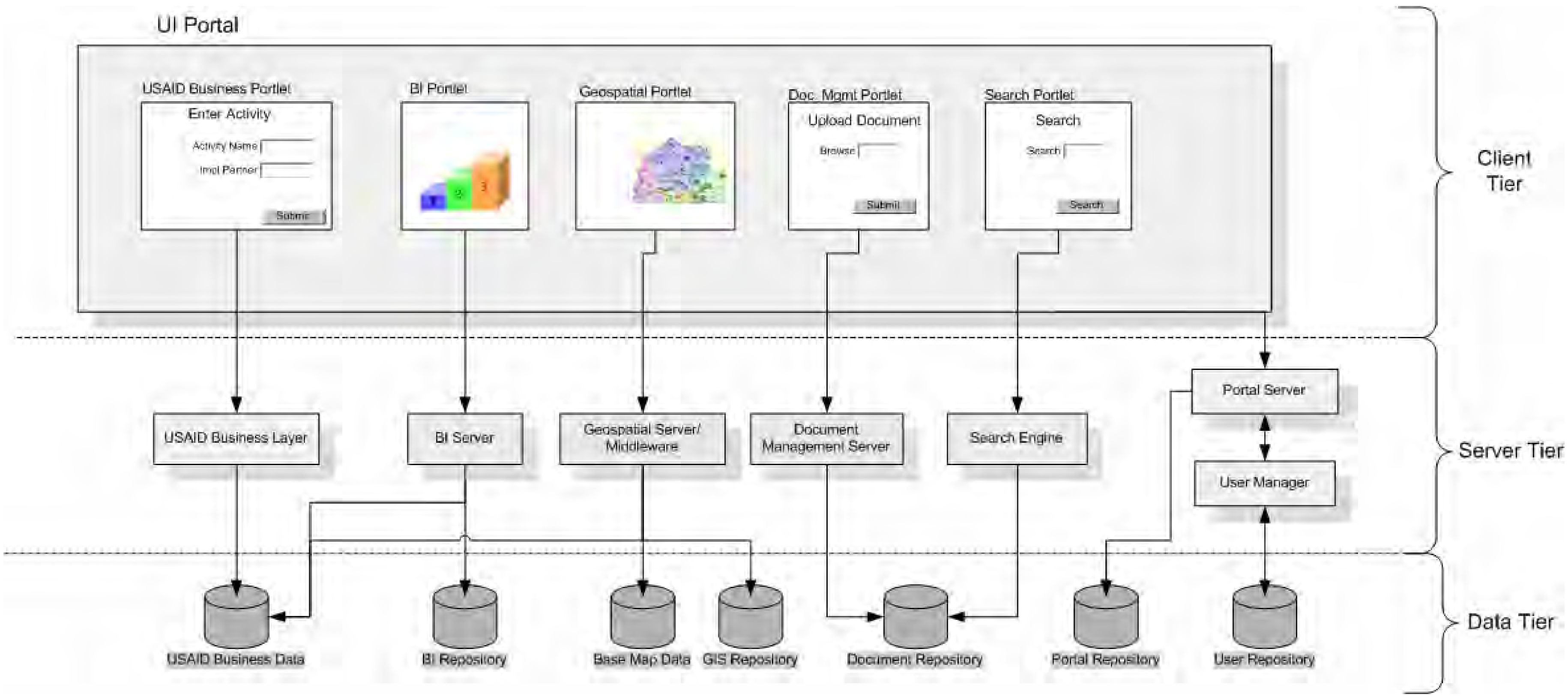

Figure 5.1 Structural View of MIS/GIS

6 Mapping Requirements to Systems Design

The Mapping of Requirements to Systems Design is detailed in the completed Multi-dimensional Requirements Views (MRV's). Effectively, the MRV's show what is often referred to as Behavioral Design. The Application and Data "swim lanes," now complete, map user initiated events to subsequent actions in the software application and data layers of the system

The Application and Data swim lanes have been decomposed to the lowest possible detail. Figure 6.1 shows the MRV diagramming convention.

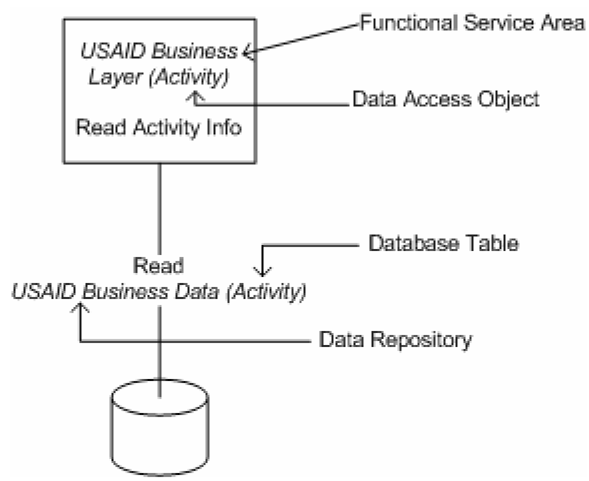

Figure 6.1 MRV legend

Italicized Text in the application box indicates the Functional Service Area performing the action. In the example above, the USAID Business Layer is Reading Activity information. When the Functional Service Area is the USAID Business Layer, the text in parentheses indicates the Data Access Object (See Section 8.1) that is performing the action. Italicized Text in the Data Repository label identifies the Data Repository. When the repository is USAID Business Data, the text in parentheses indicates the table(s) being accessed.

7 Integration Strategy – the ESB

From inception, the Booz Allen team has intended to build the MIS/GIS according to a Service-oriented Architecture, due its advantages in providing a flexible and adaptable system. We believe this is especially true for systems like the MIS/GIS, where new requirements may be discovered as time passes and products may be replaced as needs change.

Central to our approach for service-based integration is the Enterprise Service Bus, or ESB. While there are many benefits of an ESB, in terms of the requirements of the MIS/GIS, the ESB provides two key capabilities for integrating COTS packages. The ESB hosts Web Services, and it also provides a variety of mechanisms to route Web Service requests between applications. Put another way, inter-application messaging is implemented by the

ESB, so no single software component requires any knowledge of another component. In this way, one COTS solution may be seamlessly replaced with another solution. Typically, ESB products provide multiple ways to accomplish this, but the ANE MIS/GIS will use the Business Process Execution Language (BPEL), an XML-based scripting language that defines interactions between services, as its method to integrate Web Services. If one component is replaced by another, updating the business process becomes a matter of editing the BPEL script—see Figure 7.1 below.

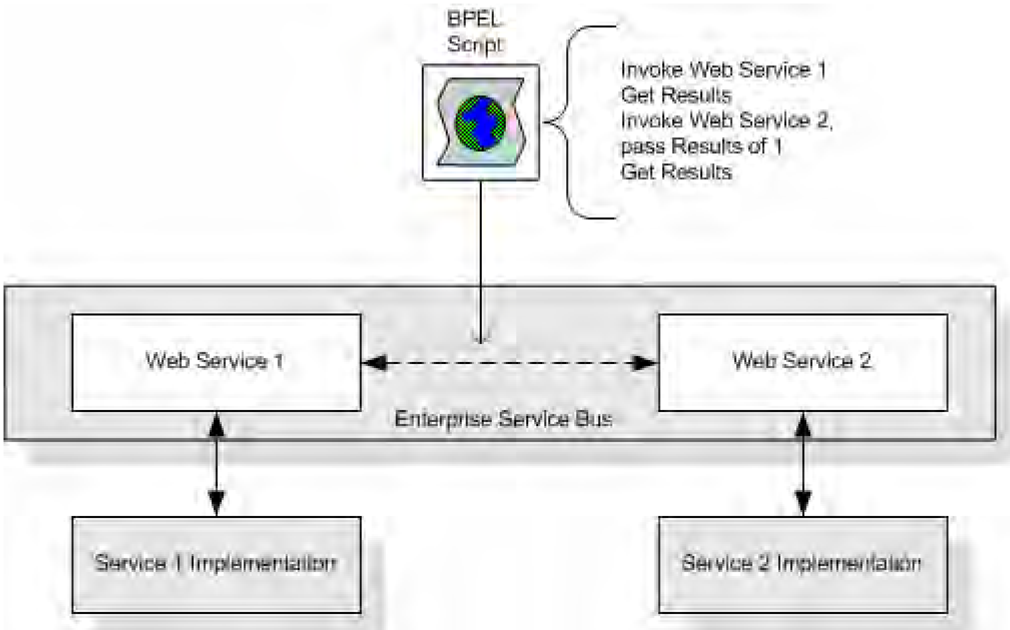

Figure 7.1 ESB and BPEL

8 Detailed Designs

This section provides detailed designs of the more complex Functional Service Areas, and interactions between components.

8.1 USAID Business Layer

The primary purpose of the USAID Business Layer is to serve as a window into the USAID Business Data. This encompasses two major activities: One, data entry and data collection functions (as specified in the Enterprise Activity Roadmap); and two, searching and displaying USAID business data.

The Booz Allen team will be implementing the business layer according to the industry-standard Front Controller, Data Access Object (DAO), and Data Transfer Object (DTO) design patterns [CJE]. The Struts and Hibernate open-source frameworks provide free, enterprise quality implementations of the Front Controller and DAO patterns, respectively.

The architecture is a layered, multi-tier approach, consisting of Java Server Pages (JSP's), Struts, a Service Layer, Hibernate, and the USAID Business Data. Figure 8.1 shows these layers, along with the steps that occur throughout the system as a result of user input. The process is nearly identical for all user input events, so the example given is not specific to any operation.

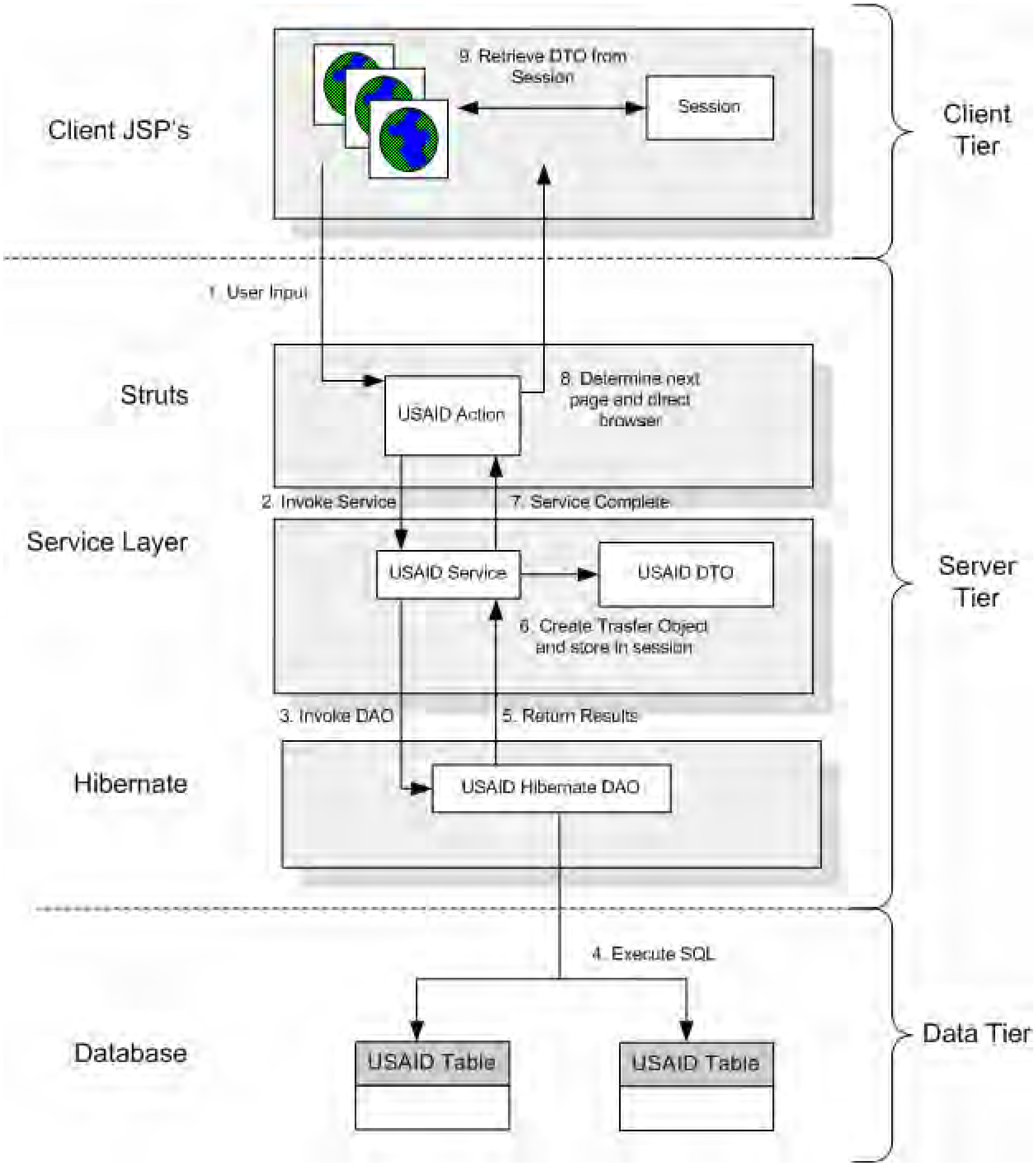

Figure 8.1 USAID Business Layer Architecture

User Input in the Client Tier passes control to the Struts framework. From there, control is passed to the Service layer, a component custom developed to decouple the interaction between the Struts and Hibernate frameworks. The Service layer then uses a Data Access Object, provided by Hibernate, to access the relevant database tables.

The Hibernate Data Access Object will then execute SQL code, and pass the results back to the Service layer. The Service layer then creates a Data Transfer Object and places it in the HTTP session. Control passes back to Struts, which determines the next JSP to load. The JSP can then retrieve the results of the previous operation, by accessing the Data Transfer Object placed in the HTTP session in step 6.

8.2 Integration with External USAID Systems

An important aspect of the long-term viability of the MIS/GIS is the ability to integrate with other USAID systems, particularly Phoenix, PSIP and JAMS. Phoenix is a crucial production system, so allowing the MIS/GIS pilot system to access Phoenix during MIS/GIS development and testing presents unacceptable risk for USAID. PSIP and JAMS are both in active development and thus may not provide reliable, or even complete interfaces. Therefore, those systems must be simulated via database tables that reflect data that would be obtained from each. The MIS/GIS will provide an integration component for Phoenix, but the approach would be identical for PSIP and JAMS. The architecture is depicted in Figure 8.2 below.

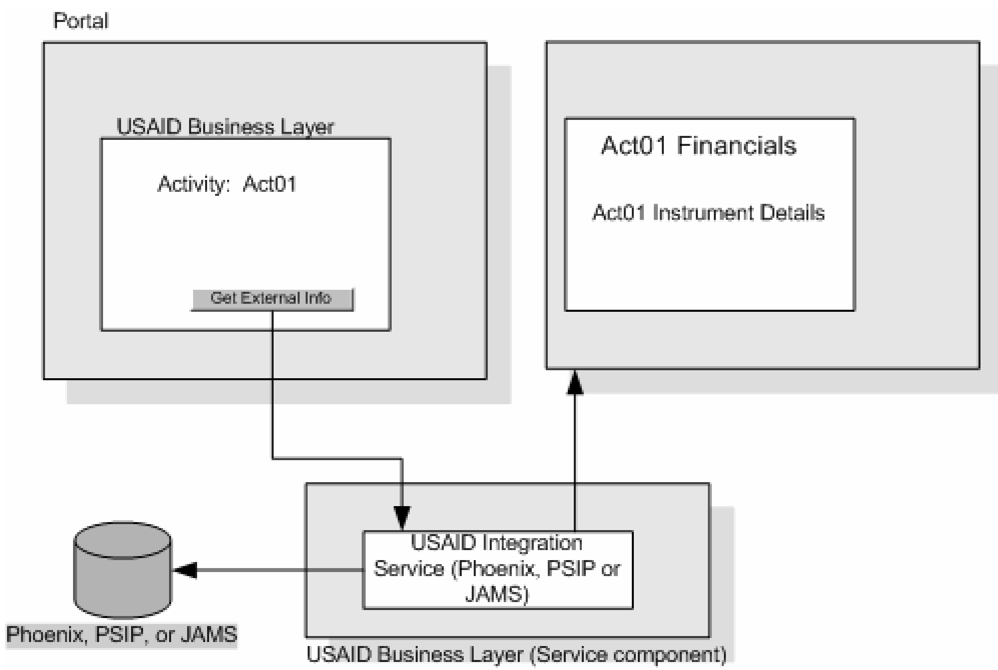

Figure 8.2 USAID Integration Service

Access to a USAID system is initiated by some user request. The request will pass to the USAID Integration Service, a component of the Service layer described in the previous section. During the demonstration project, data stored in the Phoenix table(s) (example data sets from Phoenix stored in the SOA Sandbox) will be accessed by this component. In future implementations, this component can be easily replaced with an implementation that actually accesses Phoenix. This underscores a benefit of the Service layer: By decoupling Struts and Hibernate, it is possible to implement services in the Service layer that do not access Hibernate. Rather, they may invoke external Web Services, initiate access to other repositories, and so on.

8.3 Data Integration

The purpose of the Data Integration Functional Service Area is to translate mission data from disparate data sources into USAID standard structured data as defined by the Data Model detailed in Section 4. Data Integration will follow one of two approaches, the Extraction Transform Load (ETL) approach and the Data Integration Service approach. Figure 8.3 shows the ETL approach.

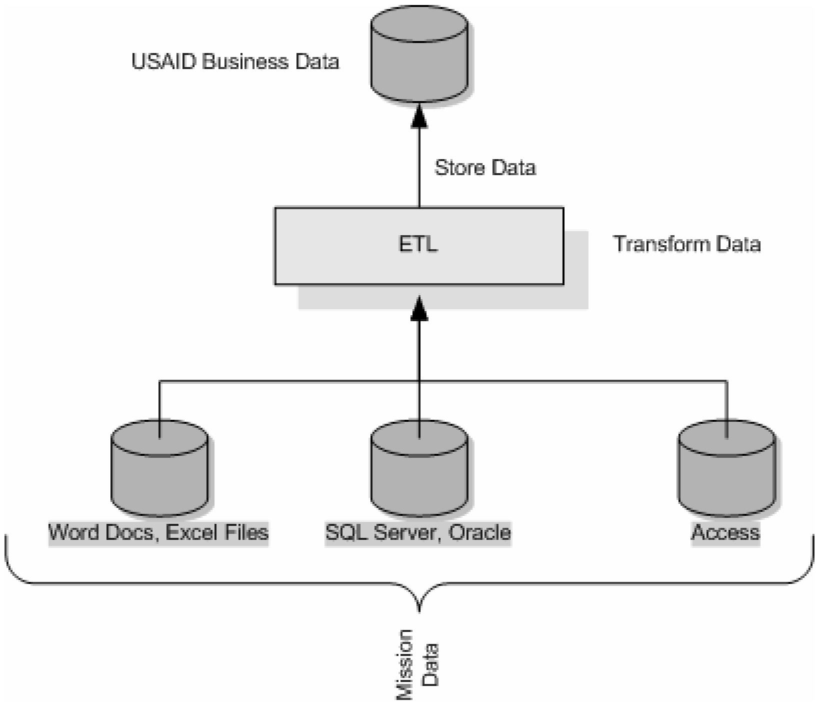

Figure 8.3 ETL Data Integration

The ETL method is appropriately named—data is extracted from all sources, transformed by a software tool, and stored in the defined USAID structure in the USAID Business Data repository. Figure 8.4 shows the Data Integration Service approach.

In this approach, data is left in its native forms, and the Data Integration Service provides an abstraction layer over the data sources. The Data requester will access data through the Data Integration service, with no knowledge of the native formats of the actual data. The Data Requester needs to understand the schema of the intended data format, so that feasible queries can be made to the Data Integration Service.

The Data Integration Service is more flexible, because new data sources can be dynamically added without Data Requesters being aware of any change. However, a Data Requester must be able to discover the metadata of the intended data format, and many tools may not support that capability.

In practice, Data Integration will likely be a hybrid approach. Some tools will likely not support the Data Integration Service approach; therefore, data they require will have to be uploaded via the ETL process.

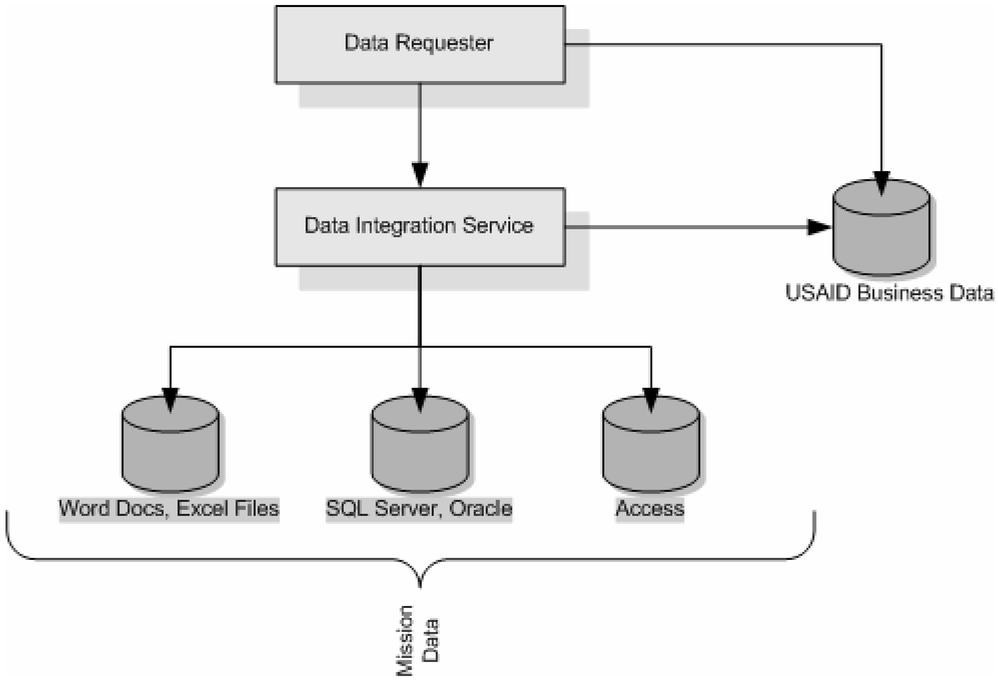

Figure 8.4 Data Integration Service

8.4 Business Intelligence and GIS toolset

Both the Business Intelligence and Geospatial toolsets follow the multi-tier architecture, as shown in Figure 8.5. Each accesses multiple repositories, and each provides a web interface, which will be presented to the user as a Portlet in the Portal.

The nature of the MIS/GIS requires some degree of tight coupling between the Business Intelligence and Geospatial tool systems. If both components provide support for Web Services, it is preferable to accomplish integration via the ESB. Since geospatially-enabled reporting is a common functionality, geospatial vendors tend to support reporting tool vendors, and vice versa. The exact integration approach we take will depend on vendor support.

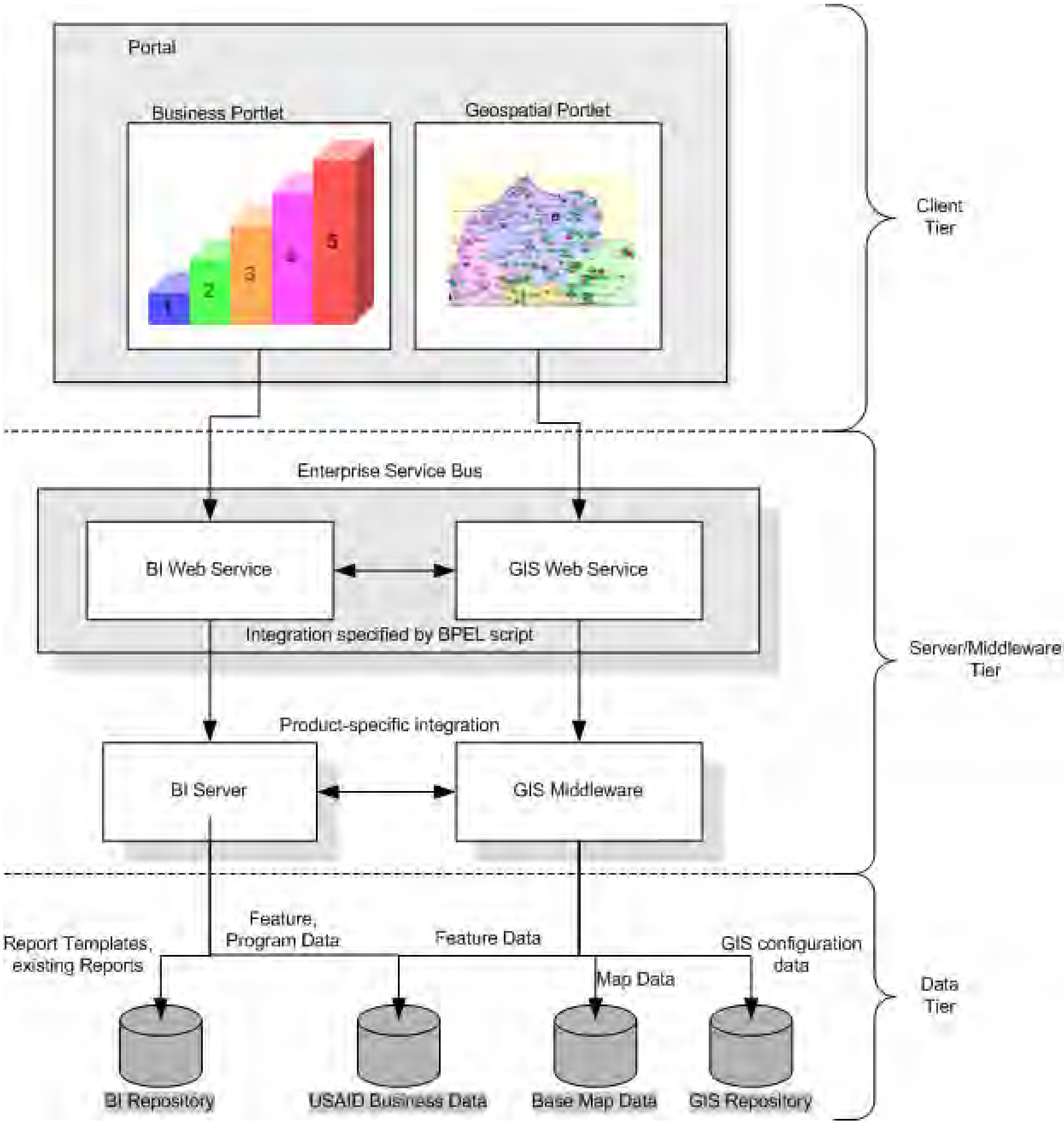

Figure 8.5 BI and GIS tools

8.4.1 Gazetteer Service

The Gazetteer Service looks up Geospatial coordinates based on a supplied place name, thus providing a simple method for assigning a geospatial location to a city, province, or country. The National Geospatial-Intelligence Agency provides this public Web Service. The MIS/GIS will implement a simple wrapper around this service to provide the same functionality, as shown in Figure 8.6.

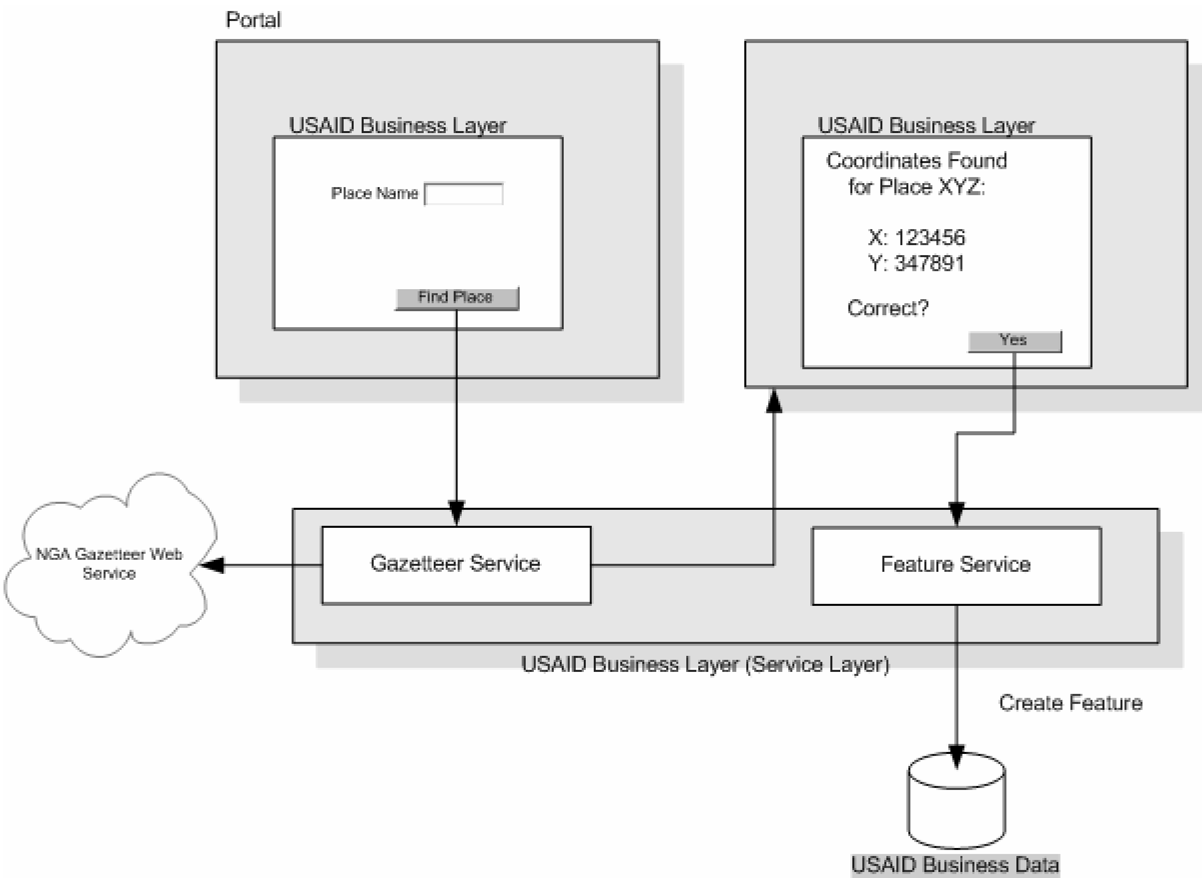

Figure 8.6 Gazetteer Service

8.5 Portal and User Manager Integration

As part of the Needs Assessment, a number of user groups were identified and certain activities must be restricted to certain groups. The ability to filter content based on a user's role is provided by the Portal and the User Manager. We will be using Microsoft Active Directory as the User Manager, which is widely supported by Portal implementations. Figure 8.7 shows the interaction between the Portal and User Manager.

It should be noted that a similar process will be followed when a user logs in. However, the user will need to have his or her credentials authenticated by the User Manager before the groups that the user belongs to can be determined.

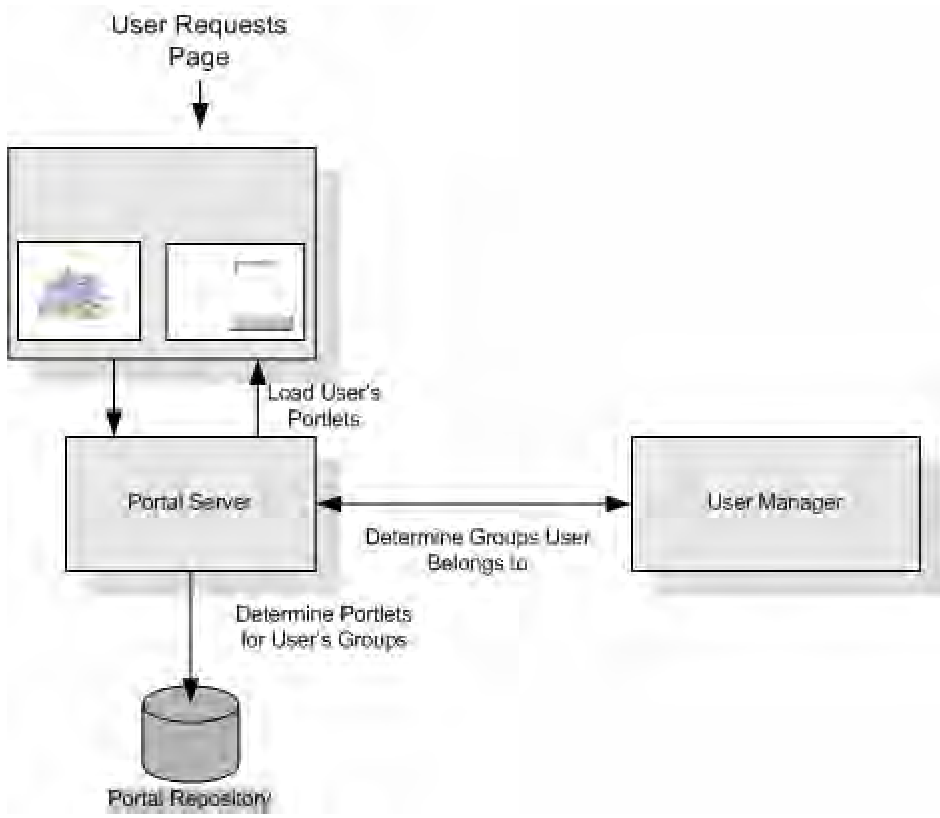

Figure 8.7 Portal and User Manager Interaction

9 COTS Evaluation

The COTS Evaluation process will be performed as part of the Demonstration phase. The Booz Allen team will evaluate the COTS products based on performance, ease of integration, user look and feel, and ability to meet the system objectives as depicted in the MRV and Vignettes.

10 Conclusion

The Booz Allen team believes this to be a robust and flexible architecture for the MIS/GIS system. At this point, we now move into the Demonstration phase, where this approach will either be validated or necessary refinements to improve the design will be identified.

11 Glossary

Business Intelligence Software – software product that facilitates the collection and analysis of data.

Business Process Execution Language (BPEL) – XML-based language intended to model and implement complex business processes. A key benefit is that it enables interaction between Web Services.

Data Model – conceptual representation of the data elements and their relationships to one another within an Information System. Usually depicted in UML or Entity-Relationship diagrams.

Design Pattern – A specific, documented software design or implementation technique that is widely applicable across many domains. First documented by the “Gang of Four” in [GoF].

Enterprise Service Bus (ESB) – class of middleware software products intended to facilitate the development of Service-oriented architectures. Typical features of an ESB include hosting of Web Services, routing requests among multiple Web Services, and data transformation services.

Entity-Relationship Diagram (ER or ERD) – commonly used diagramming technique for data modeling. An ER diagram consists of entities and relationships between entities. An entity is depicted as a box, and a relationship is a line drawn from one entity to another. An ER diagram will be implemented in a relational database—entities in the diagram become tables, while relationships are implemented as Foreign Key relationships between tables. See [DM] for a tutorial on Entity-Relationship Modeling.

Middleware – class of software products that act as intermediaries between different components. The two types of Middleware that are important to the ANE MIS/GIS are J2EE servers and Enterprise Service Buses.

Multi-tier Architecture – software architectural style in which a software application’s components exist in multiple logical tiers. Typically, there is a presentation tier, a business tier, and a data tier. See [TTA] for a tutorial, including an excellent diagram.

Portal – Web-based software system that aggregates other web-based applications into a single interface. A user will see the portal as a single entry point, without knowing that the applications contained within are deployed across multiple systems.

Portlet - basic application unit of a Portal. Typically, each web application in a Portal is displayed in one Portlet, and a Portal is composed of many Portlets.

Relational Model – Common implementation model of modern Database Management Systems (DBMS). Based on mathematical theory, but usage is made more accessible via the Structured Query Language, or SQL.

Service-oriented Architecture (SOA) – Software and Software Systems design approach that emphasizes loosely coupled systems and integration based on open standards. Typically, open standards are publicly available XML-based protocols. The most important of these standards are SOAP and WSDL.

Simple Object Access Protocol (SOAP) – XML based communication protocol used to pass messages and data between applications.

Unstructured Data – data or data file that is not persisted in a relational database. Typically adheres to a proprietary, binary format. Examples include Word, Excel and PDF.

Web Service - software interface that promotes interoperability via support of XML-based open standards. Typically, a Web Service is an interface that communicates with other applications via SOAP, and advertises the services it offers via WSDL.

Web Services Description Language (WSDL) – XML-based language used by applications to define the services that it provides to external applications.

XML (eXtensible Markup Language) - a language used to create other languages (metalanguage). Uses of XML range widely, but commonly include defining open data formats and communication protocols. XML is plain-text, and therefore is easily transmitted and read by disparate computer systems (i.e. it is platform-neutral).

12 References

[CJE] Deepak Alur, John Crupi and Dan Malks. "Core J2EE Patterns," Prentice Hall, 2003.

[DM] Data Modeling: Entity Relationship Model: http://www.utexas.edu/its/windows/database/datamodeling/dm/erintro.html.

[GoF] Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides. "Design Patterns: Elements of Reusable Object-Oriented Software," Addison Wesley, 1995.

[HIB] Hibernate: http://www.hibernate.org.

[STR] Struts: http://struts.apache.org.

[TTA] Three-tier (computing): http://en.wikipedia.org/wiki/Three-tier_%28computing%29.