Developing applications that harness the Agriculture and Natural Resource grid, and linking your own data.

Matthew Laurenson and Seishi Ninomiya
(National Agriculture Research Center, National Agriculture Research Organization)

Background

Weather data, digital elevation models and background maps are important inputs to decision support software for agriculture, weather risk assessments, resource management, and civil engineering. There are many sources of such data on the Internet, but no standards for data access. The Agriculture and Natural Resource grid lets software applications request such data in a consistent way from the diverse range of databases on the Internet.

Introduction to Distributed Computing

Distributed computing approaches link computers in different geographical locations in order to solve problems. Figure 1 shows a distributed computing approach to agricultural decision support. In this approach, data is retrieved from, and saved in a variety of computers, facilitating maintenance and data sharing. We briefly introduce some important concepts in distributed computing such as client server and broker architectures, remote procedure calls, and passing objects by value and then review the respective strengths of the distributed computing implementations RMI, CORBA, and SOAP.

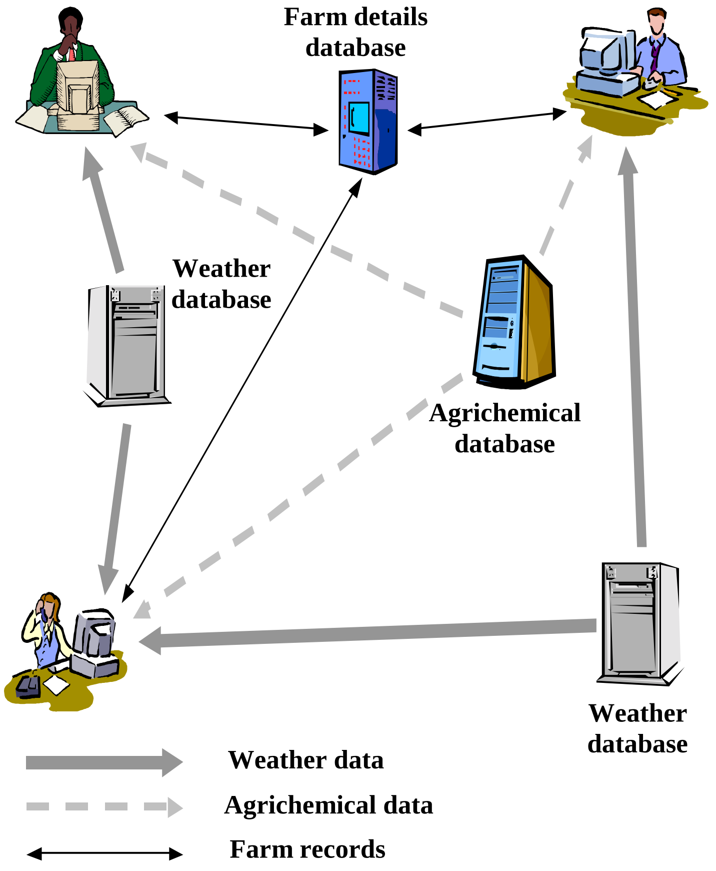


Figure 1 Distributed computing approach to decision support

Basic Broker Concepts

We discuss the general idea of broker operation (Figure 3), and how applications can request data and metadata from a broker. We describe the logical elements of the process, rather than the details of programming.

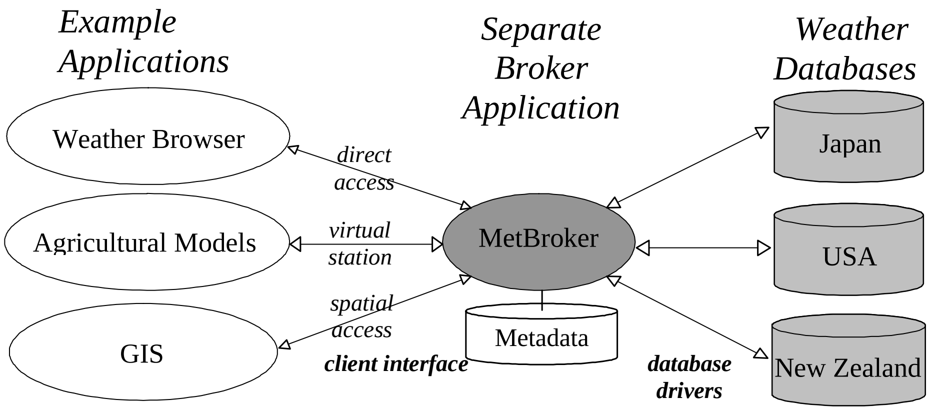
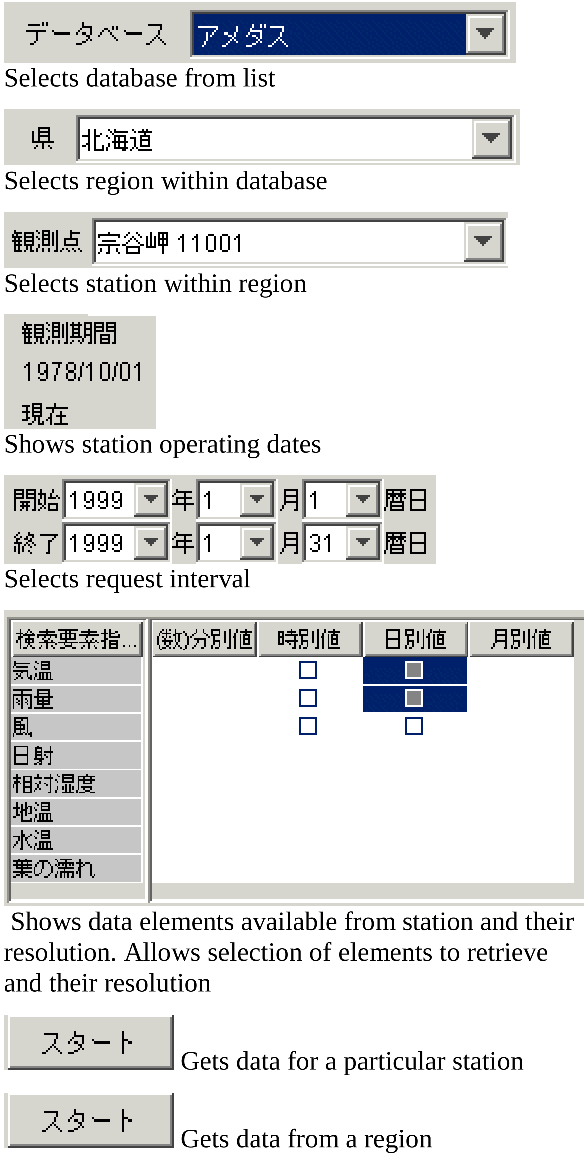


Figure 3 MetBroker structure

Applications can request both data and metadata (details of what data are available) from a broker. Metadata is used to inform users about what data is available from each database, and to display data correctly (for example showing the location of weather stations on a map).

The format of data requests and results is independent of how particular databases are implemented (for example whether the database is relational or file-based). This makes it easy for applications to use newly-linked databases.

Java applications can communicate with brokers using a Java protocol called Remote Method Invocation (RMI). However, we have developed a Simple Object Access Protocol (SOAP) based interface (a Web service) which allows applications written in languages like Delphi and Visual Basic to communicate directly with MetBroker. We hope to develop similar SOAP interfaces to the other brokers in the Agricultural and Natural Resource Grid.



データベース アメダス

Selects database from list

県 北海道

Selects region within database

観測点 宗谷岬 11001

Selects station within region

観測期間
1978/10/01
現在

Shows station operating dates

開始 1999 年 1 月 1 暦日
終了 1999 年 1 月 31 暦日

Selects request interval

検索要素指…	(数)分別値	時別値	日別値	月別値
気温		□	☑	
雨量		□	☑	
風		□	□	
日射				
相対湿度				
地温				
水温				
葉の濡れ				

Shows data elements available from station and their resolution. Allows selection of elements to retrieve and their resolution

Gets data for a particular station

Gets data from a region

Figure 2 MetBroker JavaBeans

Retrieving weather data into a program

This section shows how Java programs can retrieve data from a broker (we use MetBroker as an example). We provide two demonstration programs. The first is a 20 line “text mode” program run from the command line.

The second program provides a Graphical User Interface (GUI) for requesting weather data, and uses a set of JavaBeans that we have developed (Figure 2). These JavaBeans let application developers easily link GUI applications to brokers.

These beans can easily be linked together, so that selections a user makes in one bean are seen and responded to by other beans.

We will demonstrate how to use these beans in a rapid application development tool like Borland JBuilder by creating an application that shows what data is available from stations in any MetBroker database, and can retrieve and display the data.

We discuss how MetBroker can be accessed with simple browsers by using Java servlets as an intermediary.

We briefly describe how the library of MetBroker classes used by such programs is packaged into Java Archive (JAR) files.

Connecting a new database to MetBroker

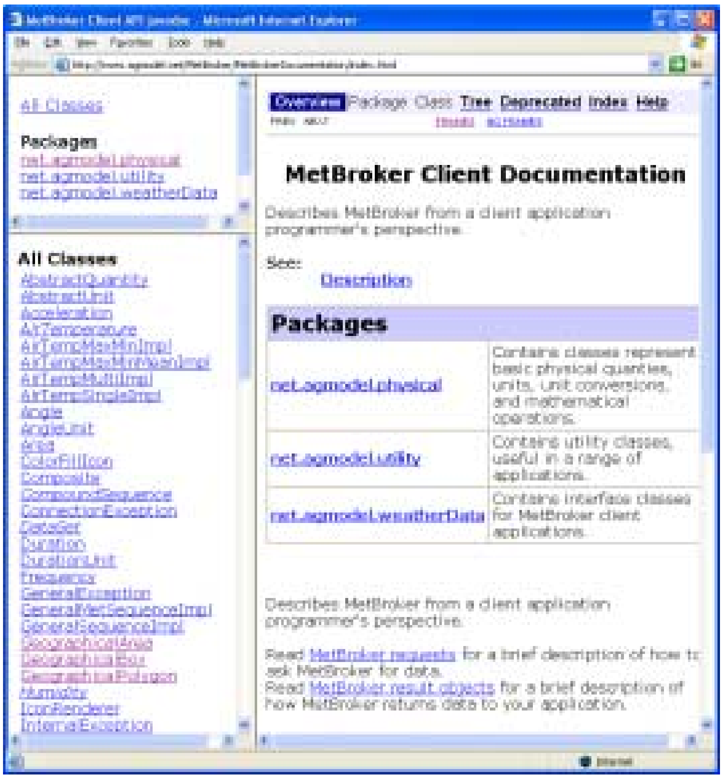
The design of broker’s makes it relatively easy to link to new databases, by separating the code specific to a particular database into a “driver”, much like a Windows printer driver. We outline the three main tasks that such a driver must perform, and introduce examples of linking to a relational database and a file-based database. We briefly discuss options for storing data in a database if one doesn’t already exist.

Support Resources

The interfaces and classes that make up the Agricultural and Natural Resource Grid are documented online in Web pages generated from source code comments (Figure 4).

Broker source code is released under the GNU Lesser General Public License, which lets those interested view and modify the source code, but which requires that any modifications be released under the same license.

The Grid is supported through discussion groups in English and Japanese.



All Classes

Packages
net.agmodel.physical
net.agmodel.utility
net.agmodel.weatherData

All Classes
AbstractQuantity
AbstractUnit
Acceleration
AirTemperature
AirTempMaxMinImpl
AirTempMaxMinMeanImpl
AirTempMultiImpl
AirTempSingleImpl
Angle
AngleUnit
Area
ColorFillIcon
Composite
CompoundSequence
ConnectionException
DataSet
Duration
DurationUnit
Frequency
GeneralException
GeneralMetSequenceImpl
GeneralSequenceImpl
GeographicalArea
GeographicalBox
GeographicalPolygon
Humidity
IconRenderer
InternalException

MetBroker Client Documentation

Describes MetBroker from a client application programmer's perspective.

See:
Description

Packages	
net.agmodel.physical	Contains classes represent basic physical quantities, units, unit conversions, and mathematical operations.
net.agmodel.utility	Contains utility classes, useful in a range of applications.
net.agmodel.weatherData	Contains interface classes for MetBroker client applications.

Describes MetBroker from a client application programmer's perspective.

Read MetBroker requests for a brief description of how to ask MetBroker for data.
Read MetBroker result objects for a brief description of how MetBroker returns data to your application.

Figure 4 MetBroker online documentation