Applying Scientific Workflows to Manage Sensor Data

Gilberto Z. Pastorello Jr[1], Claudia B. Medeiros[1], André Santanchè[2]

[1] Institute of Computing – University of Campinas (UNICAMP)
Av Albert Einstein, 1251 – Campinas – SP – Brazil
{gilberto,cmbm}@ic.unicamp.br

[2]DCEC – UNIFACS
Av Cardeal da Silva, 747 – Salvador – BA – Brazil
santanche@unifacs.br

Abstract. *There is a world wide effort to create infrastructures that support multidisciplinary, collaborative and distributed work in scientific research, giving birth to the so-called e-Science environments. At the same time, the proliferation, variety and ubiquity of sensing devices, from satellites to tiny sensors are making huge amounts of data available to scientists. This paper presents a framework with a twofold solution: (i) using a specific kind of component – DCC – for homogeneous sensor data acquisition; and (ii) using scientific workflows for flexible composition of sensor data and manipulation software. We present a solution for publishing sensor data tailored to distributed scientific applications.*

1. Introduction

In the recent past, e-science initiatives focused in research areas that had need for high performance computing, e.g., meteorology, genomics, particle physics. This has fostered research in Computer Science mainly in computer clusters and grids, and in data-intensive support systems. Currently we see a movement to offer computational support to a number of other research areas, including environmental planning and monitoring, agriculture, biodiversity, social sciences, and arts [Hey and Trefethen 2005, Almes et al. 2004, Roberto et al. 2006]. Supporting these new areas involves research in other Computer Science fields, such as databases (storage, retrieval and integration of multimodal and heterogeneous sources), compilers, human-computer interfaces. In particular, data generated by sensing devices are increasingly important to scientific research and applications. This paper concerns supporting sensor data acquisition, manipulation and publication.

We are facing the proliferation of several kinds of sensing devices, from satellites to tiny sensors. This has opened up new possibilities for us to understand, manage and monitor a given environment, from the small – e.g., a room – to the large – e.g., the planet. In particular, wireless sensor networks (WSN), i.e., networks of communicating small sensing devices, powered by batteries and with limited storage and processing facilities, are subject to intensive research. Network nodes are frequently heterogeneous, generating distinct kinds of data at different time intervals. From the data perspective, challenges include dealing with integration of heterogeneous sources, data redundancy, data streams and real-time data, data fusion and summarization, all subject to node, sensor and communication failures. From the network point-of-view, challenges comprise physical device management, event detection and notification, power management, dynamic reconfiguration of nodes and network, and support to different simultaneous applications, among others. However, the proliferation, variety and ubiquity of these devices add new dimensions to the problem of heterogeneous data management.

Figure 1 outlines our proposal to deal with these issues, where layers denote different data access and manipulation levels, from data sources to applications. The continuous lines denote a data flow path and the dotted lines denote a reference to data sources. The first layer contains the data sources such as services, DBMS, files (e.g., text, imagery), and, in particular, sensor and their auxiliary devices. The second layer, detailed in section 2, contains a specific kind of component: *Digital Content Components* (DCC) [Santanchè et al. 2007], which have Semantic Web conformant annotations. Both data and data sources[1] are encapsulated within DCCs, so uniform interfaces are available to the applications, for dynamic (e.g., sensors) and stable (e.g., time series files) sensor data sources; and for dynamic (e.g., stream) and stable (e.g., on image) sensor generated data.

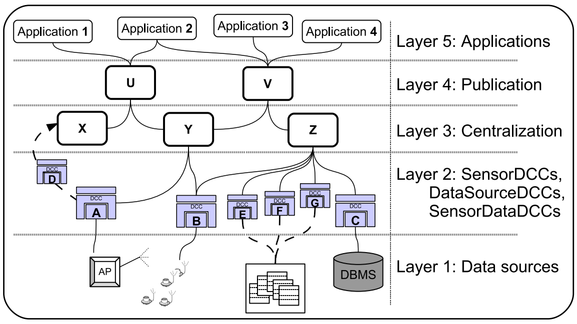

Figure 1. Management layers

The third layer contains elements that centralize and manipulate the encapsulated sensor data from layer 2, offering functions such as data fusion, summarization, classification and sampling. In this layer, the processing elements can be single DCCs and/or scientific workflows. Workflows are applied to control and compose the basic functions so that the data are tailored to fit applications' needs. The fourth layer has the publication and data access mechanisms, offering high abstraction level interfaces. Applications in Layer 5 are regarded as clients of Layers 3 and 4.

Sensor data have particular requirements to be dealt with, specially concerning data streams manipulation and data fusion schemes. Sensor networks present an even bigger challenge as some of these solutions can be implemented within the network. Our DCC implementations provide solutions to these aspects, including the possibility of using intra and/or extra-network algorithm implementations. In this paper we explore the composition of these solutions using workflow activities, which are transparently executed by invoking DCC operations.

This proposal has the following advantages:
(1) it provides homogeneous access to heterogeneous sensing devices;
(2) it enables applications to have multiple views of sensing data, by taking advantage of scientific workflows to mediate sensor data access. This fosters reuse of solutions for managing these data;
(3) it eliminates the need for an application to concern itself with whether a data source is static or dynamic. Items (1) and (3) are presented in detail in [Pastorello Jr et al. 2007]. This paper's main contribution lies on issue (2), presenting workflow specifications and categories for sensor data manipulation.

[1]We distinguish between data – the bits and bytes – and data sources – device or resources that provide data. This distinction can become blurred at times – e.g., a file can be treated either as data or as a data source.

2. Revision on DCC and Encapsulation of Resources

A *Digital Content Component* (DCC) is a unit of content and/or process reuse, which can be employed to design complex digital artifacts [Santanchè and Medeiros 2005, Santanchè et al. 2007]. From a high level point of view, a DCC can be seen as digital content (data or software) encapsulated into a semantic description structure. As shown in Figure 2, it is comprised of 4 sections:
(i) the content itself (data or code, or another DCC), in its original format. In the example, the content is a driver for communicating and gathering data from a MICAz sensor (`www.xbow.com/Products/productsdetails.aspx?sid=101`);
(ii) the declaration, in XML, of a structure that defines how DCC internal elements relate to each other (here, delimitating the object code of the driver);
(iii) specification of an interface, using adapted versions of WSDL and OWL-S – e.g., the `getTemp` and `subscribeGetTemp` operations, in the example;
(iv) metadata to describe functionality, applicability, etc., using OWL (in the example, the DCC is declared as belonging to the `TemperatureSensorDCC` class and being located at the `longitude` and `latitude` specified.
Interface and metadata are linked to ontology terms – e.g., the `getTemp` operation has as input parameter a timestamp as defined by the "Time" concept of NASA's SWEET [Raskin and Pan 2003] ontology.

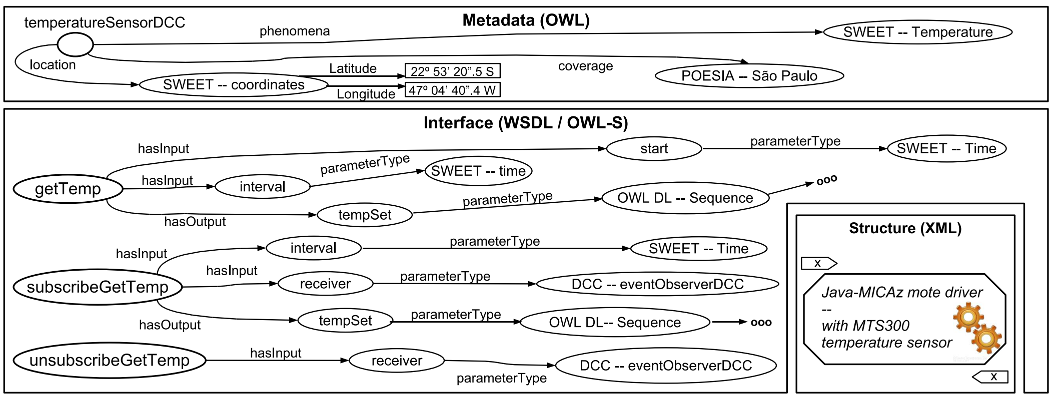

Figure 2. DCC schematic structure (from [Pastorello Jr et al. 2007])

There are two main kinds of DCC – process and passive. A *ProcessDCC* encapsulates any kind of process description that can be executed by a computer (e.g., software, sequences of instructions or plans). Their interfaces declare operations they can execute. Non-process DCCs, named *PassiveDCCs*, consist of any other kind of content (e.g., a text or video file); their interfaces declare their *potential functionality*, in the sense that the operations can be requested from them, but their execution code is not part of the content (e.g., a video player software is not part of a video). Since passive components by definition cannot embed executable code, operation implementations are encapsulated into special ProcessDCCs called *CompanionDCC* (e.g., a piece of music `M` stored in a PassiveDCC can be played by attaching `M` to a suitable music player in a CompanionDCC). We refer the reader to [Santanchè et al. 2007] for details the DCC infrastructure.

Recalling Figure 1, the second layer contains Passive and ProcessDCCs respectively encapsulating data and data sources. DCCs that provide access to sensor data, called *DataSourceDCCs*, play a role comparable to that of a mediator to access the data produced by a sensor. A specialization of a DataSourceDCC is the *SensorDCC*, which takes care of accessing sensors and collecting their data. One SensorDCC can encapsulate one (DCC **B**) or more sensors (**A**) by, for instance, encapsulating an access point of a wireless sensor network. Data can be delivered to the next layer in its original format (**C**) or encapsulated in another DCC, as done by **A**, delivering **D**. **D**, **E**, **F** and **G** are examples of *SensorDataDCC*, a PassiveDCC used for sensor data encapsulation and annotation. For more details on encapsulation of resources the reader is referred to [Pastorello Jr et al. 2007]. Layer 3 (**Y** and **Z**) has data organization and centralization features (pre-processing, summarization and fusion). Finally, the fourth layer has the publication and access control features, and offers raw and processed data to applications in Layer 5. Features of layers 3 and 4 can be implemented either as software, event-based composition (using publish/subscribe techniques), or flow-oriented compositions (using workflows). The third kind is our focus in this paper and is explored next.

3. Scientific Workflows

3.1. Basic Concepts

A *workflow* is a specification (or model) of a *process*, which is a set of inter-dependent steps needed to complete a certain task. A *Workflow Management System* (WFMS) is a computer system for specification, execution and management of workflows [WfMC 1995]. A *scientific workflow* is a specification of a process that describes a scientific experiment [Wainer et al. 1996]. While most business workflows are first specified and then executed several times, a scientific workflow is often specified while an experiment is conducted, focusing on process documentation. Other singular characteristics include: (i) emphasis on data centric processes; (ii) a high degree of flexibility on specification and adaptation; (iii) support to uncertainty at specification and execution time, as well as a great number of exceptions during execution; (iv) possibility of modifying a workflow during its execution.

These characteristics pose problems to workflow specification, for which there are several proposals (e.g., WS-BPEL). Our workflow data model [Pastorello Jr et al. 2005, Medeiros et al. 2005] is compliant with international standards and induces a methodology for scientific workflow specification. Our model has been implemented in the WOODSS [Medeiros et al. 2005] framework for scientific workflow management. The several abstraction levels of a workflow are stored in a relational database and can be reused as workflow building blocks. Moreover, the specification of types and of abstract workflows capture the notion of *workflow design* independent of execution aspects, allowing design reuse. This supports the need scientists have for sharing and reusing not only executable procedures, but also their specification. The model requires that the users first specify types of workflow building blocks, next combine them into abstract specifications (*abstract workflows*), and finally refine these specifications into executable workflows (*concrete workflows*).

3.2. Management Workflows

Our solution adopts scientific workflows as a flexible way to coordinate the management of sensing data. Here, rather than directly accessing sensor data files, or interacting with sensor middleware, the idea is to take advantage of DCC interfaces and composition mechanisms. Thus, a solution constructed for one scenario or one set of sensors can be easily reused in different environments, adapting the DCCs that are available in the repository. Applications can select or post adequate workflow specifications in order to obtain the desired data. This solution adds reusability and flexibility for sensor data management.

The goal of a *management workflow* is to manage the sensor data in order to support complex application requests. More specifically, it controls SensorDCCs and ProcessDCCs (embedding data manipulation software), coordinating the tasks of data extraction, processing and interpretation. Furthermore, a management workflow can access SensorDataDCCs as additional data sources.

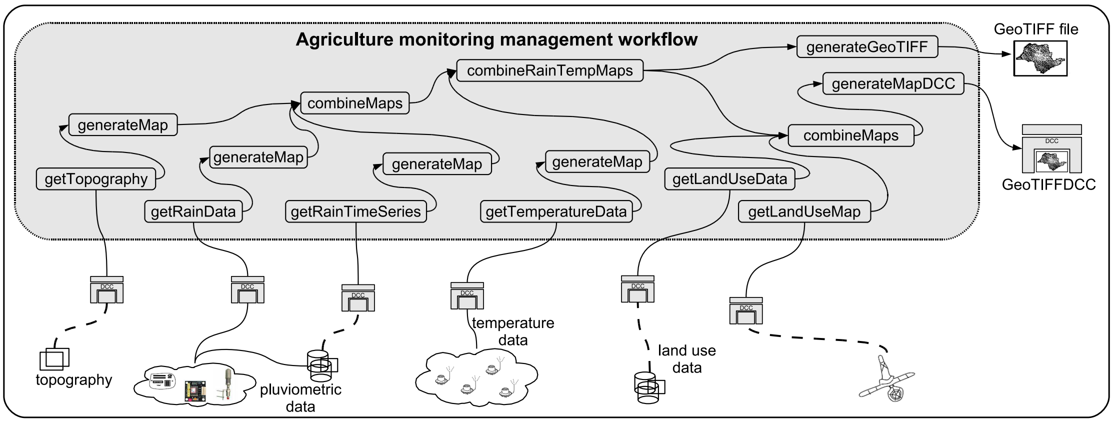

Figure 3. An example of a management workflow for crop monitoring

Figure 3 shows an example of a management workflow for agricultural monitoring – i.e., it controls conditions in a given region to help crop management. It is used to acquire data from distinct kinds of sensors and sensor-derived data files. It periodically produces derived data, here a file of a map in GeoTIFF (a bitmap format that associates geographic coordinates to pixels). For instance, activity `getRainData` accesses a SensorDCC for real time rainfall readings, while `getRainTimeSeries` accesses a SensorDataDCC for historic rainfall data. Both generate maps that are combined to detect rainfall evolution patterns, taking into account a region's topography (map generated from topographic data).

Workflow activities access DCCs for data input, and can generate other DCCs, e.g., GeoTIFFDCC. The main mechanism to determine which DCCs can be used for a given activity is based on a combination of ontology annotations and type matching (of DCC operation interfaces and metadata, and activity specification). This is supported by the DCC management infrastructure – see section 4. We point out a few aspects that characterize our solution. The monitoring workflow is only concerned with sending data requests to DCCs, *regardless of the nature* of the data sources. Moreover, the workflow *manages and publishes* sensor data, and publication results can themselves be encapsulated in a DCC.

Our workflows are specified favoring our reuse methodology [Medeiros et al. 2005]. The first step consists in the definition of the data and activity types. Next, activities are created from types: after choosing an activity type, one can choose the data that is going to be used by that activity, based on the data types of the activity's parameters. Activities may be created only at an abstract specification level – thus defining an *abstract workflow*. This workflow can be then customized to specific situations, and subsequently instantiated for execution. Management workflows can be reused and adapted to solve new data needs. To make a workflow executable, pieces of software must be associated to each of its activities. This is where the DCCs come into action. Workflow execution is carried out transparently by a WFMS, which uses DCC operations, coordinating the data flow.

3.3. Validation Workflows

We can use several validation techniques to make sure that data follow given quality or validity parameters. This evaluation can be specified using the workflow and DCC based infrastructure. An interesting aspect of these mechanisms is the possibility of offering data quality assurance to the applications.

We propose five methods for data validation, which can be used individually or in a sequential combination inside a workflow. The methods are: sampling, summarization, pre-processing, and pre- and post-condition verification. Sampling, summarization and pre-processing are basic features frequently used to extract information from data sets. In the validation context, the idea is to extract representative sets of values from sensor-produced data, perform additional processing and evaluate the results against some benchmark. Pre- and post-conditions consist in evaluating logical expressions using the sensing data as variables.

All the validation schemes can be stored and published along with the data. Thus, data production can be traced back to its source, so that all the manipulations are available to be analyzed. Pre- and post-conditions can be stored as text. Pre-processing, summarization and sampling can be stored by references to the respective DCCs used (including DCCs that encapsulate workflows).

3.4. Publication Workflows

Publication is the final aspect of data management in our work. Management workflows are geared towards supplying data for specific application needs (e.g., generating erosion maps or controlling the temperature in a factory environment). Publication workflows, on the other hand, are general purpose data providers.

Functionalities to publish sensor data include: (i) data fusion schemes, even exerting device reconfiguration control; (ii) data summarization and sampling, with configuration parameters; (iii) application of statistical analysis over the readings; (iv) data classification, filtering, clustering and many other mining related techniques. These functionalities are accessed and composed into a publication workflow via management operations offered by individual and aggregated operations on SensorDCCs.

4. Implementation Issues

Figure 4 illustrates the main modules of the architecture. Scientists (the main users) interact with it via the GUI (Graphical User Interface) to design workflows and DCCs, and monitor workflow and/or DCC execution. The architecture relies on the following subsystems: (i) WOODSS [Medeiros et al. 2005] – a scientific workflow specification and documentation environment developed at UNICAMP; (ii) ANIMA [Santanchè et al. 2007], an infrastructure developed to support DCC execution and management; (iii) a set of modules to design DCCs, eventually reusing and modifying DCCs from a repository [Santanchè and Medeiros 2005]; and (iv) a basic infrastructure needed to monitor and execute workflows, including the WFMS.

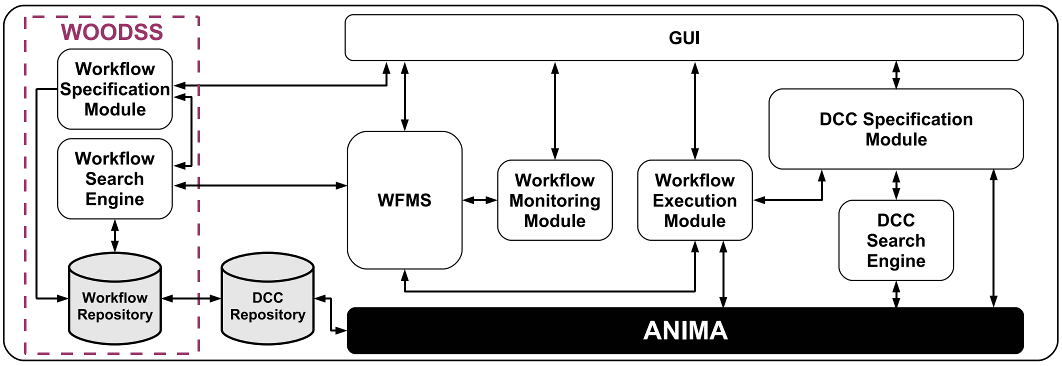

Figure 4. The main modules of the architecture.

In more detail, WOODSS allows users to specify scientific workflows, using our methodology, and to annotate both workflows and data manipulated by them. Workflow specifications (at all abstraction levels), their components, and annotations are stored in the Workflow Repository. Scientists can construct a workflow from scratch, or reuse and adapt stored workflows, retrieved with help from the Workflow Search Engine, based on workflow metadata – e.g., are there any workflows that create erosion maps? Repository records point to external elements they invoke (e.g., a Web Service) or manipulate (e.g., data sets) – [Medeiros et al. 2005].

DCC specification and management are handled by the modules on the right of the figure. Users interact with the GUI either to specify a new DCC from scratch, or to construct a DCC reusing and adapting existing stored components. The goal of the DCC Search Engine is to help DCC design and composition, supporting the user in finding the most appropriate DCC to reuse for a given application purpose – e.g., is there any SensorDataDCC that encapsulates rainfall time series for São Paulo State? This engine's search mechanisms are based on a set of algorithms that combine type annotations, metadata and interface matching [Santanchè and Medeiros 2005].

We implemented SensorDCCs for the MICAz mote and for the TelosB mote (`www.xbow.com/Products/productsdetails.aspx?sid=126`). These SensorDCCs have the same interfaces and were tested with temperature and light readings. SensorDataDCCs implemented include temperature, light, rainfall map and rainfall map set.

To select a workflow for execution, a user (or an application) sends a request to the WFMS, which invokes the Workflow Search Engine to retrieve the desired workflow from the Workflow Repository. All data and DCCs needed to execute the workflow are all retrieved in this search step (e.g., our SensorDCCs).

Subsystems (i) through (iii) are already implemented, and we do not intend to implement a workflow engine. Rather, we will use some available system.

Meanwhile, subsystem (iv) has been replaced by a simple coordination mechanism, extracted from Anima's synchronization modules. This allows simulating reasonably complex workflows, including those with parallel branches, synchronization and loops. The GUI has distinct modules that support graphical workflow design, DCC specification and construction of composite DCCs [Santanchè et al. 2007]. Present implementation efforts are concerned with two issues: first, we are concentrating on the development of more DCCs for sensors. Next, we will support internal interactions among WOODSS, WFMS and DCC modules for their integrated execution. Present integration occurs only at the repository level (the Workflow Repository points at DCCs stored in the DCC Repository). Moreover, the DCC Search Engine is not yet integrated into the system. We also intend working on this.

5. Related Efforts

We have commented on related work throughout the paper. There remains to compare our proposal with alternative approaches, in particular regarding two aspects. One aspect concerns the (general-purpose) data management solutions, which we propose solving through scientific workflows. Another issue is homogeneous access to resources, achieved by encapsulating the sensors within DCCs.

Scientific workflows are extensively used in e-Science, e.g., orchestrating Web services, or specifying execution of experiments in a grid environment [Meyer et al. 2006, Yu and Buyya 2005]. Other efforts include applying workflow technologies [Cavalcanti et al. 2005, Ludäscher et al. 2006, Zhao et al. 2006, Braghetto et al. 2007], distributed scientific data [Cavalcanti et al. 2002] and multi-modal scientific data management [Torres et al. 2006]. The major difference of these approaches to ours is the homogeneous treatment of data, data sources, and software. To the best of our knowledge, no proposals exist to use such workflows to manage access to heterogeneous sensing devices. Other alternatives for data management solutions include: (i) Specialized implementation, e.g., an entire software system for one specific application, which has the classic overhead of unnecessary repetition of work, hard maintenance, lack of standardization and interoperability. (ii) Other composition techniques, such as a publish-subscribe scheme [Eugster et al. 2003], which are also promising, but are not suited for flow (or process) oriented executions.

From the point-of-view of homogeneously accessing the data, one approach alternative to ours is to directly use Web services to publish data, and to have access to sensors. However, this has two drawbacks: (1) unlike PassiveDCCs, Web services do not actually encapsulate data, and thus are always associated with some specific implementation; (2) a Web environment is mandatory for Web services, while DCCs can also be used in a standard programming environment, regardless of the Web (and its overhead). Other accessing solutions considered are: (i) Specialized (and language specific) implementations; (ii) Software components and communication middlewares, such as CORBA, DCOM and .NET, EJB, and others; (iii) WSN middleware as defined by [Hadim and Mohamed 2006]. The first approach has the same drawbacks of the specialized implementation for data management. Components and general middleware lack flexibility, semantic descriptions, and, more importantly, homogeneous treatment of data, devices and software. WSN middleware [Gibbons et al. 2003, Madden et al. 2005, Yao and Gehrke 2002] are centered either on specific platforms or specific applications, none consider accessing data and software through homogeneous interfaces.

6. Concluding Remarks

With the possibility huge amounts of sensor data production, efficient management of these data is mandatory. In scientific research this is an even more sensible problem, since both the data sources and the applications accessing the data are typically heterogeneous and distributed. We described a solution to provide means to access and process sensor data in this scenario, aiming at flexibility and reusability of solutions. The main contribution of this work is the specification and implementation of a framework that: (i) provides distributed access and processing features to sensor data using DCCs, and (ii) flexible composition mechanisms using workflows for managing these DCCs. Ongoing work involves inclusion and evaluation of more sensor data sources and improving the execution mechanisms.

Acknowledgements. This work is supported by FAPESP (grant 2004/14052-3) and partially financed by a CNPq grant, the WebMaps II project and an HP Digital Publishing grant.

References

Almes, G., Cummings, J., Birnholtz, J. P., Foster, I., Hey, T., and Spencer, B. (2004). CSCW and cyberinfrastructure: opportunities and challenges. In *Proc. of the 2004 ACM Conf. on Computer Supported Cooperative Work*, pages 270–273.

Braghetto, K. R., Ferreira, J. E., and Pu, C. (2007). Using control-flow patterns for specifying business processes in cooperative environments. In *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*, pages 1234–1241.

Cavalcanti, M. C., Mattoso, M., Campos, M. L., Simon, E., and Llirbat, F. (2002). An Architecture for Managing Distributed Scientific Resources. In *Proc. of the 14th Int. Conf. on Scientific and Statistical Database Management*.

Cavalcanti, M. C., Targino, R., Baião, F., Rössle, S. C., Bisch, P. M., Pires, P. F., Campos, M. L. M., and Mattoso, M. (2005). Managing structural genomic workflows using Web services. *Data & Knowledge Engineering*, 53(1):45–74.

Eugster, P., Felber, P. A., Guerraoui, R., and Kermarrec, A. (2003). The many faces of publish/subscribe. *ACM Computing Surveys*, 35(2):114–131.

Gibbons, P. B., Karp, B., Ke, Y., Nath, S., and Seshan, S. (2003). IrisNet: An Architecture for a World-Wide Sensor Web. *IEEE Pervasive Computing*, 2(4).

Hadim, S. and Mohamed, N. (2006). Middleware Challenges and Approaches for Wireless Sensor Networks. *IEEE Distributed Systems Online*, 7(3).

Hey, T. and Trefethen, A. E. (2005). Cyberinfrastructure for e-Science. *Science*, 308:817–821.

Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E. A., Tao, J., and Zhao, Y. (2006). Scientific workflow management and the Kepler system. *Concurrency and Computation: Practice & Experience*, 18(10):1039–1065.

Madden, S. R., Franklin, M. J., and Hellerstein, J. M. (2005). TinyDB: An Acquisitional Query Processing System for Sensor Networks. *ACM Transactions on Database Systems*, 30(1):122–173.

Medeiros, C. B., Perez-Alcazar, J., Digiampietri, L., Pastorello Jr, G. Z., Santanchè, A., Torres, R. S., Madeira, E., and Bacarin, E. (2005). WOODSS and the Web: Annotating and Reusing Scientific Workflows. *SIGMOD Record*, 34(3):18–23.

Meyer, L., Annis, J., Wilde, M., Mattoso, M., and Foster, I. (2006). Planning spatial workflows to optimize grid performance. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 786–790.

Pastorello Jr, G. Z., Medeiros, C. B., Resende, S. M., and Rocha, H. A. (2005). Interoperability for GIS Document Management in Environmental Planning. *Journal on Data Semantics*, 3(LNCS 3534):100–124.

Pastorello Jr, G. Z., Medeiros, C. B., and Santanchè, A. (2007). Providing Homogeneous Access for Sensor Data Management. Technical Report IC-07-012, Institute of Computing, State University of Campinas.

Raskin, R. and Pan, M. (2003). Semantic Web for Earth and Environmental Terminology (SWEET). In *Proc. of the Workshop on Semantic Web Technologies for Searching and Retrieving Scientific Data*.

Roberto, P. A., Santos, R. L. T., Gonçalves, M. A., and Laender, A. H. F. (2006). On RDBMS and Workflow Support for Componentized Digital Libraries. In *Proc. of the XXI SBBD*, pages 87–101.

Santanchè, A. and Medeiros, C. B. (2005). Self Describing Components: Searching for Digital Artifacts on the Web. In *Proc. of XX SBBD*, pages 10–24.

Santanchè, A., Medeiros, C. B., and Pastorello Jr, G. Z. (2007). User-centered Multimedia Building Blocks. *Multimedia Systems Journal*, 12(4):403–421.

Torres, R. S., Medeiros, C. B., Gonçalves, M. A., and Fox, E. A. (2006). A digital library framework for biodiversity information systems. *International Journal on Digital Libraries*, 6(1):3–17.

Wainer, J., Weske, M., Vossen, G., and Medeiros, C. B. (1996). Scientific Workflow Systems. In *Proc. of the NSF Workshop on Workflow and Process Automation Information Systems*.

WfMC (1995). The Workflow Reference Model. Technical report, WfMC. TC-1003.

Yao, Y. and Gehrke, J. (2002). The Cougar Approach to In-Network Query Processing in Sensor Networks. *SIGMOD Record*, 31(3):9–18.

Yu, J. and Buyya, R. (2005). A taxonomy of scientific workflow systems for grid computing. *SIGMOD Record*, 34(3):44–49.

Zhao, Y., Wilde, M., Foster, I., Vöeckler, J., Dobson, J., Gilbert, E., Jordan, T., and Quigg, E. (2006). Virtual data grid middleware services for data-intensive science. *Concurrency and Computation: Practice & Experience*, 18(6):595–608.