DaltOn: An Infrastructure for Scientific Data Management

Stefan Jablonski¹, Olivier Curé², M. Abdul Rehman¹, and Bernhard Volz¹

¹ University of Bayreuth, Bayreuth, Germany
{Stefan.Jablonski, Abdul.Rehman, Bernhard.Volz}@uni-bayreuth.de
² Université Paris Est, S3IS/IGM France
ocure@univ-mlv.fr

Abstract. It is a common characteristic of scientific applications to require the integration of information coming from multiple sources. This aspect usually confronts end-users with data management issues which involve the transportation of data from one system to another as well as the syntactic and semantic integration of data, i.e. data come in different formats and have different meanings. In order to deal with these issues in a systematic and well structured way, we propose a sophisticated framework based on process modeling. In this paper, we present the three major conceptual architectural abstractions of the system and detail its execution.

1 Introduction

Information integration aims to enable the rapid development of new applications requiring information from multiple sources. This task is becoming a critical issue for both businesses and individuals. Its complexity is mainly due to the uprising of data volumes and the proliferation of sources, types of information. For instance, in other papers we report from a medical application [4] which enables patients to perform self-medication safely and efficiently. This web-based application supports patients with many services including a drug proposition service. The quality of this service partly depends on the ability to integrate coherently valuable information sources (e.g. databases) and thus to answer patient's inquiries in a consistent manner. This integration is not limited to be a syntactical issue (i.e. format alignments) but essentially a semantic issue, i.e. the information contained in the sources may not agree on a common semantics and hence produce inconsistent medical conclusions.

Furthermore data integration is not a “one shot approach” as it may be needed to integrate new sources and to take care of the schema evolution of already integrated sources. Thus we promote to incorporate data integration in the information system underpinning an application. We consider a process-based information system to be the ideal candidate for such incorporation. We call this approach Data Logistics (DaLo, [9]) and summarize it in Section 2. Section 3 presents the framework responsible for the data management issues within the DaLo, named DaltOn, and emphasizes all execution aspects on a concrete micrometeorology example. Section 4 relates our research to other approaches and in Section 5 we conclude this paper.

2 Workflow Management and Data Logistics

As described in several publications [5] [9] we pursue a special approach to workflow (process) management in medical and scientific applications which is called Data Logistics (DaLo). This approach supports a very flexible execution of workflows as compared to the traditional way of prescribed workflow execution in conventional workflow management systems [8]. DaLo-workflows propose many interesting features but for the purpose of this publication only the focus on modeling data specific issues is relevant. What does this mean? In a first phase of the workflow specification the principle structure of a workflow is defined. This is the usual way of specifying what steps have to be executed in what order, what people (or systems) are responsible to perform those steps and what applications have to be called when a workflow step has to be executed. Next, in a second phase of specification, data management issues are focused. This means that issues related to data management must be defined. We will explain this feature with an example in Section 3.

We call each issue that has to be dealt with when modeling and executing a workflow “perspective”. Thus we differentiate at least the following perspectives: The functional perspective describes what step has to be executed; the behavioral perspective defines the execution order of work steps, the organizational perspective determines agents eligible and responsible to perform a work step and the operational perspective describes tools (applications) used when a certain work step is executed. All the data management related issues are dealt by the data perspective. The underlying meta model for this perspective oriented approach is called Perspective Oriented Process Management (POPM) which is presented in [10].

Fig. 1 shows a process model taken from a real world scenario of a scientific workflow from micrometeorology research. Before we go into more detail of the process and its execution, we want to describe the structure of the process according to POPM. Each step of a process depicted as a rectangle shows the functional perspective. The small rectangle underneath the right side of a process step denotes the operational perspective. A black arc and a small rectangle over the arc (showing data items) is considered as the data perspective which deals with both the description of input and output data of work steps and data flow between work steps as well. The gray arcs depict the behavioral perspective which realizes execution dependencies. It is out of the scope of this paper to discuss all the perspectives and their implementation, instead our focus is on the implementation of the data perspective. In the next sections we will show how our framework implements the data perspective and copes with data management issues.

As already mentioned, the workflow depicted in Fig. 1 is taken from micrometeorology research where weather data is acquired and then analyzed. In data acquisition phase, scientists collect datasets from diverse sensors at different locations, prepare them, dump the prepared datasets at an intermediate place - so called “FileServer”, and finally move them from the FileServer to a central repository (EcoDatabase). In the first step “GetData”, data are extracted from sensors and then moved to the next step “ValidateData” which uses the in-house built application “ValApp” for validating the extracted data. Then after validation the (extracted and validated) data are dumped into the FileServer at a different location. In order to move the data from the FileServer to the central repository, the “DataSelection” step extracts the data and

moves it to the next step “StoreSensorData”, in case interpolation is not required. If it is required then the step “Interpolation” will be executed; otherwise the data will be moved directly to “StoreSensorData”. The step “StoreSensorData” is then responsible for storing the data into EcoDatabase using the “DBEco” application.

Fig. 1. An example workflow from the micrometeorology domain

It is worth to consider the overall architecture of our workflow management system before starting a detailed discussion of those parts that deal with data management issues. In principle, the architecture comprises one component for each perspective in use. In the context of this paper, the implementation of the data perspective is of most interest. Data perspective of the POPM approach is implemented by DaltOn (Data Logistics and Ontology based integration).

Whenever a data transmission between work steps of a workflow takes place, DaltOn is called. A data transmission is always taking place in between two work steps regardless whether data must be moved physically or not. Data are exchanged between data sources which are associated with regarding work steps. Due to different notions and namespaces of participating applications of a DaLo workflow, format, terminology and ontology transformations are needed. The constitutive idea of this data integration task has been published in [5]; in this paper we contribute the architecture of the DaltOn framework. Section 3 presents this architecture in more detail; especially its components are introduced and their orchestration is depicted.

3 Architecture of DaltOn

3.1 Introductory Example

DaltOn is a framework which deals with data management issues such as data exchange, semantic and syntactic data integration in well structured and transparent way. Thus it facilitates domain users (especially scientists in scientific workflows) in managing domain related processes by allowing DaltOn to care about data complexity. Fig. 2 depicts a part of the example workflow shown in Fig. 1. It highlights a scenario of the two work steps “DataSelection” and “StoreSensorData”, especially it is focusing on the data transfer between these steps.

Fig. 2. Focusing data transfer between work steps

In Fig. 2 we want to zoom into the data perspective, i.e. into the overall transmission of data between work steps. Each work step consumes input data (IN_x) and produces output data (OUT_x). Therefore the input and output of the work step “Data-Selection” in Fig. 2 are IN1 and OUT1 and those for the work step “StoreSensorData” are IN2 and OUT2 respectively. The output of the “DataSelection” step (OUT1) is a document containing weather data (single items are separated by spaces) taken from sensors in a proprietary format called “PWD”. The schema is defined in terms of an ObjectID (OID) which identifies each data item uniquely, a TimeStamp (TS) that gives the time at which data has been recorded, a HardwareError code (HE) which shows the status of the (sensor) hardware, the VisibilityPerOneMinute value (VOM) which reflects the visibility, an InstantPresentWeather code (IPW) that provides NWS codes and WaterIntensity value (WI) which is showing the intensity of water at one minute average. The input of the work step “StoreSensorData” (IN2) is again a document that contains weather data but this time the format is not PWD but XML. Furthermore the weather data requires filtering since records with a VOM greater than 2000 are not integrated into the database and thus can be excluded.

Also a terminological transformation is required since codes in the OUT1 dataset will be integrated with values in the IN2 dataset. The schema of IN2 is defined as ObjectID (OID), TimeStamp (TS), Status, Visibility, InstantPresentWeather (IPW) and WaterIntensity (WI). Obviously, OUT1 must then somehow be related to IN2 since the output of “DataSelection” should become the input of “StoreSensorData”. Here some major conceptual questions arise: Is OUT1 syntactically and semantically compatible with IN2? If some incompatibility shows up, how can data transformations are performed? And how are the data actually transported? We introduce “WeatherData” as a kind of common data structure compatible to OUT1 and IN2, respectively, and the data that have to be transported from “DataSelection” to “StoreSensorData”. Then the tasks to be performed by DaltOn are first to convert OUT1 into “WeatherData”, transport it and finally convert “WeatherData” into IN2.

3.2 Architectural Components

Fig. 3 depicts the architecture of DaltOn and its associated components. DaltOn has three major conceptual architectural abstractions, namely Data Provision, Data Integration and the internal Repository.

Data Provision aims at data exchange between data producing steps (sources) and data consuming steps (sinks). It consists of two components, namely Data Transportation (DT) and Data Selection/Filtering (DSF). DT handles the physical data

transportation between sources and sinks by utilizing wrapper objects which encapsulate each source and sink. DSF is responsible for extracting the dataset based on end-users' selection and filtering criteria through configuration data at modeling time, hence only the data which are required by sink are extracted. Wrappers are supporting the communication with the sources; they provide a uniform interface for the access of each source/sink and extract or insert the data using the source's/sink's proprietary format and access method (e.g. SQL statements in case the source/sink is a relational database).

Fig. 3. DaltOn's architectural components

Here, it is worth to mention that DaltOn is not restricted to be used solely in the context of workflow management. Source and sink are two abstract components which represent a data producer and a data consumer, respectively. In our context, both data producers and data consumers are work steps; however, they can represent conventional applications in other contexts as well. Therefore data wrappers work as adapters to the data sources in order to provide a uniform access since each (type of) source usually has its own interface and is accessed differently.

Data Integration instead aims at syntactic and semantic transformations. Accordingly, this module encircles two components, namely Format Conversion (FC) and (Ontology-based) Semantic Integration (SI). FC is not only responsible for converting data formats between sources and sinks but can also be used by other components of the DaltOn framework. For instance, the SI component consumes and produces the data only in XML format. This representational format can totally be independent from the formats issued by sources and required by sinks. The FC component then takes care about these types of format conversions.

SI deals with data integration using a (semantic) mediation mechanism based on ontologies. In order to operate properly, it requires the following from a DaLo workflow: (i) a reference ontology, (ii) local ontologies associated to both applications and whose elements (i.e. concepts and properties) relate to elements of the reference ontology (iii) schema for the IN and OUT data and (iv) mappings between elements of the schemata and their respective local ontology. The semantic mediation is performed via matching concepts [15] from one local ontology to another. Since we discussed the theory and the algorithms of the SI component in detail in [5], interested readers should refer to this publication in order to get more details on this component.

Another conceptual abstraction in the DaltOn framework is the Repository. This encompasses information about all operations of DaltOn performed on each data item. It also contains information needed by every components of DaltOn, in particular SI with its reference and local ontologies, document schemata, mappings between schema and ontology elements, instance documents and ontology alignments.

3.3 Execution Semantics

DaltOn is invoked by the workflow management system (WfMS) each time when a work step is ready to be executed. Thus DaltOn prepares and moves the data necessary for performing that particular work step of a process.

We will now demonstrate how DaltOn components interact with each other and with WfMS using the example shown in Fig. 2. The logical sequence of messages occurring during this exemplary run of the DaltOn framework is depicted using sequence diagram in Fig. 4. Data Integration operations performed by DaltOn can be divided into three phases as show in the diagram. The first phase is responsible for extracting the data from the source (in our example the work step “DataSelection”) and converting it into the format that can be acceptable by the SI component later on. The second phase performs semantic integration of the data taken from step “DataSelection” in order to make it compatible with step “StoreSensorData” (the description of the example showed that there is some semantic integration necessary). Finally the third phase is performed which converts the data into a format that is understood by the “StoreSensorData” and transports the converted data to the location where “StoreSensorData” expects it to be. When DaltOn is invoked by the WfMS it stores all the information necessary for the whole integration process into its local repository (we did not depict these intermediate storage tasks/messages as they would have overcrowded Fig. 4). This information comprises the local ontologies of source and sink (“DataSelection” and “StoreSensorData”), the reference ontology that connects local ontologies, information about formats of source and sink (“PWD” and “XML”) and references where data can be found (data wrapper for “FS-Select”), where it is to be placed (data wrapper for “DBEco”) and a selection criteria that determines which subset of data should be used (“VOM not greater than 2000”).

In the first phase, the data extraction from “DataSelection” is started by DaltOn with the “PrepareSelectionStatement” call to the DSF. The DSF then reads the information stored in the repository and creates a statement that is valid for the data source (“DataSelection”) and that reflects the selection criteria provided by the user during modeling time of the process. After the statement has been prepared, DaltOn asks the DT component to move the data from the source (“DataSelection”) into the internal repository (message “MoveData”) using the selection criteria generated in the previous call. As DT component cannot access the data directly, it uses a data wrapper for the source to extract the selected subset (message “GetData”). After the data has been extracted and moved into the repository, DaltOn triggers FC for format conversion since data extracted from “DataSelection” is in a proprietary format (“PWD”) but the SI component expects it to be in XML. Thus the FC component converts the data from PWD to a XML representation in reaction to the “Convert” message received from DaltOn and also stores the resultant data into repository. The second phase, the semantic integration, can then be started. DaltOn thus calls the SI component

(message “Integrate”). In order to perform the semantic integration, the SI component extracts the data, all the relevant ontologies (two local ontologies and the reference ontology) and some mapping information from the repository. After the integration has been finished, the (converted) data is put again into the repository. This also demarks the end of the second phase.

Fig. 4. Sequence diagram showing the interaction of DaltOn components

The third and last phase, the transportation of the data, is initiated by DaltOn with a call to the FC component (message “Convert”) since the format of the output of the SI component may differ from that which is expected by the step “StoreSensorData”. After the conversion DaltOn instructs the DSF component to prepare an insertion statement (message “PrepareInsertionStatement”). Because a work step may consume data combined from several sources, special statements are necessary to tell clearly which part of the combined data item has to be stored. After this insertion statement has been prepared, DaltOn calls the DT component for transporting the data to a place where the work step “StoreSensorData” expect it (message “MoveData”). Again DT uses a data wrapper – this time for the sink, i.e. “StoreSensorData”.

It is noteworthy to mention that all the data exchange in between components is performed through the DaltOn Repository. This allows the Repository to track down all operations performed on the data and record them for later use (Data Provenance). DaltOn does not need to systematically carry out all integration steps; e.g. if both source and sink, use the same format and ontology, only data transportation is performed. Thus all the calls for the FC and SI components can be omitted.

The sequence diagram of Fig. 4 also shows the design rationale of DaltOn: every sub-component (e.g. SI, DSF, FC, DT) and the Repository are in principle independent of each other. This allows for exchanging and re-combining the sub-components easily in the future and reduces the resources necessary for maintaining the system. This design fosters the sustainability of the DaltOn system.

4 Related Work

We want to relate our approach to two domains, data integration solution provided by scientific workflows and data integration systems in scientific applications.

Kepler [11] is a scientific workflow system which has been developed for scientists like ecologist, geologist and biologists and it is built on Ptolemy II, a PSE from electrical engineering [14]. Kepler uses a semantic mediation based approach to integrate heterogeneous data [3]. Concerning the semantic data integration, the approaches of Kepler and DaltOn's SI are quite similar as they both aim to transform data semantically. This transformation uses ontologies and mappings, called registration mappings, to semi-automatically generate mappings between services with heterogeneous schemas, named structural types. Although the objectives of these systems are almost identical, the design of the solution is different as Kepler does not use the semantics of the ontologies to generate the mappings. In fact, end-users define registration mappings in the form of correspondences between queries on the service output/input, named ports, and contextual paths on the ontologies. The system then treats the paths to generate mappings between ports. We argue that SI is a real semantic approach as it exploits concept definitions to perform the alignment of ontologies. These ontologies are represented in Description Logics [1] and thus propose some associated reasoning procedures (concept subsumption, instance checking, knowledge base consistency) that are used during ontology matching [15].

Regarding syntactic data integration and data transportation, in Kepler the end-user has to introduce explicitly predefined and specialized actors for format conversion and for data transportation. Taverna [7] is a scientific workflow management system like Kepler; it is part of the myGrid project [12]. The Taverna workbench allows users to graphically build, edit, and browse workflows. It started life in bioinformatics applications and workflows are mostly used for the specification and execution of ad hoc in silico experiments using bioinformatics resources. These resources might include information repositories or computational analysis tools providing a Web Service based or custom interface. The workflows are enacted by the FreeFluo [6] engine and can be monitored within the Taverna workbench. In order to convert data formats, Taverna follows almost the same approach as Kepler by introducing so-called "Shims". Shims are little services (local workers, beanshell processors/code snippets, and in some cases, nested workflows) which transform one format to another. DaltOn approach differs from these systems in the way that it handles format conversion (syntactical conversion) implicitly by its so called FC component. Taverna doesn't support semantic data integration directly [13], instead the end-user typically needs some kind of specialized translation services which transform the schemas and perform the mappings as well. Thus for semantic data integration, the end-user needs to access a service that knows how to map schemas and ontologies. Whereas DaltOn approach provides a system which deals with this type of semantic integration issue in a systematic way so that end-user doesn't need to introduce such type of services.

BACIIS [2] (Biological and chemical information integration system) is an on-demand information integration system for life science web-databases. The architecture of BACIIS is based on the mediator-wrapper approach augmented with a knowledge base. The approach is similar to DaltOn approach as it follows a semantic mediation for data integration and extracts data via implementation of specific

wrappers. It provides web based interface on which end-users define queries to diverse web databases. BACIIS aims at integrating life science data sources and produces results as an integrated view on a web-based interface, whereas DaltOn aims at data exchange between diverse heterogeneous data sources. In addition DaltOn provides a conceptual separation between data retrieval (data provision) and integration mediation. Also in BACIIS, the concepts underlying Data Logistics have not been considered. In BACIIS, mediator transform data from its source database format to the internal format used by the integration system, while in DaltOn, format conversion is dealt with in a systematic way.

5 Conclusion

We are stressing in this paper that data integration is a critical issue for information management and processing. We argue that a novel and valuable approach to ensure proper communication between systems and applications is to incorporate the data integration and exchange solution within a perspective oriented process modeling. DaltOn is such a framework and implements the data perspective of the POPM approach. The most important advantages to adopt such an approach are the following: The first advantage is an increased readability of the scientific workflows. In conventional approaches (cf. Section 4) data management steps (syntactic and semantic conversions, filtering etc.) are mixed up with "real" workflow steps which describe the scientific analysis to be performed. Through the integration of POPM with DaltOn these two issues can adequately be separated (cf. Section 2); thus work steps for scientific analysis are not disguised by work steps that deal with data management issues. Second there is a clear modularization inside the DaltOn framework. This software engineering principle nicely fosters the adaptation of DaltOn to varying application scenarios. For example, if a new format needs to be integrated, only the FC component of DaltOn needs to be adjusted. The same applies to the incorporation of new data sources (e.g. a new type of sensors); here only a new or adjusted wrapper component must be provided to DaltOn. Third, especially in the scientific domain, users are interested in information about what happened to their data during the execution of a workflow. This well recognized request is often referred to as data provenance. The DaltOn repository holds all information about tasks performed on the data inside a workflow and can provide it to the user— according to the demand of data provenance. Scientists are then able to retrace every single operation performed on data. Another major advantage is the possibility to integrate scientists in the workflow, i.e. the workflow management system can directly interact with users. This is possible because POPM fosters the inclusion of organization models into the organizational perspective. Thus scientists that are developing a process can clearly define which step(s) must be executed by a person rather than a computerized agent. This is helpful in scenarios where decisions are based on users' experiences which determine the continuation of a scientific analysis.

Last but not least re-use of existing concepts such as processes, data, selection criteria and applications is fostered. The modeling environment for POPM provides libraries for each type of concept that store the definitions of these and allow sharing information across many applications. DaltOn can be improved in many different

ways. For instance, we aim to extend the relation between the internal Repository, the data integration and the data provenance issue.

References

1. Baader, F., et al.: *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, New York (2003)
2. Ben-Miled, Z., Li, N., Baumgartner, M., Liu, Y.: A decentralized approach to the integration of life science web databases. *Informatica (Slovenia)* 27(1), 3–14 (2003)
3. Bowers, S., Ludäscher, B.: An Ontology-Driven Framework for Data Transformation in Scientific Workflows. In: Rahm, E. (ed.) *DILS 2004. LNCS (LNBI)*, vol. 2994, pp. 1–16. Springer, Heidelberg (2004)
4. Curé, O.: Semi-automatic Data Migration in a Self-medication Knowledge-Based System. In: Althoff, K.-D., Dengel, A., Bergmann, R., Nick, M., Roth-Berghofer, T.R. (eds.) *WM 2005. LNCS (LNAI)*, vol. 3782, pp. 373–383. Springer, Heidelberg (2005)
5. Curé, O., Jablonski, S.: Ontology-based Data Integration in Data Logistics Workflows. In: Parent, C., Schewe, K.-D., Storey, V.C., Thalheim, B. (eds.) *ER 2007. LNCS*, vol. 4801, Springer, Heidelberg (2007)
6. Freefluo Workflow Enactment Engine (visited: November 27, 2007), <http://freefluo.sourceforge.net/>
7. Hull, D., et al.: Taverna: a tool for building and running workflows of services, *NAR (web service issue)* (2006)
8. Jablonski, S., Bussler, C.: *Workflow Management – Modeling Concepts, Architecture and Implementation*. International Thomson Computer Press, London (1996)
9. Jablonski, S.: Process Based Data Logistics: Data Integration for Healthcare Applications. In: *1st European Conference on eHealth (ECEH 2006)*, Fribourg, Switzerland (October 2006)
10. Jablonski, S., Volz, B.: Database Based Implementation of Meta Modeling Concepts (in German). *Datenbank Spektrum* 24 (to appear 2008)
11. Ludäscher, B., et al.: Scientific Workflow Management and the Kepler System. *Concurrency and Computation: Practice & Experience* 18(10), 1039–1065 (2006)
12. myGrid Website (visited November 27, 2007), <http://www.mygrid.org.uk>
13. Oinn, T.: Taverna: Lessons in creating a workflow environment for the life sciences in *Concurrency and Computation: Practice and Experience. Grid Workflow Special Issue* 18(10), 1067–1100 (2005)
14. Ptolemy II Website, <http://ptolemy.eecs.berkeley.edu/ptolemyII>
15. Shvaiko, P., Euzenat, J.: A Survey of Schema-Based Matching Approaches. In: Spaccapietra, S. (ed.) *Journal on Data Semantics IV. LNCS*, vol. 3730, pp. 146–171. Springer, Heidelberg (2005)