Agent-Oriented Data Curation in Bioinformatics

Simon Miles

School of Electronics and Computer Science
University of Southampton
Southampton, SO17 1BJ, UK
sm@ecs.soton.ac.uk

Abstract. The practitioners of bioinformatics require increasing sophistication from their software tools to take into account the particular characteristics that make their domain complex. For example, there is a great variation of experience of researchers, from novices who would like guidance from experts in the best resources to use to experts that wish to take greater management control of the tools used in their experiments. Also, the range of available, and conflicting, data formats is growing and there is a desire to automate the many trivial manual stages of in-silico experiments. *Agent-oriented software development* is one approach to tackling the design of complex applications. In this paper, we argue that, in fact, agent-oriented development is a particularly well-suited approach to developing bioinformatics tools that take into account the wider domain characteristics. To illustrate this, we design a *data curation tool*, which manages the format of experimental data, extend it to better account for the extra requirements placed by the domain characteristics, and show how the characteristics lead to a system well suited to an agent-oriented view.

1 Introduction

Bioinformatics is a fast-growing field in which biological data is analysed and shared using software tools. However, due to the field's success, the size and complexity of the data being produced is increasing fast. It also means that new, relatively inexperienced researchers are constantly being recruited. Together, these characteristics make it hard for organisations to ensure that work is being undertaken on the best available data and with the best available tools.

Several strands of research aim to support the bioinformatics community in managing the complexity of their experiments. In our own recent work, we have focused on recording the *provenance* of experimental results [8]. The provenance of a piece of data is the process that led to that data, and *process documentation* is the documentation of processes from which the provenance of data is discovered. We have determined a number of provenance-related use cases in bioinformatics [12] through interviews with scientists, such as the comparison of two experiment runs to determine why results were different, and justifying that the experiment was performed in a valid way to others. We have provided software to record and maintain process documentation in *provenance stores.*

At a more fine-grained level, one way in which tool developers could help the bioinformatics community is to design tools that, from the start, take into account the wider characteristics of the domain. Such characteristics include the wide variety of data formats available and differences in the level of expertise of those using the tools. However, providing such flexibility can substantially add to the complexity of a system.

Agent-oriented software development is an approach to designing complex, flexible applications in terms of *agents* [18]. Here, we follow the definition of agents as autonomous, pro-active, flexible and social entities [11]. In this paper, we argue that agent-oriented software development is well suited to the design of manageable, re-usable software in the bioinformatics domain. We believe this hypothesis is correct because the characteristic demands of the bioinformatics domain fit well with the problems agent-oriented systems attempt to solve.

Our hypothesis is motivated by our work in designing a tool to solve a specific bioinformatics problem: managing the heterogeneous formats of data so that it remains parsable into the future. In this paper, we propose the design of a *data curation tool* that trawls a bioinformatician's provenance store for data in obsolete formats and converts that data to novel formats. The tool has a set of *conversions*, each stating an obsolete format to convert from and a novel format to convert to. We consider both a non-agent based solution and an agent based solution, and show that the properties of the latter are necessary for meeting the wider demands of the bioinformatics domain. Such demands come from the particular characteristics of the domain and two in particular. The first domain characteristic is the *variation in expertise* between novice researchers who cannot accurately determine which data formats may become obsolete and experts who wish to gain greater control over the management of their tools. The second characteristic is the desire to automate the many trivial manual stages of bioinformatics experiments, to give time for answering more valuable research questions. We demonstrate that an agent-oriented development method is well-suited to designing the tool, and that, because the requirements come from the domain rather than being particular to the tool, the approach applies more widely to other bioinformatics tools.

The paper is organised as follows. In Section 2, we define a few key characteristics of the bioinformatics domain. Section 3 describes concepts, such as *provenance* and *workflow*, and implementing technologies that support advanced bioinformatics. We propose a tool that manages the curation of data recorded during in-silico experiments and specify two different designs in Section 4. We then examine how the different designs of the tool map to an agent-oriented view in Section 5, and generalise this to the whole bioinformatics domain. Finally, in Section 6, we infer the benefits of an agent-oriented development approach for bioinformatics tools. The specific contributions of this paper are as follows.

– A (non agent-oriented) design for a tool that provides *data curation* of process documentation in a bioinformatics organisation.
– An agent-oriented view of the tool, in which the properties of the design are mapped to agent attributes.

– A general mapping from the characteristics of the bioinformatics domain to the properties of an agent-based system, leading to the argument that many bioinformatics tools would benefit from agent-oriented development.

2 Bioinformatics Domain Characteristics

The bioinformatics domain has a number of consistently apparent characteristics, due to both the way in which the field has developed and the nature of the biological data being processed. One striking characteristic is the willingness of the community to make the data it has produced through experiment available in large, publicly accessible databases and biological data analysis tools, such as those hosted by the European Bioinformatics Institute [4] and National Center for Biotechnology Information [13]. This has made the rapid development of the field possible. Other pervading characteristics include the rapid increase in new tools and databases that small laboratories are making available, and the complexity and unmanageability that the software involved can reach as analysis scripts and processes are extended and combined.

For this paper, we highlight three other domain characteristics that help to illustrate the usefulness of an agent-oriented approach to bioinformatics tool development. These are named and described below, and referred to throughout the rest of the paper.

Heterogeneous Data Formats There is a wide range of data formats in use, with several formats for any one particular type of data, analysis tools that accept data only in one format and new data formats appearing frequently as new tools are developed.

Variation in Expertise An increasing number of researchers are being employed in bioinformatics organisations, including many that are relatively new to the discipline, coming from biology, computer science, and other fields such as physics. Organisations are a mixture of established, experienced bioinformatics researchers along with novices, who are yet to discover which available tools and data sources are best suited to their problems.

Desire for Automation While bioinformatics is based on software tools, there is a wide range of tasks which require manual effort, including, first, copying and pasting between tools accessed through websites and, second, using spreadsheets to sort and filter results. In consequence, there is a strong community enthusiasm for automating the tedious manual parts of experimental process, so that more time can be spent on the more interesting and valuable research questions.

3 Provenance-Aware Experimentation

In order to accurately describe the purpose and design of our bioinformatics data curation tool, it is first necessary to introduce some key concepts and technologies, and explain how they have been used to enhance a bioinformatician's work in practice.

3.1 Workflow and Grid Computing

Experiments follow particular experimental processes, which can be expressed as *workflows*. A workflow coordinates the data and control flow between multiple *services* to achieve a more complex goal than each can achieve individually. Services, in this case, could take many forms, such as standardised, remotely deployed components such as Web Services, or locally executed Perl or Tcl scripts.

Grid computing [6, 17] allows the enactment of workflows to take advantage of the spare capacity on other processors, so that experiments which were previously infeasible to perform because they would have taken months or years to run on a single machine can be completed in days or hours. Grid computing is made possible by a set of software that manages and provides a consistent view of multiple distributed resources. Commonly used Grid software includes Condor, which can enact distributed workflows defined as *directed acyclic graphs* (DAGs), and the Chimera Virtual Data System [7], which manages the dependencies between data items and can generate a workflow (DAG) that will produce the most recent version of a given data item.

3.2 Provenance

The provenance of a piece of data is the process that led to that data, and *process documentation* is the documentation of that process [10]. Therefore, the provenance of an experimental result is represented by the documentation of the experiment process. Recording process documentation as an experiment is performed allows many valuable questions to be asked afterwards, such as the following.

- Given that the results of two runs of an experiment were different, was this caused by a difference in the input data or because different versions of analysis tools were used?
- What input data contributed to the production of this result?
- Was any data source used in this experiment licensed such that the result cannot be patented?
- Did the experimental process follow the plan as originally conceived?

Process documentation is recorded by each client and service in a workflow. It is stored in a *provenance store*, which provides structured and persistent storage of process documentation. Each service uses a provenance store it considers reliable, which means that potentially each service may record process documentation to a different provenance store. This situation is shown in Figure 1. However, this is the logical view of provenance recording and there is no reason why multiple services cannot use the same physical provenance store.

A simplified view of the contents of the provenance store contents after recording has taken place is shown in Figure 2. For each experiment run, a set of interactions are documented. Each interaction consists of (at least) a request and a response message. Each message contains data: the request contains the input arguments of the service request and the response contains the output

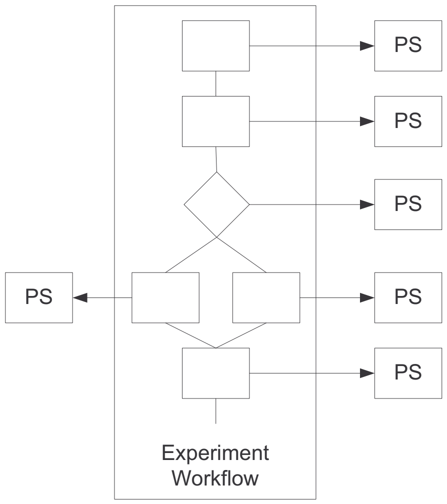

Fig. 1. A workflow comprised of multiple services, with each service recording to a different provenance store

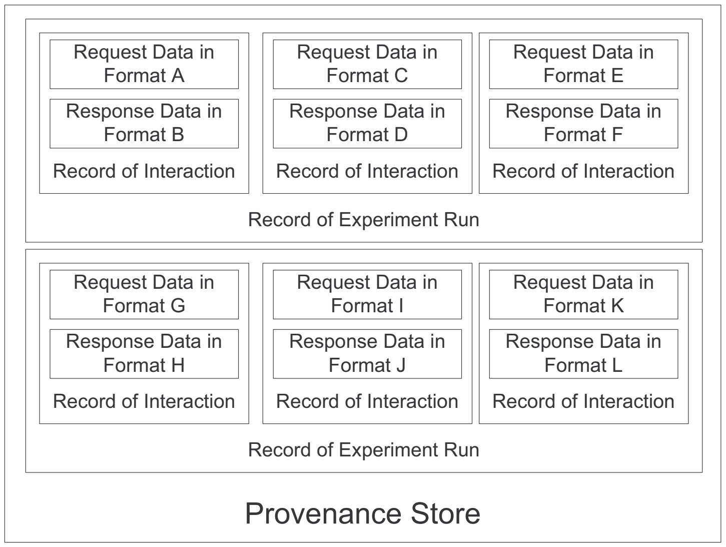

Fig. 2. The contents of a provenance store after a workflow has executed

results of the service's processing. The format of each piece of data is explicitly recorded in the process documentation.

3.3 Protein Complexity Experiment

As part of the PASOA project [15], we have worked with bioinformaticians to make use of Grid computing resources, to record process documentation and apply it in solving a number of use cases. The aim of the particular experiment we focus on is to analyse the information content of different amino acid groupings, and was applied to a range of protein sequences from the RefSeq database [14]. By deploying the experiment in the Virtual Data System, described in the section above, we allow the bioinformatician to potentially utilise a vast range of international resources, making the full experiment feasible in a reasonable timescale.

The experiment includes an instantiation of all the key concepts. The experiment is encoded as a workflow (Condor DAG) of services (mostly Tcl and UNIX shell scripts) and process documentation is recorded into a provenance store during or after an experiment's enactment. The data exchanged between services uses bioinformatics formats (in particular, FASTA [5] is the original encoding of the protein sequences). A single provenance store, implemented as a Web Service, was used by all services. This work provides a convenient testbed for illustrating the development of bioinformatics tools and assessing an agent-oriented approach. The details of the experiment are beyond the scope of this paper but are explained fully in [9].

4 Data Curation Tool

Understanding the provenance of a result means, in part, examining the experimental process' input and intermediate data. However, domain characteristic **Heterogeneous Data Formats** means that some data formats will become *obsolete*. An example of a data format that has become obsolete in reality is the Staden data format previously used by the Staden project [16]. Therefore, when examining the provenance of a result long after an experiment has been performed, it is possible that no tool will be available to correctly interpret the data. This is a problem that is solved by good *curation*. Amongst other goals, curation should ensure that data always remains available in at least one format that can be parsed. In practice, this means translating data from obsolete formats to *novel* formats, before a situation occurs where no tools available are capable of parsing the obsolete format.

We propose a *data curation tool* that trawls the provenance store for data in obsolete formats and converts that data to novel formats. The tool will have a set of *conversions*, each stating an obsolete format to convert from and a novel format to convert to.

4.1 Scenario

To provide context for the tool's use, we describe a sample scenario which illustrates the domain characteristics described in Section 2. We define the social structure of a bioinformatics organisation and the resources available to each researcher in that organisation. We assume that the researchers are all performing experiments with similar properties to the Protein Complexity Experiment described in Section 3.3.

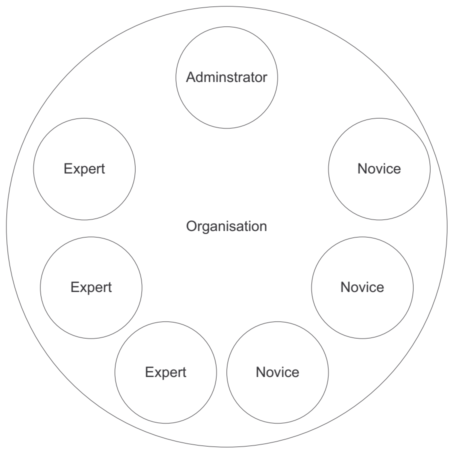

Fig. 3. An example human role structure in an organisation

Figure 3 shows the relevant human role structure in the testbed organisation. There are several researchers, some *experts*, some *novices*. Experts are researchers who have plenty of experience of bioinformatics resources and opinions about which are better or worse. Novices are researchers without this bioinformatics experience and knowledge. They may take the lead from the organisation to which they belong, and the experts within it. An *administrator* coordinates the dissemination of information and advice amongst researchers.

An individual bioinformatics researcher has access to resources of the form shown in Figure 4. The researcher has a set of experiments, expressed as workflows, that they are modifying and performing to test hypotheses. The inputs and outputs of workflows are stored in a *data store*, e.g. a database, and documentation of the process which led to each experiment result is kept in a *provenance store*. In our case, this means storing a record of the execution of each workflow run.

4.2 Tool Designs

The workflow for performing data curation is shown in Figure 5. For each conversion in a conversion list, from an obsolete format to a novel format, that the

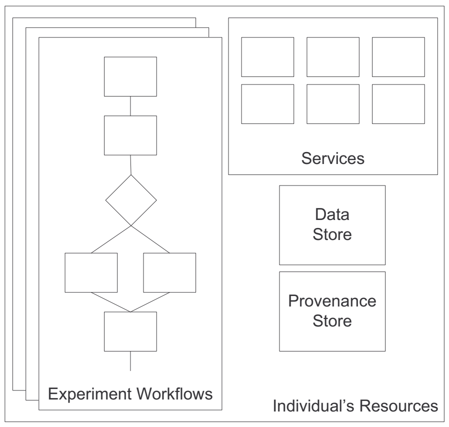

Fig. 4. The resources available to each bioinformatician

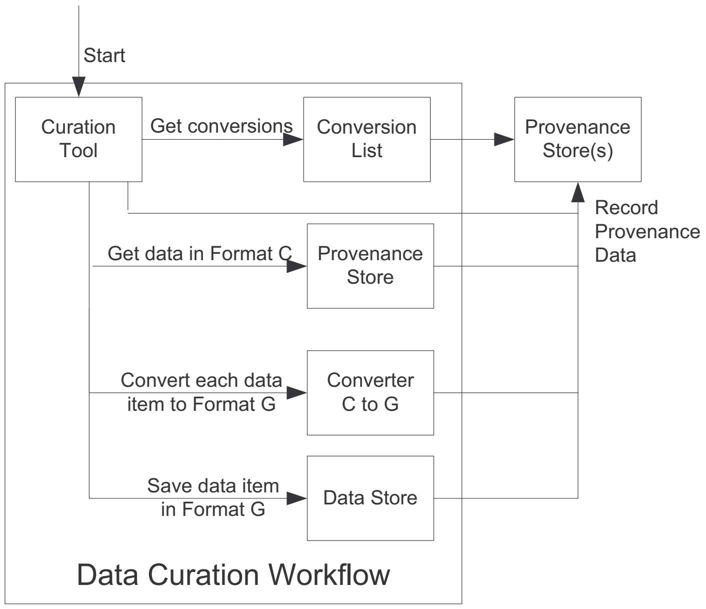

Fig. 5. A process for converting data in obsolete formats to novel formats

researcher has requested, the curation tool will extract all data in the obsolete format. For each data item, the curation tool will then use an appropriate converter to translate the data item to the novel format. Several suites of conversion tools are available, including those contained in BioJava [1] and BioPerl [2]. Finally, the converted data is stored in the researcher's data store. On running the data curation workflow, all interactions are recorded in provenance stores (in our testbed's case, a single provenance store). Because the chain of interactions documents a workflow run in which data in the obsolete format is converted to data in the novel format, the researcher can later discover the original data that some novel data is a conversion of, and then the experiment that produced that original data.

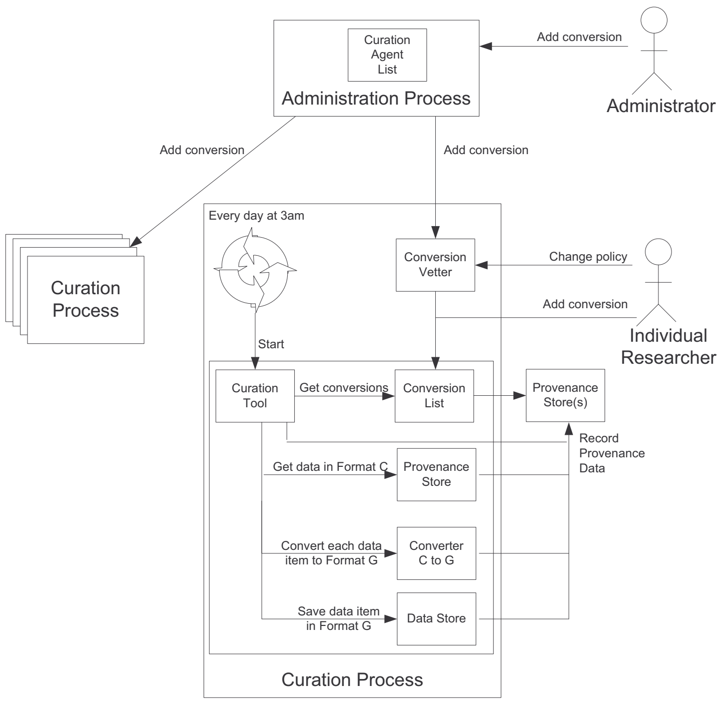

Fig. 6. A personalised, automated and cooperative data curation process

Due to domain characteristic **Variation in Expertise** (specified in Section 2), new researchers may not know which data formats are becoming obsolete and which formats are likely to remain useful. Also, due to domain characteristic **Desire for Automation**, we would prefer that the researcher does not have to repeatedly initiate the tool's use. Therefore, we can envisage an extension to the tool, shown in Figure 6. In this extension, the organisation, via the administrator, sends recommended conversions to the researcher's curation tools. The conversions are vetted against an individual researcher's policy, so that novices' tools automatically apply the recommended conversions while experts' tools ignore the advice and use only conversions determined by the expert researcher.

5 Agent-Oriented Systems

A software system can be viewed as a set of interacting *agents* if it has the following appropriate *properties*.

- Agents have *localised control*, so entities within the system must be perceived to make decisions on the basis of the information immediately available to them.
- Agents have *social ability*, so the same entities must send communications between each other.
- Agents are *pro-active*, acting to achieve *goals* on their own initiative when the context is appropriate for them to do so.

As a consequence of *localised control* and *social ability*, we note that agents are able to refuse commands from other agents within the system, if the local information determines this to be the best decision.

In the sections below, we consider the difference between systems that can be viewed as agent-oriented and those that cannot. We will first examine the data curation tool, and then the bioinformatics domain more generally.

5.1 Non-Agent-Oriented Data Curation

The initial design of our data curation tool was not readily mapped to a set of autonomous agents. By itself, Figure 5 describes a system that does not have agent properties. Specifically, each entity merely follows the commands given by other entities so there is no *localised control*, reducing the social communications to invocations. Each entity only acts in reaction to invocations and the workflow as a whole has to be externally triggered, so there is no *pro-activity.*

5.2 Agent-Oriented Data Curation

In constrast, the second, more powerful version of the tool, contains entities readily mapped to agents. Mapping the agent properties to the design in Figure 6 is valid because each curation and administration process has *localised control*, deciding on whether and when to perform suggested conversions or suggest new conversions respectively. A curation process is initiated *pro-actively* when the context is most suitable (at an appointed time of night, though the context could be refined to be more sensistive to other use of the researcher's resources). The adminstration and curation processes are social, in that conversions are suggested from the former to the latter.

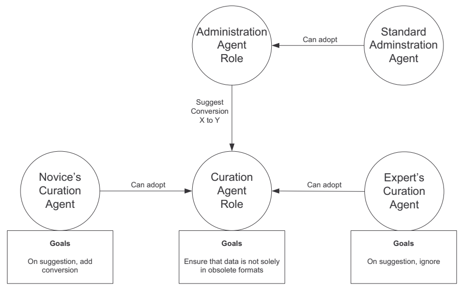
Administration Agent Role
Can adopt
Standard Adminstration Agent
Suggest Conversion X to Y
Novice's Curation Agent
Can adopt
Curation Agent Role
Can adopt
Expert's Curation Agent
Goals
On suggestion, add conversion
Goals
Ensure that data is not solely in obsolete formats
Goals
On suggestion, ignore

Fig. 7. An agent-oriented design of the data curation tool

Figure 7 is an agent-oriented view of the tool designed in Figure 6. We divide the tool's operation into two *roles*: administration agent role and curation agent role, corresponding to the adminstration process and curation process in Figure 6. Each role can be fulfilled by different concrete agents at different times or for different researchers. Every administration agent can send suggestions of conversions to all curation agents and every curation agent has the goal to ensure that any data in the researcher's provenance store is not kept solely in an obsolete format.

A standard administration agent will adopt the administration agent role and perform the administration process given in Figure 6. A novice's curation agent performs the curation process and accepts all suggestions from the administration agent. An expert's curation agent performs the curation process but ignores suggestions from the administration agent. Because the definition is given in terms of roles, it is clear that as a novice becomes more experienced, they can change the behaviour of their curation agent.

5.3 Domain Characteristics and Agent Properties

Examining the domain in general, we can see that the domain characteristics given in Section 2 provide requirements for exactly those properties that an agent-based system provides.

First, due to the **Variation in Expertise** domain characteristic, many researchers will benefit from using the experience of experts within their organisation. Each researcher has control over their own resources, which their tools act on. Therefore, tools will be improved by *social ability*, where information from one person can affect the operation of another person's tool.

Also due to the **Variation in Expertise** domain characteristic, some researchers will have greater expertise than others, so should have control over how information from others are applied by their tools. Tools will be better for having *localised control*, where decisions are made that match the preferences and resources of individual researchers even when information and requests for action come from external sources.

Due to the **Desire for Automation** domain characteristic, researchers wish to reduce the amount of time spent initiating trivial manual processes. *Pro-active* tools are always applied in a given context should automatically run in that context without the user having to manually initiate this.

In Table 1, we summarise how the properties of agent-oriented systems meet the requirements of named domain characteristics and how this property is instantiated in the data curation tool.

System Property	**Domain Characteristic**	**Instantiation**
Social ability	Variation in Expertise, Hetereogeneous Data Formats	Organisation suggests data format conversions to researchers
Localised control	Variation in Expertise	Ability to set whether suggested conversions are used
Pro-activity	Desire for Automation	Automatic curation every night

Table 1. Mapping of agent properties to bioinformatics tool characteristics

6 Conclusions

The fact that the designs of good bioinformatics tools, i.e. ones that take into account the wider characteristics of the domain, are readily mapped to agent-oriented systems is not merely an interesting fact. We can go one stage further and say that, because bioinformatics tools that can be modelled in terms of agents will be better than those that cannot, such tools would be more likely to be well designed if developed in a way that modelled the system in terms of agents from the start. In this way, the beneficial properties of an agent-based system are *assumed*, making the design both simpler and less likely to exclude the beneficial properties. A range of agent-oriented development approaches exist (a broad survey can be found in [18]), each taking agents, with their inherent properties, as the building blocks of the system. Also, approaches such as that described by Corradini et al. [3], can be applied to wrap and integrate multiple agent-oriented tools.

In this paper, we have presented the design of a bioinformatics data curation tool, converting data in obsolete formats to formats more likely to be interpretable in the future. The tool was extended to take into account characteristics of the bioinformatics domain: the mixture of novices and experts within an organisation and the push for automation of process initiation. We demonstrated that the extra requirements that this extension demanded were exactly those met by a system that can be viewed in terms of autonomous agents.

The properties of agents, and the abstraction that agent-oriented development provides by assuming them, are those required by a bioinformatics tool to ensure it can be re-used and relied upon by a range of researchers. The *social ability* of agents means that novices can take advantage of the expertise of more experienced researchers. The *localised control* of agents means that tools can be personalised to the researcher and experienced researchers can have control over their use. Finally, the *pro-activity* of agents means that tasks are performed automatically in the appropriate context, rather than requiring the scientist to manually trigger them. In conclusion, agent-oriented software development is very applicable to the development of bioinformatics tools.

7 Acknowledgements

This research is funded by the PASOA project (EPSRC GR/S67623/01).

References

1. BioJava. http://www.biojava.org/, 2006.
2. BioPerl. http://www.bioperl.org/, 2006.
3. Flavio Corradini, Leonardo Mariani, and Emanuela Merelli. An agent-based approach to tool integration. *International Journal on Software Tools for Technology Transfer (STTT)*, 6(3):231–244, August 2004.
4. European Bioinformatics Institute. http://www.ebi.ac.uk/, 2006.
5. FASTA Format Description. http://en.wikipedia.org/wiki/FASTA_format, 2006.
6. I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of Supercomputer Applications*, 15(3):200–222, 2001.
7. I. Foster, J. Vockler, M. Wilde, and Y. Zhao. The virtual data grid: A new model and architecture for data-intensive collaboration. In *In Proc. of the CIDR 2003 First Biennial Conference on Innovative Data Systems Research*, January 2003.
8. Paul Groth, Michael Luck, and Luc Moreau. A protocol for recording provenance in service-oriented grids. In *Proceedings of the 8th International Conference on Principles of Distributed Systems (OPODIS'04)*, Grenoble, France, December 2004.
9. Paul Groth, Simon Miles, Weijian Fang, Sylvia C. Wong, Klaus-Peter Zauner, and Luc Moreau. Recording and Using Provenance in a Protein Compressibility Experiment. In *Proceedings of the 14th IEEE International Symposium on High Performance Distributed Computing (HPDC'05)*, July 2005.
10. Paul Groth, Simon Miles, Victor Tan, and Luc Moreau. Architecture for provenance systems. Technical report, University of Southampton, October 2005.
11. N. R. Jennings, K. Sycara, and M. Wooldridge. A Roadmap of Agent Research and Development. *International Journal of Autonomous Agents and Multi-Agent Systems*, 1(1):7–38, 1998.
12. Simon Miles, Paul Groth, Miguel Branco, and Luc Moreau. The requirements of recording and using provenance in e-science experiments. Technical report, School of Electronics and Computer Science, University of Southampton, UK, January 2005.
13. National Center for Biotechnology Information. http://www.ncbi.nlm.nih.gov/, 2006.
14. NCBI Reference Sequences. http://www.ncbi.nlm.nih.gov/RefSeq/, 2006.
15. Provenance-Aware Service-Oriented Architecture. http://www.pasoa.org, 2006.
16. Staden DNA sequence analysis tool. http://staden.sourceforge.net/, 2006.

17. Robert Stevens, Hannah Tipney, Chris Wroe, Tom Oinn, Martin Senger, Phillip Lord, Carole Goble, Andy Brass, and May Tassabehji. Genome science performed with e-science tools. In *Proceedings of the UK e-Science All Hands Meeting 2004*, Nottingham, UK, August 2004.
18. G. Weiss. Agent orientation in software engineering. *Knowledge Engineering Review*, 16(4):349–373, 2002.