Common motifs in scientific workflows: An empirical analysis

Daniel Garijo, Pinar Alper, Khalid Belhajjame, Oscar Corcho, Yolanda Gil, Carole Goble

Common Motifs in Scientific Workflows: An Empirical Analysis

Daniel Garijo*, Pinar Alper †, Khalid Belhajjame†, Oscar Corcho*, Yolanda Gil‡, Carole Goble†
*Ontology Engineering Group, Universidad Politécnica de Madrid. {dgarijo, ocorcho}@fi.upm.es
†School of Computer Science, University of Manchester. {alperp, khalidb, carole.goble}@cs.manchester.ac.uk
‡Information Sciences Institute, Department of Computer Science, University of Southern California. gil@isi.edu

***Abstract*—While workflow technology has gained momentum in the last decade as a means for specifying and enacting computational experiments in modern science, reusing and repurposing existing workflows to build new scientific experiments is still a daunting task. This is partly due to the difficulty that scientists experience when attempting to understand existing workflows, which contain several data preparation and adaptation steps in addition to the scientifically significant analysis steps. One way to tackle the understandability problem is through providing abstractions that give a high-level view of activities undertaken within workflows. As a first step towards abstractions, we report in this paper on the results of a manual analysis performed over a set of real-world scientific workflows from Taverna and Wings systems. Our analysis has resulted in a set of *scientific workflow motifs* that outline i) the kinds of data intensive activities that are observed in workflows (*data oriented motifs*), and ii) the different manners in which activities are implemented within workflows (*workflow oriented motifs*). These motifs can be useful to inform workflow designers on the good and bad practices for workflow development, to inform the design of automated tools for the generation of workflow abstractions, etc.**

I. Introduction

Scientific workflows have been increasingly used in the last decade as an instrument for data intensive scientific analysis. In these settings, workflows serve a dual function: first as detailed documentation of the method (i. e. the input sources and processing steps taken for the derivation of a certain data item) and second as re-usable, executable artifacts for data-intensive analysis. Workflows stitch together a variety of data manipulation activities such as data movement, data transformation or data visualization to serve the goals of the scientific study. The stitching is realized by the constructs made available by the workflow system used and is largely shaped by the environment in which the system operates and the function undertaken by the workflow.

A variety of workflow systems are in use [10] [3] [7] [2] serving several scientific disciplines. A workflow is a software artifact, and as such once developed and tested, it can be shared and exchanged between scientists. Other scientists can then reuse existing workflows in their experiments, e.g., as sub-workflows [17]. Workflow reuse presents several advantages [4]. For example, it enables proper data citation and improves quality through shared workflow development by leveraging the expertise of previous users. Users can also re-purpose existing workflows to adapt them to their needs [4]. Emerging workflow repositories such as myExperiment [14] and CrowdLabs [8] have made publishing and finding workflows easier, but scientists still face the challenges of re-use, which amounts to fully understanding and exploiting the available workflows/fragments. One difficulty in understanding workflows is their complex nature. A workflow may contain several scientifically-significant analysis steps, combined with various other data preparation activities, and in different implementation styles depending on the environment and context in which the workflow is executed. The difficulty in understanding causes workflow developers to revert to starting from scratch rather than re-using existing fragments.

Through an analysis of the current practices in scientific workflow development, we could gain insights on the creation of understandable and more effectively re-usable workflows. Specifically, we propose an analysis with the following objectives:

1) To reverse-engineer the set of current practices in workflow development through an analysis of empirical evidence.
2) To identify workflow abstractions that would facilitate understandability and therefore effective re-use.
3) To detect potential information sources and heuristics that can be used to inform the development of tools for creating workflow abstractions.

In this paper we present the result of an empirical analysis performed over 177 workflow descriptions from Taverna [10] and Wings [3]. Based on this analysis, we propose a catalogue of *scientific workflow motifs*. Motifs are provided through i) a characterization of the kinds of data-oriented activities that are carried out within workflows, which we refer to as *data-oriented motifs*, and ii) a characterization of the different manners in which those activity motifs are realized/implemented within workflows, which we refer to as *workflow-oriented motifs*. It is worth mentioning that, although important, motifs that have to do with scheduling and mapping of workflows onto distributed resources [12] are out the scope of this paper.

The paper is structured as follows. We begin by providing related work in Section II, which is followed in Section III by brief background information on Scientific Workflows, and the two systems that were subject to our analysis. Afterwards we describe the dataset and the general approach of our analysis. We present the detected scientific workflow motifs in Section IV and we highlight the main features of their distribution

across the analyzed workflows in section V. Finally, we distill the main findings of our study (in Section VI) and conclude by outlining our future plans in Section VII.

II. Related Work

Our motifs could be observed as higher-level patterns observed over scientific workflow datasets. "Workflow patterns" have been extensively studied in the last two decades [15]. Work in this area is primarily focused on outlining the inventory of workflow development constructs provided by different workflow languages and the ways of combining those constructs. Classification models have also been developed to detect additional patterns in structure, usage and data [13]. Scientific workflows are characterized by the lack of complex control constructs, where the order of execution is determined by the availability of data. As observed by a recent study [9], these systems largely support data-flow patterns[1] and even bring-about new ones with their varied handling of data tokens. Data-flow patterns outline ways of managing data resources during workflow execution, such as visibility, data interaction and transfer. These patterns are orthogonal to the scientific data-oriented function undertaken by the processing steps i.e. our motifs. In this regard our work complements the workflow patterns research with its focus on pinpointing characteristic data-oriented activities, rather than an analysis of workflow languages, token handling or data dependencies. Our work is also based on an analysis of empirical evidence of how data-intensive activities have been implemented against different environments, rather than specifying what is theoretically possible with the given constructs and a few examples.

Another work, somewhat closer to our study in spirit, is an automated analysis of workflow scripts from the Life Science domain [16]. This work aims to deduce the frequency of different kinds of technical ways of realizing workflow steps (e.g. service invocations, local "scientist-developed" scripting, local "ready-made" scripts) etc.). [16] also drills down into the category of local ready-made scripts, to outline a functional breakdown of their activity categories such as data access or data transformation. While this provides an insight into the kind of activities undertaken in workflows, it is not representative as it is restricted only to the local service task types, which are only one of many ways to process data in scientific workflows. Our approach is different from this work as it is based on a manual analysis, which allows us to detect activities in many technical realizations (not just local services). We're also categorizing at a much finer grain some of their generic "Conversion" and "Operation" activities (e.g. into filtering, splitting and data cleaning and so on).

Problem Solving Methods is another area of related work. PSMs describe the reasoning process to achieve the goal of a task in an implementation and domain-independent manner [11]. Some libraries aim to model the common processes in scientific domains [5], although they focus in *in vitro* experiments instead of scientific workflows (*in silico* data oriented experiments).

[1]http://www.workflowpatterns.com/patterns/data/

III. Preliminaries

For the purposes of our analysis, we use workflows specified and used in the Taverna [10] and Wings [3] workflow systems. The choice of these two systems is due to :

1) The similarity in the kinds of workflow modeling constructs they provide. When compared to other scientific workflow systems both Taverna and Wings are observers of the pure data-flow paradigm, they provide no explicit mechanism for even the most basic control constructs such as conditionals and looping, while there are implicit ways of doing this.
2) The variety of execution environments in which these systems operate. While Taverna provides users with a means to specify workflows that mainly make use of autonomous third party services, Wings provides an environment in which users have relatively more control over the resources and the analysis operations that scientists use in their experiments. Within Wings, analysis tools and data artifacts are made part of the environment prior to workflow design, workflows are then composed of using these encapsulated software/tools.

A. Taverna and Wings workflow systems

Taverna[2] is an open-source workflow system, which has been initially devised for *in-silico* experimentation in the life-sciences. Recently, it has been extended to include several other domains including Biodiversity, Chemistry and Astronomy. Taverna is characterized with its design to operate in an open-world setting with the ability to access remote third party web services "in the wild" and compose them into data-intensive pipelines. The Taverna environment (in its default configuration) is characterized by its open-typing approach, where the only types that are supported are singleton values of Strings, byte arrays and nested collections of singletons.

Wings[3] uses semantic representations to describe the constraints of the data and computational steps in the workflow. Wings can reason about the constraints of the workflow components (steps) and the characteristics of the data and propagate them through the workflow structure. Wings also separates the physical and the logical layers of the workflow at distinct levels. It is typically configured to use Pegasus [1] as execution engine, which handles distributed execution, data movement and optimization.

B. Description of the sets of workflows analyzed

For our analysis, we have chosen 177 heterogeneous workflows in a variety of domains. We have analyzed most of the Wings workflow set (66 workflows), and part of the Taverna set (111 out of 874 workflows). For Wings, we have analyzed workflows from Drug Discovery, Text Mining and Genomics domains. For Taverna we have analyzed workflows that were

[2]http://www.taverna.org.uk/

[3]http://www.wings-workflows.org

available in myExperiment [14], by inspecting the official myExperiment groups with a sufficient number of workflows in them. We have looked at Cheminformatics, Genomics, Astronomy, Biodiversity, Geo-Informatics and Text Mining workflows. We have also included a group, named IST600, that includes workflows from novice scientists as part of a post-graduate course on scientific workflows. The distribution of workflows to domains is not even, as it is also the case in myExperiment. The numbers of workflows analyzed from each domain can be seen in Table I:

TABLE I
NUMBER OF TAVERNA (T) AND WINGS (W) WORKFLOWS ANALYZED

Domain	Number of workflows
DRUG DISCOVERY	7 (W)
ASTRO	11 (T)
BIODIV	12 (T)
CHEMINFORMATICS	7 (T)
GENOMICS	69 (38 (T) + 31 (W))
GEO-INFORMATICS	6 (T)
IST600	21 (T)
TEXTANALYSIS	44 (11 (T) + 31 (W))
TOTAL	177

C. Approach for workflow analysis

We have performed a bottom-up manual analysis comprised of two orthogonal dimensions, outlining what kind of data-oriented activity is being undertaken by a workflow step and how that activity has been realized. For example, a visualization step (data oriented activity) can be realized in different ways: via a stateful multi-step invocation, through a single stateless invocation (depending on the environmental constraints and nature of the services), or via a sub-workflow.

We have not outlined the possible motifs we predict to occur upfront, instead we have built up the motif list as we progressed with the inspections. For each workflow, we have recorded the number of occurrences of motifs. We illustrate some of the motifs via two workflows from our data set. Figure 1 outlines a Wings workflow that makes comparison of the input protein structures and drugs, sorts and merges the results obtained. Figure 2 outlines a Taverna workflow that performs a 2-D time alignment over protein data through the use of an external service. Due to space considerations, we highlight only some of the motifs in the examples.

IV. SCIENTIFIC WORKFLOW MOTIF CATALOGUE FOR ABSTRACTING WORKFLOWS

This section introduces details on the scientific workflow motifs detected in our analysis. An overview is provided in Table II.

A. Data-Oriented Motifs

1) Data Retrieval: Workflows exploit heterogeneous data sources, remote databases, repositories or other web resources exposed via SOAP or REST services. Scientific data deposited in these repositories are retrieved through query and retrieval steps inside workflows. Certain tasks within the workflow are responsible for retrieving data from such external source into the workflow environment. We also observed that certain data integration workflows contain multiple linked retrieval steps, being essentially parameterized data integration chains.

2) Data Preparation: Data, as it is originally retrieved, may need several transformations before being able to be used in a workflow step. The most common activities that we have detected in our analysis are:

- Format Transformation: Heterogeneity of formats in data representation is a known issue in many scientific disciplines. Workflows that bring together multiple access or analysis activities usually contain steps for format transformations. These steps named "Shims" [6] typically preserve the contents of data, while converting its representation format.
- Input Augmentation and Output Splitting: Data access and analysis steps that are handled by external services or tools typically require well formed query strings or structured requests as input parameters. Certain tasks in workflows are dedicated to the generation of these queries through an aggregation of multiple parameters. The reverse operation occurs for output processing. Outputs of data access or analysis steps could be subject to data extraction or splitting to allow the conversion of data from the service specific format to the workflows internal data carrying structures (nested lists). An example of this is provided in the workflow of Figure 2, for each service invocation (e.g. *getJobState*) there are steps (e.g. *getJobState_Input*) that are responsible for creating the correctly formatted inputs for the service and output splitting steps (e.g. *getJobState_output*) that are responsible for parsing the results returned from the service.
- Data Organization (Merging, Grouping, Sorting, Filtering): The datasets brought into a pipeline may not be subject to analysis in their entirety. Data could further be filtered, sampled or could be subject to extraction of various subsets. In addition to filtering, certain tasks are dedicated to merging data sets created by different branches of workflows. Sorting or grouping results is also observed under this category. Examples of merge and sort activities are given in the Wings workflow in Figure 1, where the results of both branches of the workflow are first sorted creating different files that are then merged for presenting a single output result.

3) Data Movement: Certain analysis activities that are performed via external tools or services require the submission of data to a location accessible by the service/tool (i.e., a web or a local directory respectively). In such cases the workflow contains dedicated step(s) for the upload/transfer of data to these locations. The same applies to the outputs, in which case a data download/retrieval step is used to chain the data to the next steps of the workflow. The data deposition of the workflow results to a specific server would also be included in this category. In Figure 2, the *DataUpload* and *DownloadResults* steps ship data to the server on which the analysis will be done, and also retrieve back the results via a

dedicated download step.

4) Data Cleaning/Curation: We have observed the steps for cleaning and curating data as a separate category from data preparation and filtering. Typically these steps are undertaken by sophisticated tooling/services, or by human interactions. A cleaning/curation step essentially preserves and enriches the content of data (e.g., by a user's annotation of a result with additional information, detecting and removing inconsistencies on the data, etc.).

5) Data Analysis: This motif refers to a rather broad category of tasks in diverse domains. An important number of workflows are designed with the purpose of analyzing different features of input data, ranging from simple comparisons between the datasets to complex protein analysis to see if two molecules can be docked successfully. An example is given in the workflow of Figure 2 with a processing step named *warp2D*, and the steps named *SMAPV2* in Figure 1 with a ligand binding sites comparison of the inputs.

6) Data Visualization: Being able to show the results is as important as producing them in some workflows. Scientists use visualizations to show the conclusions of their experiments and to take important decisions in the pipeline itself. Therefore certain steps in workflows are dedicated to generation of plots and graph outputs from input data.

B. Workflow-Oriented Motifs

We divide these motifs in two different groups, depending on whether motifs are observed *within* or *among* workflows.

Intra workflow motifs:

1) Stateful/asynchronous and stateless/synchronous invocations: Certain activities such as analysis or visualizations could be performed through interaction with stateful (web) services that allow for creation of jobs over remote grid environments. These are typically performed via invocation of multiple operations at a service endpoint. An example is given in the workflow of Figure 2, where the service invocations *warp2D*, *getJobStatus*, and *DownloadResults* are consecutively executed in order to be able to perform one analysis via an external service. On the other hand stateless activities require a single step for service or tool invocation.

2) Internal macros: this category refers to those groups of steps in the workflow that correspond to repetitive patterns of combining tasks. An example can be seen in Figure 1, where there are two branches of the workflow performing very similar operations (SMAP and Sorting).

3) Human interactions versus computational steps: We have observed that scientific workflows systems increasingly make use of human interactions to undertake certain activities within workflows. Data curation and cleaning are typical examples of such activities.

Inter Workflow Motifs:

1) Atomic workflows: Our review has shown that a significant number of workflows perform an atomic unit of functionality, which effectively requires no sub-workflow usage. Typically these workflows are designed to be included in other workflows. Atomic workflows are the main mechanism of modularizing functionality wthin scientific workflows.

TABLE II
SCIENTIFIC WORKFLOW MOTIFS

- Data-Oriented Motifs
  - Data Retrieval
  - Data Preparation
    - Format Transformation
    - Input Augmentation and Output Splitting
    - Data Organisation
  - Data Analysis
  - Data Curation/Cleaning
  - Data Moving
  - Data Visualisation
- Workflow-Oriented Motifs
  - Intra-Workflow Motifs
    - Statefull (Asynchronous) Invocations
    - Stateless (Synchronous) Invocations
    - Internal Macros
    - Human Interactions
  - Inter-Workflow Motifs
    - Atomic Workflows
    - Workflow Overloading
    - Composite Workflows

2) Workflow overloading: Our analysis has shown that authors tend to deliver multiple workflows with same functionality, but operating over different input parameter types. An example is performing an analysis over a String input parameter, or performing it over the contents of a specified file. Overloading is a direct response to the heterogeneity of environments in which workflows are used.

3) Composite workflows: The usage of sub-workflows appears as a motif for exploiting modular functionality from multiple workflows. This motif refers to all those workflows that have one or more sub-workflows included in them (in some cases, these sub-workflows offer different views of the global workflow). Our review has provided empirical evidence of adoption of this motif in Taverna and Wings systems.

V. RESULTS

In this section, we report on the frequencies of the data- and workflow-oriented motifs within workflows, and discuss how they are correlated with the workflow domains[4].

Figure 3 illustrates the distribution of data-oriented motifs across the domains. The analysis of this figure shows the predominance of the data preparation motif, which constitutes more than 60% of the data preparation motifs in the majority of domains. This is an interesting result as it implies that data preparation steps are more common than any other category of activity, specifically those that perform the main (scientifically-significant) functionality of the workflow. The abundance of these is one major obstacle for understandability. Figure 3 also demonstrates that within domains, such as Genomics, Astronomy or Biodiversity, where curated common scientific databases exist, workflows are used as data retrieval clients against these databases.

Drilling down to Data Preparation, Figure 4 shows the dominance of Input Augmentation and Output Splitting motifs

[4]Results available at http://www.myexperiment.org/packs/310.html

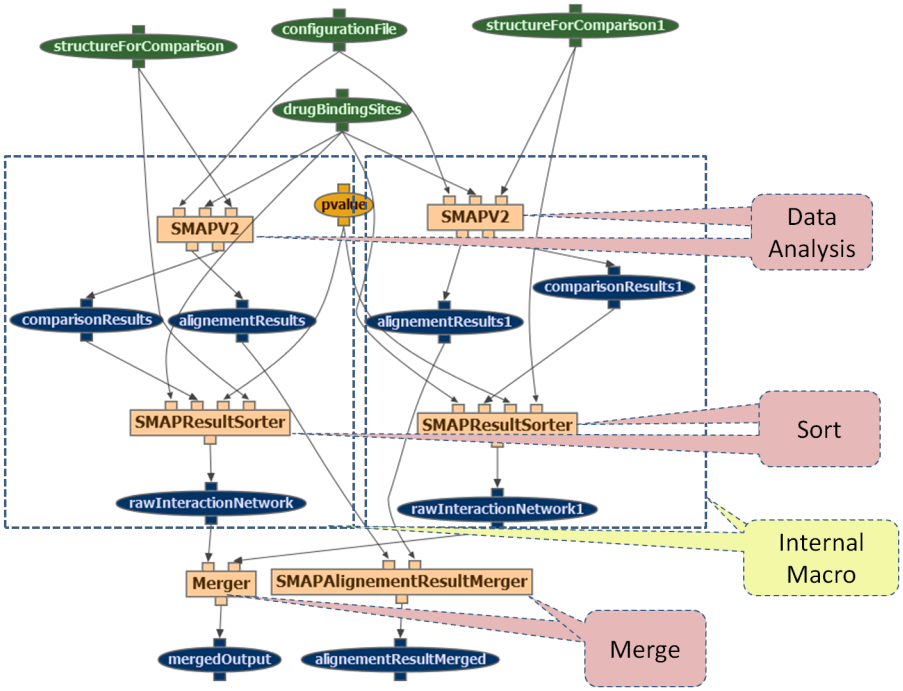

Fig. 1. Sample Motifs in a Wings Workflow for drug discovery. A comparison analysis is performed on two different input datasets (SMAPV2). The results are then sorted (SMAPResultSorter) and finally merged (Merger, SMAPAlignementResultMerger).

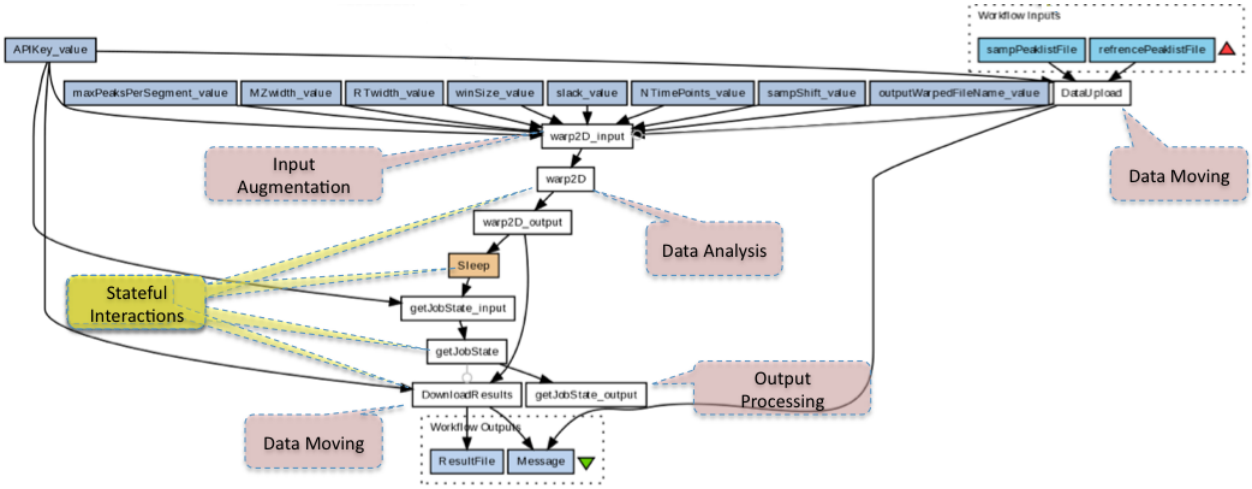

Fig. 2. Sample Motifs in a Taverna Workflow for Functional Genomics. The workflow transfers data files containing proteomics data to a remote server and augments several parameters for the invocation request. Then the workflow waits for job completion and inquires about the state of the submitted warping job. Once the inquiry call is returned the results are downloaded from the remote server.

for most domains. These activities can be seen as adapters that help plugging data analysis capabilities into workflows. Their absence in Drug Discovery can be attributed to the fact that it is a Wings domain, which points-out to a more systematic difference between the Taverna and Wings environments. Figure 4 also demonstrates how the existence of a widely used common data structure for a domain, in this case Astronomy[5], reduces the need for (domain-specific) data transformations in workflows.

[5]http://www.ivoa.net/Documents/VOTable/

As displayed in the comparative Figure 5 for the genomic domain, in Wings input augmentation and output splitting steps are much less required (25% vs 60%) as the inputs are strongly typed and the data analysis steps are pre-designed to operate over typed data. Within Figure 6 we observe that Wings workflows do not contain any data retrieval or movement steps as data is pre-integrated into the workflow environment (data shipping activities are carried out behind the scenes by Wings's scheduling infrastructure) whereas in Taverna the workflow carries dedicated steps for querying

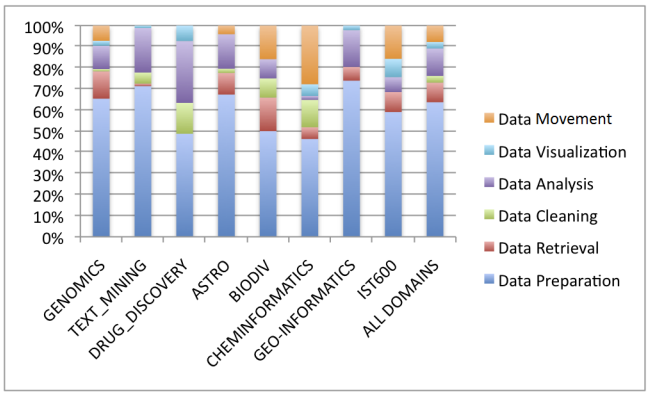

Fig. 3. Distribution of Data-Oriented Motifs per domain

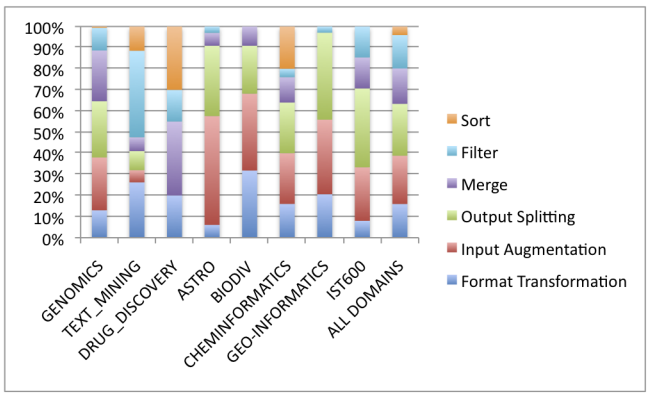

Fig. 4. Distribution of Data Preparation motifs per domain

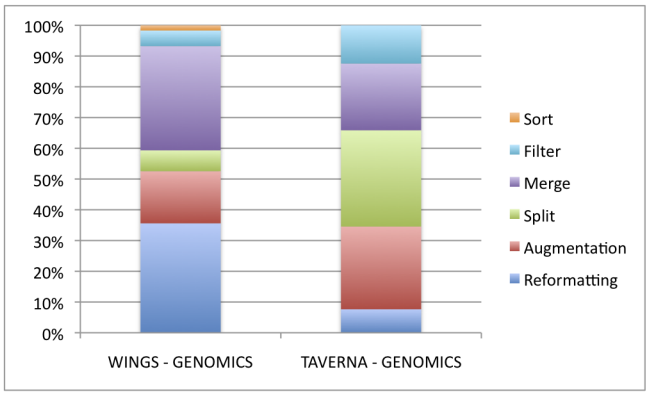

Fig. 5. Data Preparation Motifs in the Genomics Workflows

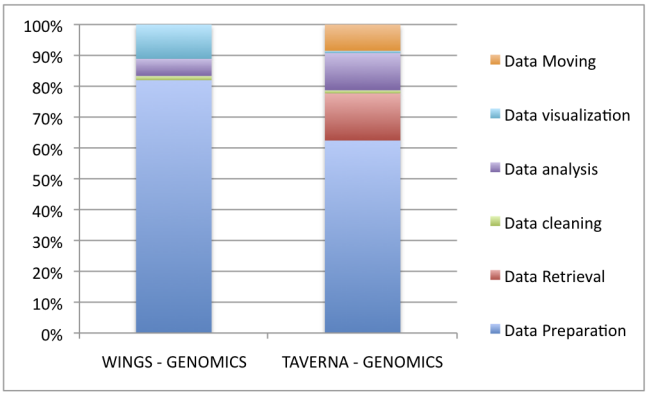

Fig. 6. Data-Oriented Motifs in the Genomics Workflows

databases and shipping data to necessary locations for analysis.

The impact of the environmental difference of Wings and Taverna on the workflows is also observed in the workflow-oriented motifs (Figure 7). Stateful invocations motifs are not present in Wings workflows, as all steps are handled by a dedicated workflow scheduling framework and the details are hidden from the workflow developers. In Taverna, the workflow developer is responsible for catering for various different invocation requirements of 3rd party services, which may include stateful invocations requiring execution of multiple consecutive steps in order to undertake a single function.

Regarding workflow-oriented motifs, Figure 8 shows that Human-interaction steps are increasingly used in scientific workflows, especially in the Biodiversity and Cheminformatics domains. Human interactions in Taverna workflows are handled either through external tools (e.g., Google Refine), facilitated via a human-interaction plug-in, or through simple local scripts (e.g., selection of configuration values from multi-choice lists). We have observed that non-trivial human interactions involving external tooling require a large number of workflow steps dedicated to deploying or configuring the external tools, resulting in very large and complex workflows. Wings workflows do not support human interaction steps.

Finally, the large proportion of the combination of Composite Workflows and Atomic Workflows motif in Figure 8 shows that the use of sub-workflows is an established best practice for modularizing functionality.

VI. Discussion

Our analysis shows that the nature of the environment in which a workflow system operates can bring-about obstacles against the re-usability of workflows.

A. *Obfuscation of Scientific Workflows*

Data-intensive scientific analysis could be large and complex with several processing steps corresponding to different phases of data analysis performed over various kinds of data. This complexity is exacerbated when the workflow operates in an open environment, like Taverna's, and composes multiple third party services supporting different data formats and protocols. In such cases the workflow contains additional steps for coping with different format and protocol requirements. This obfuscation of the workflow burdens the documentation function and creates difficulty for the workflow re-user scientists, who seeks to have a complete understanding of the function and the details of the workflow that they are re-using in order to be able make scientific claims with their workflow based studies.

Obfuscation is caused by the abundance of data preparation steps, data movement operations and multi-step stateful invocations. One way to overcome obfuscation is to encapsulate

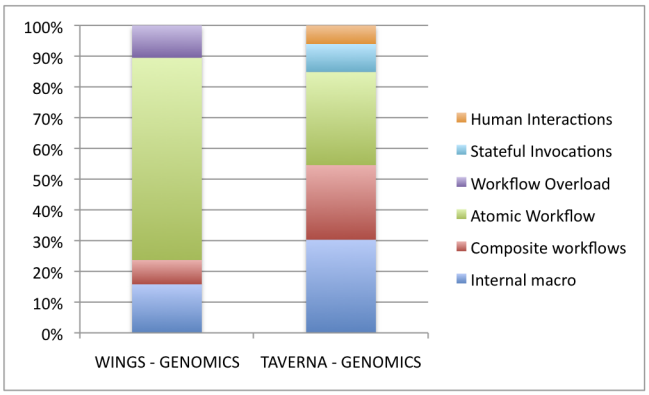

Fig. 7. Workflow-Oriented Motifs in the Genomics Workflows

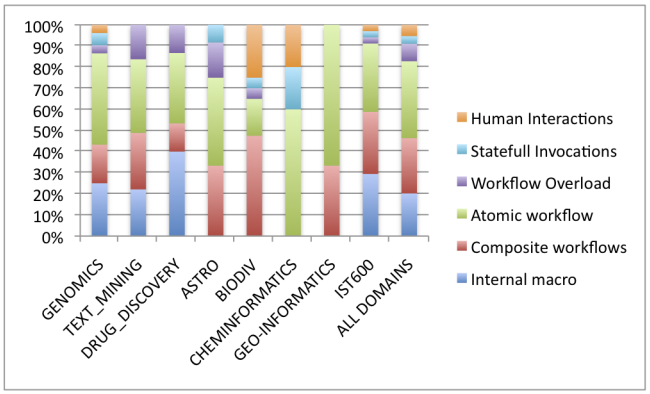

Fig. 8. Distribution of Workflow-Oriented motifs in the analyzed workflows per domain

the preparation steps and multi-step service invocations within sub-workflows. While this approach is plausible, it would quickly lead to very significant data retrieval, analysis or visualization step to turn into a workflow together with its shims, resulting wide-spread and ad-hoc sub-workflowing. A principled solution is being developed in the Taverna system through "Components". Components can be seen as sub-workflows encapsulating scientifically significant functionality and hiding technical detail. Components are described through a common metadata framework and provide support of complex structured data objects as inputs and outputs.

B. Decay of Scientific Workflows

Decay is another obstacle for the reproducibility of *in-silico* experiments and effective re-use of workflows. Widespread adoption of workflows has demonstrated that workflows decay due to the change or disappearance of remote resources that they depend on. Development of workflows is a labor intensive process, therefore workflows, even in their broken state, represent value worth attention towards repair. Table III provides an analysis of workflow evolution data from the myExperiment repository, depicting the different causes of change when authors revise a workflow from one version to the next.

When we look at the distribution of changes we see that an

TABLE III
DISTRIBUTION OF GOALS OF WORKFLOW VERSION UPDATES

Category	Frequency
FIX	
GENERAL-MAINTENANCE	40 %
BUG-FIX	11 %
IMPROVEMENT	
DOCUMENTATION IMPROVEMENT	18 %
SAMPLE-TEST DATA ADDITION	2 %
FEATURE-ADDITION	
WF-SIGNATURE-EXPANSION	12 %
USABILITY IMPROVEMENT	6.5 %
OTHER IMPROVEMENT	6.5 %
PERFORMANCE IMPROVEMENT	1 %
REFACTORING-SIMPLIFICATION	2 %

overwhelming majority are "General Maintenance" activities. These are attempts to fix workflows that are broken due to a changing environment. The second most frequent activity in workflow evolution is providing better documentation on the workflow that has been published.

Being able to repair a workflow requires understanding the workflow function, and replacing broken step or fragments with alternative or updated services. We observe that abstract workflow descriptions accompanying executable scripts, not only help in understandability but also could help in repairing as it provides a search template for alternative services, or workflow fragments.

C. Good Practices for Workflow Development

1) Sub-workflows: Encapsulation functionality in sub-workflows and reusing them in composite workflows is observed as a pattern in empirical data both in Wings and Taverna systems. As depicted in Figure 8, the Composite Workflow and Atomic Component patterns account up to more than 60% of workflow-oriented motifs observed on the workflows.

2) Workflow Overloading: An advanced form of sub-workflow development is workflow overloading, which is more common in Taverna's workflows. As workflows can be executed in different settings, authors provide overloaded versions of the same functionality in different workflows to increase the coverage on target uses. In Wings this is normally not necessary, as the components can be configured to run different configurations depending on the semantic type of the input. While we observe overloading as a good practice, a significant behavior of workflow developers in the Taverna environment is to extend their workflows with the ability to accept input from multiple ports in different formats (see Table III). We believe that such overloading behavior within a single workflow is a poor practice and should be provided over multiple workflows operating a single designated input format.

D. Towards Automated Abstraction/Annotation of workflows

Our ultimate objective is to automatically suggest a set of annotations/abstractions for workflows. These annotations would allow 1) helping the creators to describe the particular functionality of the workflows to reach a broader audience of possible re-users and 2) help in the search of workflows

with certain functionality (e.g., workflows with data retrieval, analysis and filtering). This would also be helpful from a workflow designer perspective, so as to obtain workflows that are similar to the ones being designed.

In an environment like Wings, where semantic typing is supported, it could be possible to automatically detect some Data Preparation activities by inferencing over the types of inputs and the outputs and the task types. In an open environment like Taverna, such classifications are not available, but there are other resources for inferring functionality of steps, like controlled tags on services and the names of processors. We have observed that there is a rather consistent vocabulary used to name the Data Retrieval, Data Preparation and Data Visualization steps in workflows.

Our identification of workflow oriented motifs also acts as a set of heuristics for creating abstractions over workflows, like grouping stateful interactions on a service endpoint, detection of data preparation activities to highlight the real functionality of the workflow, detecting subgroups of repeated data preparations steps (i.e., internal macros), etc.

VII. Conclusions

Workflow understanding is an impediment to re-using and re-purposing scientific workflows. To address this problem, motifs that provide high level description of the tasks carried out by the workflow steps can be effective. As a step towards this goal, we reported in this paper on an empirical analysis that we conducted using Taverna and Wings workflows, with the objective of identifying the motifs that are embedded within those workflows. In doing so, we have distinguished data-oriented motifs, which describe the tasks carried out by the workflow steps, from workflow-oriented motifs, which describe the way those tasks are implemented within the workflow.

The main findings of our analysis are as follows: there are 6 main types of data-oriented motifs and 6 types of workflow-oriented motifs that are used across different domains. The frequency in which these motifs appear depends largely on the differences among the Taverna and Wings workflow environments and differences in domains. Regarding data preparation motifs, we found that their use is correlated with the environment in which the workflow is designed. In particular, in a workflow system such as Taverna many steps in the workflow are dedicated to the moving and reconciliation of heterogeneous data sets and stateful protocol handling. On the other hand, in a workflow system such as Wings we notice that data preparation motifs, such as data moving and data reconciliation, are minimal and in certain domains absent.

As part of our future work, we plan to analyze common motifs in other workflow systems like Kepler [7] and Swift [2], and examine how they overlap with the catalogue identified in this paper. We also envisage providing tools that assist designers and users in workflow annotation using the motifs we identified. Finally, we intend to derive best practices that can be used in workflow design.

Acknowledgments

This research was supported in part by a grant from the US Air Force Office of Scientific Research (AFOSR) through award number FA9550-11-1-0104 and the Wf4Ever European project (FP7-270192). The authors would like to thank many collaborators for contributing the workflows analyzed for this work, in particular Yan Liu, Matheus Hauder, Chris Mason, and Varun Ratnakar. We also would like to thank Pinar Senkul for her comments on possible ways of inferring motifs over existing workflow scripts.

References

[1] E. Deelman, G. Singh, M. Su, J. Blythe, Y. Gil, C. Kesselman, J. Kim, G. Mehta, K. Vahi, G. B. Berriman, J. Good, A. Laity, J. C. Jacob, and D. S. Katz. Pegasus: A framework for mapping complex scientific workflows onto distributed systems. *Scientific Programming*, 13(3), 2005.

[2] I Foster, J Vockler, M Wilde, and Yong Zhao. Chimera: a virtual data system for representing, querying, and automating data derivation, 2002.

[3] Yolanda Gil, Varun Ratnakar, Jihie Kim, Pedro A. González-Calero, Paul T. Groth, Joshua Moody, and Ewa Deelman. Wings: Intelligent workflow-based design of computational experiments. *IEEE Intelligent Systems*, 26(1):62–72, 2011.

[4] Antoon Goderis, Ulrike Sattler, Phillip W. Lord, and Carole A. Goble. Seven bottlenecks to workflow reuse and repurposing. In *International Semantic Web Conference*, pages 323–337. Springer, 2005.

[5] Jose Manuel Gómez-Pérez, Michael Erdmann, Mark Greaves, Oscar Corcho, and Richard Benjamins. A framework and computer system for knowledge-level acquisition, representation, and reasoning with process knowledge. *International Journal of Human-Computer Studies*, 68(10), October 2010.

[6] Duncan Hull, Robert Stevens, Phillip Lord, Christopher Wroe, and Carole Goble. Treating shimantic web syndrome with ontologies. In *AKT Workshop on Semantic Web Services*, 2004.

[7] Bertram Ludäscher, Ilkay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A. Lee, Jing Tao, and Yang Zhao. Scientific workflow management and the kepler system. *Concurrency and Computation: Practice and Experience*, 18(10):1039–1065, 2006.

[8] Phillip Mates, Emanuele Santos, Juliana Freire, and Cláudio T. Silva. Crowdlabs: Social analysis and visualization for the sciences. In *23rd International Conference on Scientific and Statistical Database Management (SSDBM)*, pages 555–564. Springer, 2011.

[9] Sara Migliorini, Mauro Gambini, Marcello La Rosa, and Arthur H.M ter Hofstede. Pattern-based evaluation of scientific workflow management systems. Technical report, Queensland University of Technology, 2011.

[10] Paolo Missier, Stian Soiland-Reyes, Stuart Owen, Wei Tan, Alex Nenadic, Ian Dunlop, Alan Williams, Tom Oinn, and Carole Goble. Taverna, reloaded. In M Gertz, T Hey, and B Ludaescher, editors, *Procs. SSDBM 2010*, Heidelberg, Germany, 2010.

[11] Asunción Gómez Pérez and Richard Benjamins. Applications of ontologies and problem-solving methods. *AI Magazine*, 20(1), 1999.

[12] Arun Ramakrishnan, Gurmeet Singh, Henan Zhao, et al. Scheduling data-intensive workflows onto storage-constrained distributed resources. In *CCGRID*, pages 401–409. IEEE Computer Society, 2007.

[13] Lavanya Ramakrishnan and Beth Plale. A multi-dimensional classification model for scientific workflow characteristics. In *WANDS*, 2010.

[14] David De Roure, Carole A. Goble, and Robert Stevens. The design and realisation of the myExperiment virtual research environment for social sharing of workflows. *Future Generation Comp. Syst.*, 25(5):561–567, 2009.

[15] Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, Bartek Kiepuszewski, and Alistair P. Barros. Workflow patterns. *Distributed and Parallel Databases*, 14(1):5–51, 2003.

[16] Ingo Wassink, Paul E Van Der Vet, Katy Wolstencroft, Pieter B T Neerincx, Marco Roos, Han Rauwerda, and Timo M Breit. Analysing scientific workflows: Why workflows not only connect web services. *2009 Congress on Services I*, 2009(5):314–321.

[17] Jia Zhang, Wei Tan, John Alexander, Ian T. Foster, and Ravi K. Madduri. Recommend-as-you-go: A novel approach supporting services-oriented scientific workflow reuse. In *IEEE SCC*, pages 48–55. IEEE, 2011.