Composition Assistance for Multiple Existing Scientific Workflow Systems
Russell P. McIver, Andrew C. Jones, Richard J. White
{r.p.mciver | andrew.c.jones | r.j.white}@cs.cardiff.ac.uk

Introduction
Scientific Workflow Systems (SWS) can help researchers assemble complex tasks using data resources and analytic tools in combination. Current SWSs, however, require detailed knowledge of the components that might need to be assembled, and in particular can be difficult to use by scientists not accustomed to programming. In this poster we present research currently under way in order to develop tools to assist users to compose workflows.

Typical Workflow
The above is a workflow implemented in the Kepler SWS to achieve the modelling of species distributions. This example involves feeding relevant environmental and species presence data into a series of bioclimatic modelling components, with the result of this modelling then being fed into a visualisation component.

Elements of Our Approach
- Metadata Framework
  - Knowledge of components, user preferences etc.
- Assisted Composition
  - Use knowledge to provide suggestions for progress
- Abstract Components
  - Enable user to define workflow by high level goals
- Intermediate API
  - Support above elements across multiple existing SWSs

User Interaction
Users are guided through the workflow creation process, drawing on knowledge held about tasks and components (see later in poster). E.g. if the user selects an abstract component from those available, the system provides the user with options to specialise it into something concrete:

Intermediate API
We aim to make our approach as generally applicable as possible, across a number of distinct SWSs such as Kepler, Triana and Taverna. We have developed an API which insulates our software from the differences between these SWSs. It translates requests from our “assisting interface” into relevant calls to the chosen underlying system:

System Knowledge
The system stores knowledge in an ontology about component types, available components and their metadata; users, their preferences and interests and typical behaviour (dynamically updated); and domains for which workflows are being created. Here we illustrate knowledge about component types and available components.

Abstract Components
Components are maintained in a hierarchy. At the bottom of the hierarchy are concrete components which specialise and implement the abstract components:

Metadata Framework
We use metadata for two distinct purposes: assessing the compatibility of components (whether inputs and outputs can be connected) and the desirability of connecting them (whether it is appropriate -- based not only on declared data types but also on specialist knowledge of constraints on the data that is appropriate):

Acknowledgement
This research has been carried out while the first author was in receipt of a PhD studentship from Microsoft Research Europe.