GB-500: Introdução a Workflows Científicos e suas Aplicações

Professores: Luiz Gadelha, Kary Ocaña

Programa de Verão do LNCC, 2017
Laboratório Nacional de Computação Científica

29 de março de 2018

Workflows Científicos

- Um **workflow científico** consiste da especificação de um encadeamento de aplicações científicas a serem executadas e de dependências mútuas.
- Segue um ciclo de vida análogo ao dos experimentos científicos computacionais:
  - Composição, representação e modelagem de dados.
  - Mapeamento e execução.
  - Coleta de metadados e proveniência.
- Um **sistema de gerência de workflows científicos** (SGWC) permite gerenciar o ciclo de vida de workflows científicos.

Liu, J., Pacitti, E., Valduriez, P., Mattoso, M. (2015). A Survey of Data-Intensive Scientific Workflow Management. Journal of Grid Computing, 13(4), 457–493.

Workflows Científicos

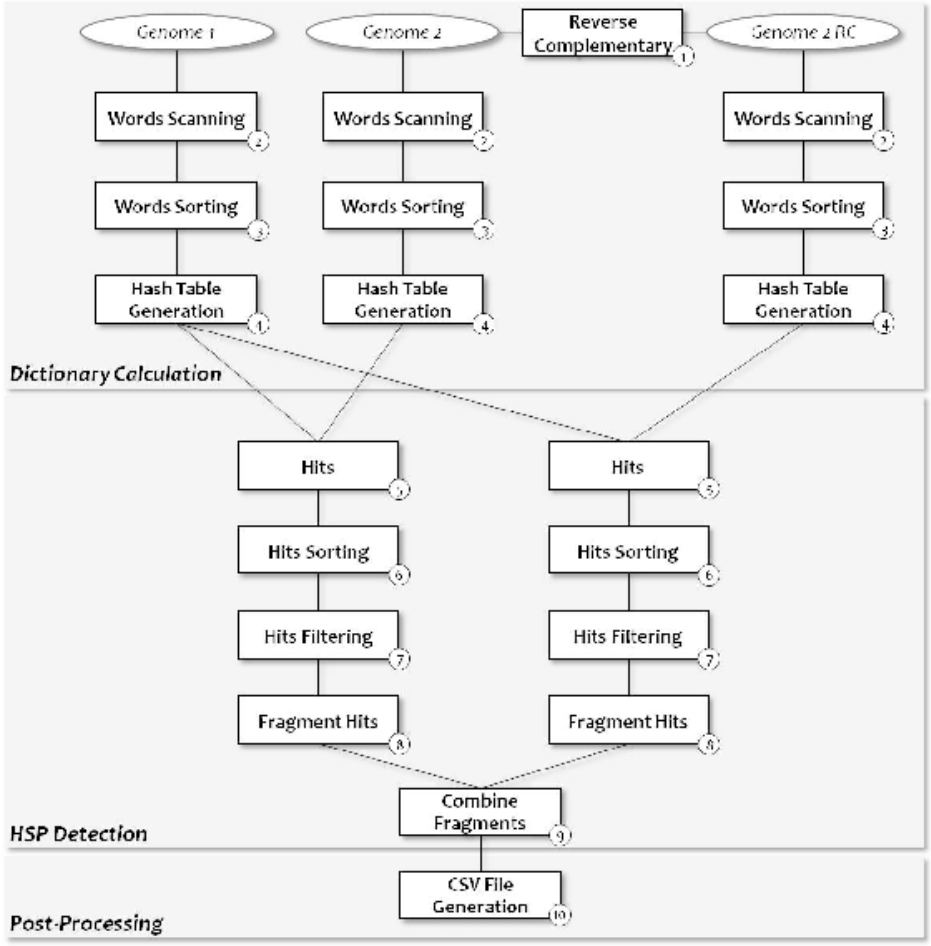
- **Sistemas de gerência de workflows científicos** (SGWC) visam a automação de experimentos científicos computacionais:
  - escalonamento de tarefas baseado em dependências de dados;
  - fluxo de dados entre tarefas;
  - execução paralela de tarefas independentes;
  - escalonamento de tarefas em ambientes de computação de alto desempenho;
  - gerência e consulta de dados de proveniência.

Exemplo: Laboratório de Bioinformática, LNCC

Sequenciador Roche 454 FLX gera 12GB a 15GB por rodada de sequenciamento.
Dados processados por várias aplicações:

- filtragem por qualidade de dados gerados pelo sequenciador.
- formatação de dados.
- comparação das sequências com bases de dados conhecidas.

Exemplo: SwiftGECKO (genômica comparativa)



Mondelli, M. L., ..., Gadelha, L. M. R. (2018). BioWorkbench: A High-Performance Framework for Managing and Analyzing Bioinformatics Experiments. arXiv:1801.03915.
`http://arxiv.org/abs/1801.03915`

Laboratório Nacional de Computação Científica

Desafios de Pesquisa em Workflows Científicos

- Acoplamento de tarefas e transferência de dados entre tarefas de um workflow.
- Modelos de programação, interface com o usuário, comunicação entre tarefas e portabilidade.
- Monitoramento: andamento de execução, algoritmos para detecção de anomalias.
- Validação de execução de workflow:
  - reproduzir um workflow no mesmo ou em outro ambiente computacional,
  - comparar a execução com modelos de desempenho e com a proveniência coletada durante a execução,
  - comparar os resultados científicos com o que era esperado.

Deelman, E. et al. (2018). The future of scientific workflows. The International Journal of High Performance Computing Applications, 32(1), 159–175. https://doi.org/10.1177/1094342017704893

Workflows Científicos

- Workflows científicos típicos:
  - *Bag of tasks* (MG-RAST, DOCK),
  - Múltiplos estágios que usam arquivos como dados intermediários (Montage, BLAST),
  - Aplicações distribuídas com pares chave-valor intermediários (histogramas de dados para física de altas energias),
  - Cadeias de tarefas do tipo MapReduce (mineração de grafos),
  - Aplicações iterativas com variação no número de tarefas (otimização, filtros de Kalman),

Deelman, E. et al. (2018). The future of scientific workflows. The International Journal of High Performance Computing Applications, 32(1), 159–175. https://doi.org/10.1177/1094342017704893

Workflows Científicos

- Workflows científicos podem ser executados *in situ* ou de forma distribuída.
- No caso *in situ*:
  - Processamento de dados, triagem, filtragem, análise ou visualização podem ocorrer enquanto o workflow está executando.
  - Essas ações ocorrem antes de se mover dados para fora de um supercomputador para análises adicionais
- Nenhum sistema de gerenciamento de workflows científicos atende tanto ao cenário *in situ* quanto ao distribuído.

Deelman, E. et al. (2018). The future of scientific workflows. The International Journal of High Performance Computing Applications, 32(1), 159–175. https://doi.org/10.1177/1094342017704893

Workflows Científicos

- Motivações para workflows *in situ*:
  - minimização da movimentação de dados através da exploração da localidade de dados, processando os dados *in place*,
  - apoiar análises interativas (*human in the loop*),
  - captura de proveniência para suportar a análise interativa e permitir execução adaptativa do experimento (*user steering*).

Deelman, E. et al. (2018). The future of scientific workflows. The International Journal of High Performance Computing Applications, 32(1), 159–175. https://doi.org/10.1177/1094342017704893

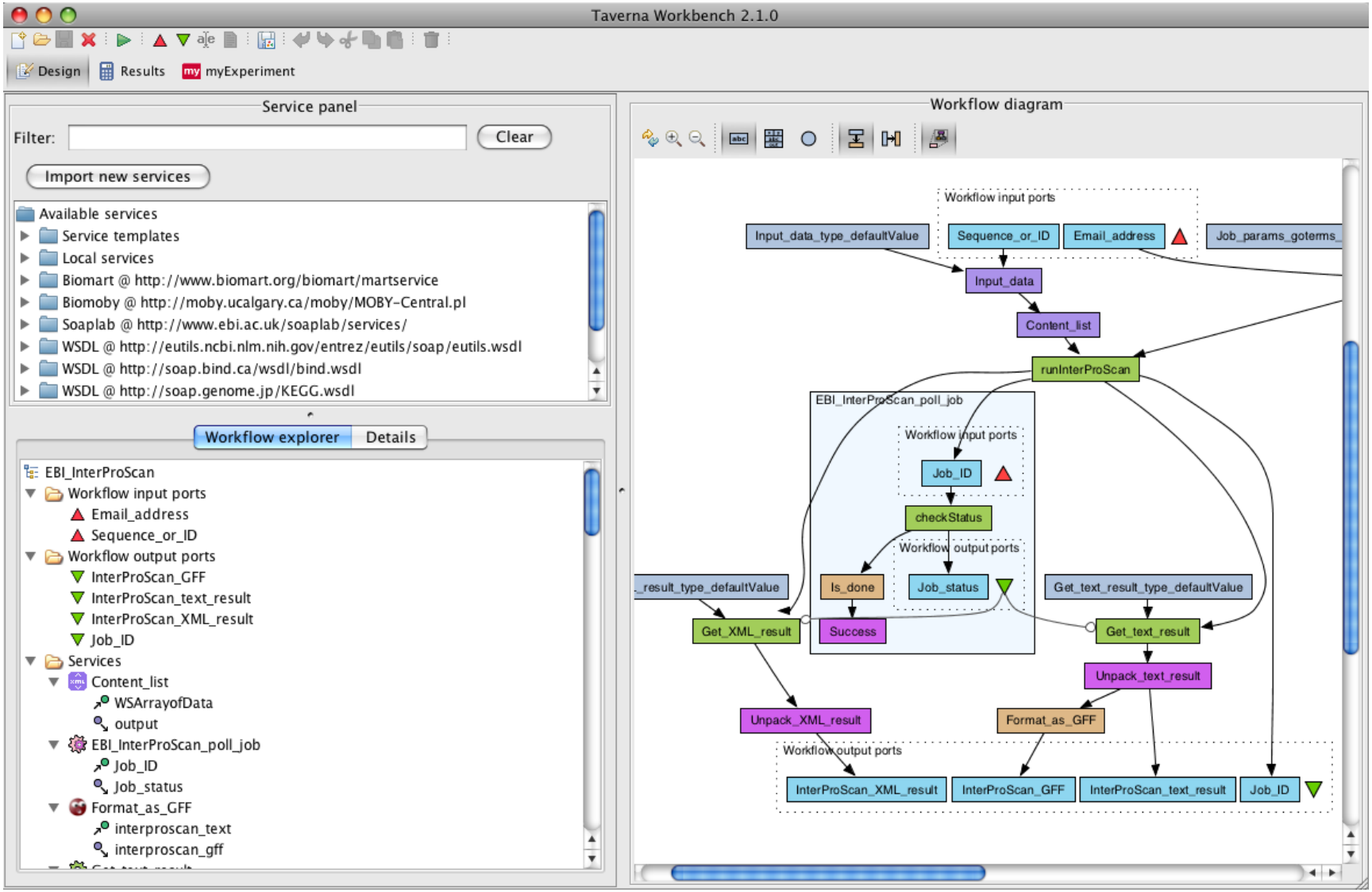
Composição de Workflows Científicos

- Etapa em que são definidas as aplicações componentes e as dependências de dados.
- Níveis de especificação:
  - Abstrato: tarefas abstratas, descritas por funcionalidade geral (p. ex., comparação de seqüências).
  - Concreto: aplicações científicas e conjuntos de dados específicos.
- Representação:
  - Textual: linguagem de programação.
  - Gráfica: interface gráfica onde nós são tarefas e arestas são dependências.

Exemplo: Composição Textual

```
type fastaseq;
type headerfile;
type indexfile;
type seqfile;
type database
{
  headerfile phr;
  indexfile pin;
  seqfile psq;
}
type query;
type output;
string num_partitions=@arg("n", "8");
string program_name=@arg("p", "blastp");
fastaseq dbin <single_file_mapper;file=@arg("d", "database")>;
query query_file <single_file_mapper;file=@arg("i", "sequence.seq")>;
string expectation_value=@arg("e", "0.1");
output blast_output_file <single_file_mapper;file=@arg("o",
                                                       "output.html")>;
string filter_query_sequence=@arg("F", "F");
fastaseq partition[] <ext;exec="splitmapper.sh",n=num_partitions>;

app (fastaseq out[]) split_database (fastaseq d, string n)
{
  fastasplitn @filename(d) n;
}

app (database out) formatdb (fastaseq i)
{
  formatdb "-i" @filename(i);
}

app (output o) blastapp(query i, fastaseq d, string p, string e, string f,
                        database db)
{
  blastall "-p" p "-i" @filename(i) "-d" @filename(d) "-o" @filename(o)
           "-e" e "-T" "-F" f;
}

app (output o) blastmerge(output o_frags[])
{
  blastmerge @filename(o) @filenames(o_frags);
}

partition=split_database(dbin, num_partitions);

database formatdbout[] <ext; exec="formatdbmapper.sh",n=num_partitions>;
output out[] <ext; exec="outputmapper.sh",n=num_partitions>;

foreach part,i in partition {
  formatdbout[i] = formatdb(part);
  out[i]=blastapp(query_file, part, program_name, expectation_value,
                  filter_query_sequence, formatdbout[i]);
}

blast_output_file=blastmerge(out);
```

Exemplo: Composição Gráfica



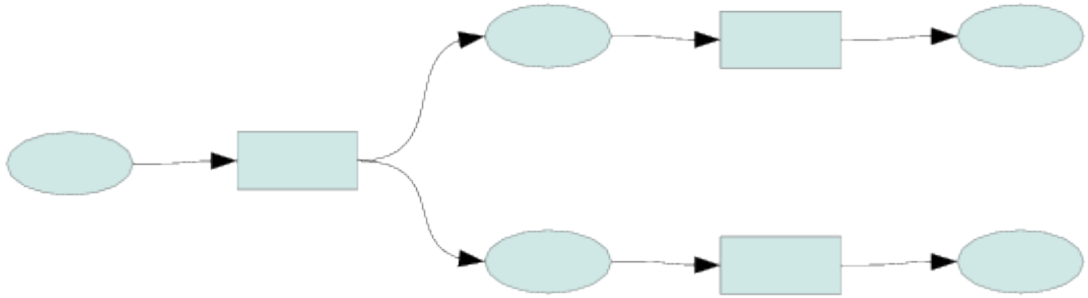
Fonte: Taverna (`http://www.taverna.org.uk`)

Laboratório Nacional de Computação Científica

Composição de Workflows Científicos: Padrões

- Foram identificados 43 padrões de composição de workflows.
- Exemplos:
  - Seqüência:

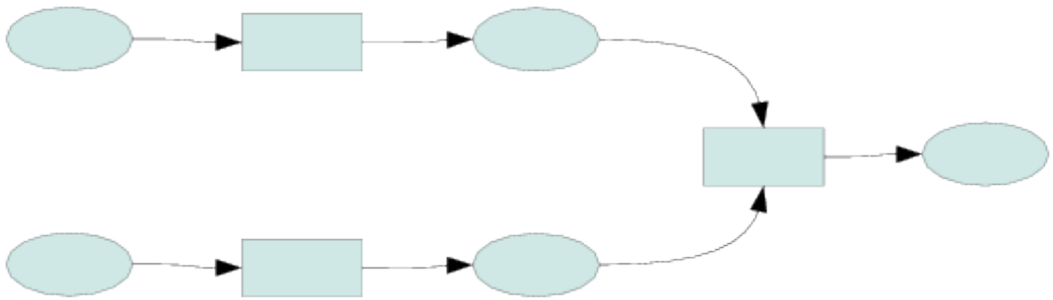
  - Bifurcação paralela:



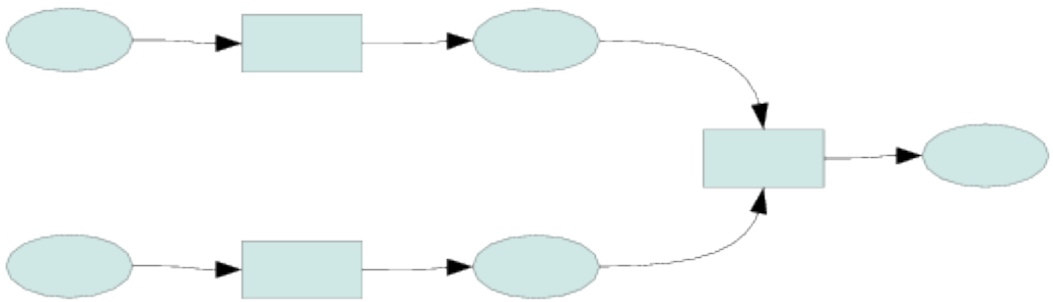
- W. van der Aalst et al. Workflow Patterns. *Distributed and Parallel Databases* 14(1):5-51, 2003.
- N. Russell et al. Workflow Control-Flow Patterns: A Revised View. BPM Center Report BPM-06-22, 2006.
- Workflow Patterns (http://www.workflowpatterns.com).

Composição de Workflows Científicos: Padrões

- Exemplos:
  - Sincronização:

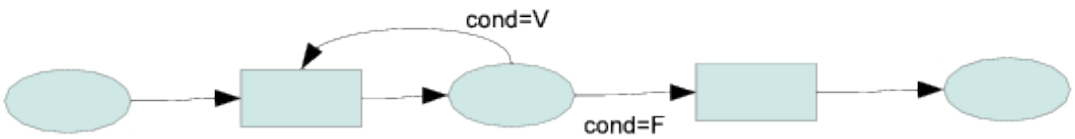


  - Fusão:

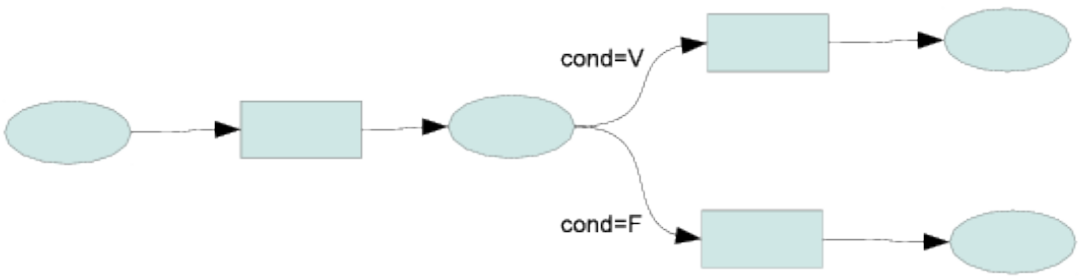


Composição de Workflows Científicos: Padrões

- Exemplos:
  - Laço:



  - Escolha exclusiva:



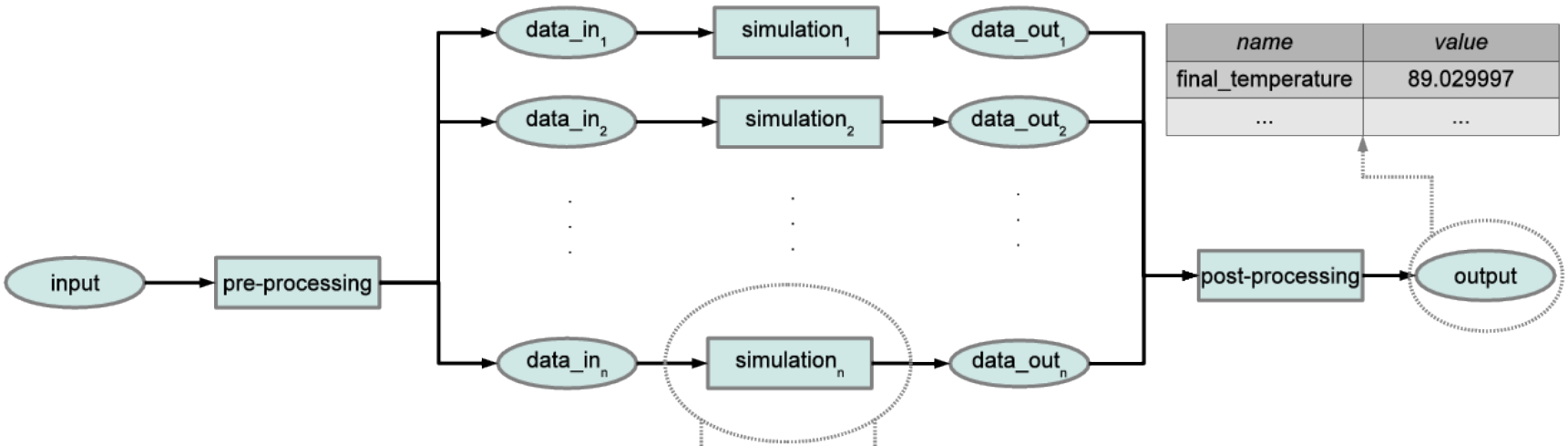
Mapeamento e Execução de Workflows Científicos

- Etapa em que são definidas as aplicações componentes concretas e os locais de execução das mesmas.
- Escalonamento da execução das aplicações componentes.
  - Interno.
  - Externo.
- Modelo de execução.
  - Execução local/*in situ*.
  - Execução remota/distribuída.
- Tolerância a falhas.
- Redundância.
- Execução adaptativa.

Coleta de metadados e proveniência

- Coleta de eventos do ciclo de vida do workflow:
  - Mudanças e evolução da especificação.
  - Consumo e produção de dados por aplicações componentes executadas.
- Gerência de metadados relacionados ao domínio científico (semântica do experimento).
- Serviço de consulta a metadados e proveniência.
- Aplicações: reprodutibilidade, verificação, análise.

Coleta de metadados e proveniência



name	*value*
final_temperature	89.029997
...	...

name	*value*
execution_site	pads.ci.uchicago.edu
version	1.15
...	...

timestamp	*mem_usage*	*cpu_load*	*io_read*	*io_write*
1314104905	126342	99	230	0
1314104906	155739	100	123	1044
...	...	...	...	...
1314112105	394523	99	0	129

Exemplos de SGWCs: SciCumulus

- Permite gerenciar workflows científicos em ambientes de nuvem.
- É composto por:
  - Especificação de workflows em XML.
  - Motor de execução baseado em álgebra relacional.
  - Sistema de gerência de proveniência.

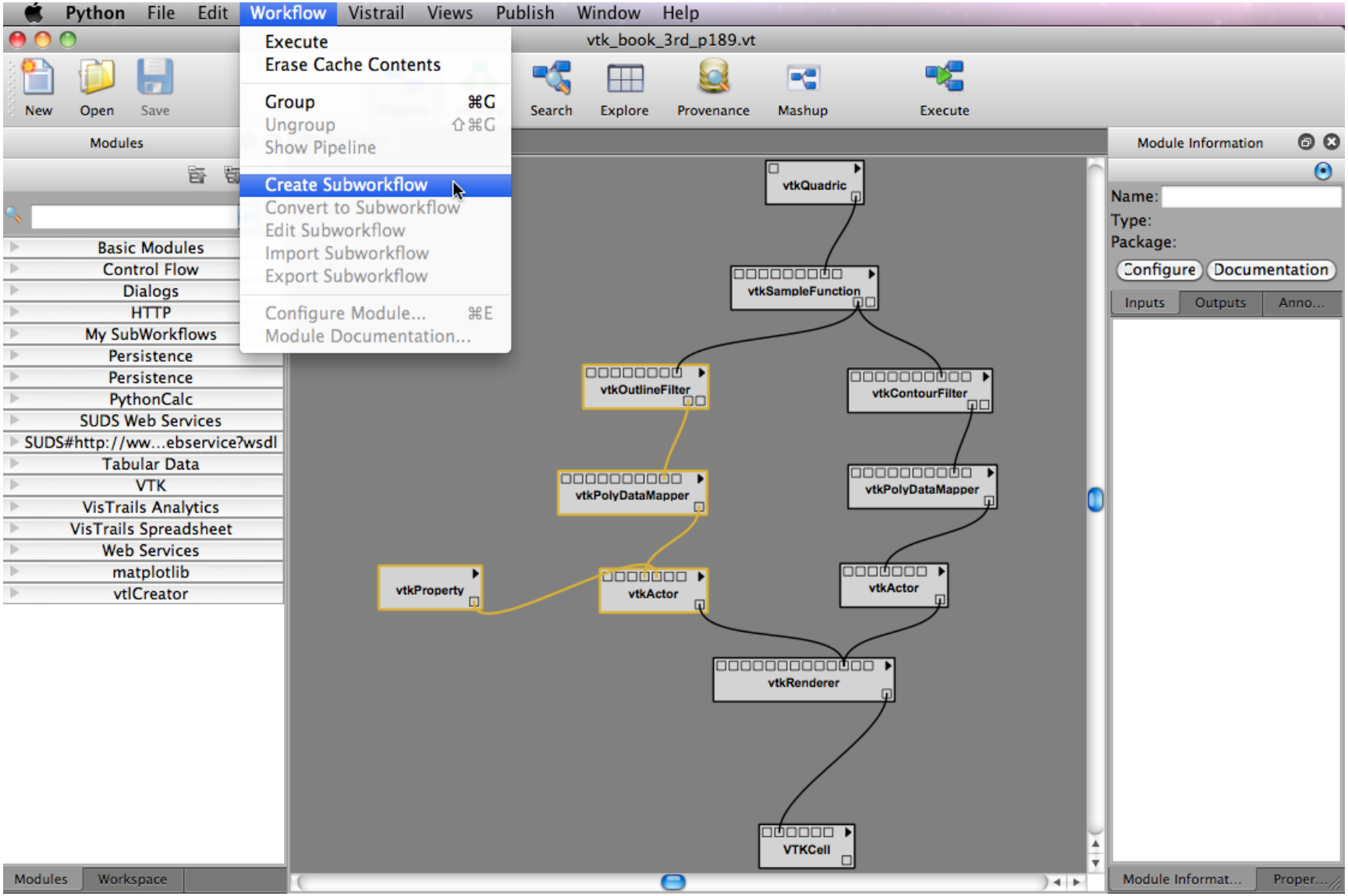
SciCumulus (`http://sourceforge.net/projects/scicumulus/`)

Exemplos de SGWCs: Vistrails

- Suporte a versionamento e gerência da evolução de workflows.
- Especificação de workflows através de interface gráfica.
- Suporte a coleta e consultas de proveniência.
- Popular na área de visualização científica.

Vistrails (`http://www.vistrails.org`)

Exemplos de SGWCs: Vistrails



Fonte: Vistrails (`http://www.vistrails.org`)

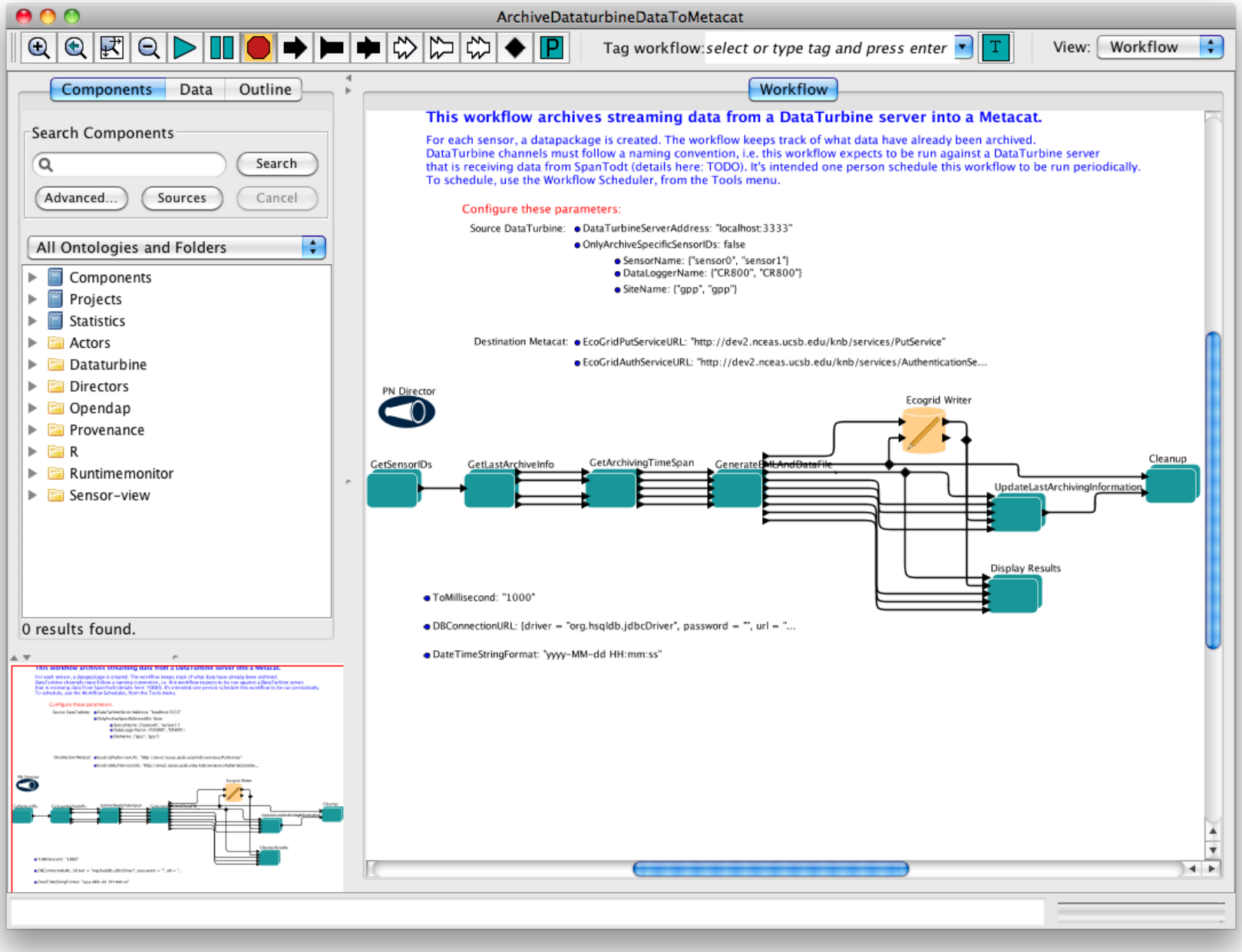
Laboratório Nacional de Computação Científica

Exemplos de SGWCs: Kepler

- Suporte a aplicações disponibilizadas através de serviços web.
- Especificação de workflows através de interface gráfica.
- Suporte a coleta e consultas de proveniência.
- Integração com ferramentas populares, como o R.

Kepler (`https://kepler-project.org`)

Exemplos de SGWCs: Kepler



Obrigado!

E-mail: lgadelha@lncc.br