Intentional Process Modeling to Share and Reuse In-Silico Experiments

Nadia Cerezo, Pierre Crescenzo, Isabelle Mirbel

LABORATOIRE

INFORMATIQUE, SIGNAUX ET SYSTÈMES
DE SOPHIA ANTIPOLIS
UMR 6070

INTENTIONAL PROCESS MODELING TO SHARE AND REUSE IN-SILICO EXPERIMENTS

Nadia Cerezo, Pierre Crescenzo, Isabelle Mirbel

Equipe MODALIS

Rapport de recherche
ISRN I3S/RR–2011-07–FR

Mai 2011

Laboratoire d'Informatique de Signaux et Systèmes de Sophia Antipolis - UNSA-CNRS
2000, rte.des Lucioles – Les Algorithmes – Bât Euclide B – B.P. 121 – 06903 Sophia-Antipolis Cedex – France
Tél.: 33 (0)4 92 94 27 01 – Fax: 33 (0)4 92 94 28 98 – www.i3s.unice.fr
UMR6070

RÉSUMÉ :

MOTS CLÉS :

ABSTRACT:

In-silico experiments, also known as simulations, are traditionally automated through scientific workflows. While those systems effectively tackle the complexity of underlying technologies, they themselves do little to ease and organize sharing and reuse. Indeed, scientific workflow models blur the line between user goals and techniques and often mix abstraction levels. We believe that an intentional model, explicitly differentiating goals from methods to achieve them, is a necessary step towards the creation of an effective sharing and reuse platform for in-silico experiments. Our contribution is threefold: (1) we extended the map model with explicit forks and parameterizable maps, (2) we proposed a map ontology in RDFS that allows computer processing and reasoning on maps and (3) we applied the map model to in-silico experiments in order to emphasize user intentions.

KEY WORDS :

intentional process modeling, scientific workflow, semantic web, reuse

Intentional Process Modeling to Share and Reuse In-silico Experiments

Nadia Cerezo, Pierre Crescenzo, and Isabelle Mirbel

I3S Laboratory, University of Nice Sophia-Antipolis - CNRS, France
{nadia.cerezo,pierre.crescenzo,isabelle.mirbel}@unice.fr

Abstract. In-silico experiments, also known as simulations, are traditionally automated through scientific workflows. While those systems effectively tackle the complexity of underlying technologies, they themselves do little to ease and organize sharing and reuse. Indeed, scientific workflow models blur the line between user goals and techniques and often mix abstraction levels. We believe that an intentional model, explicitly differentiating goals from methods to achieve them, is a necessary step towards the creation of an effective sharing and reuse platform for in-silico experiments. Our contribution is threefold: (1) we extended the map model with explicit forks and parameterizable maps, (2) we proposed a map ontology in RDFS that allows computer processing and reasoning on maps and (3) we applied the map model to in-silico experiments in order to emphasize user intentions.

Keywords: intentional process modeling, scientific workflow, semantic web, reuse

1 Introduction

In-silico experiments, commonly referred to as simulations, are experiments partially or entirely carried out via computers. In many fields, these experiments have become a major component of scientific research, allowing scientists to simulate catastrophic events in seismology and disaster handling, to tackle gigantic amounts of data in astrophysics and genetics, to predict the effects of a therapy before trying it on live subjects in medicine and so forth. Many factors contribute to the necessity of automating such experiments, most notably the volume of data to analyze and the exploratory nature of the analysis, which leads to frequent reuse and repurposing. For many years, scientists have chained the various programs that composed their simulations through scripting and often manual data conversion and transfer. Obviously, this approach is neither accessible nor robust, but its most serious problem is that it cannot leverage the wealth of resources provided nowadays by grid computing and service-oriented architecture. The need to automate the use of highly-distributed heterogeneous resources was answered in the corporate world by the concept of workflow, formally defined in [1] as the *computerized facilitation or automation of a business process, in whole or part.* This notion and the associated frameworks were adapted to scientific

needs - such as a much greater need for flexibility and a switch in priorities, from security and integrity to reproducibility and reusability concerns - and it opened a new research field: scientific workflows.

There are many scientific workflow frameworks - e.g. Taverna, Kepler, Triana, MOTEUR, Project Trident - and comparatively few surveys [2], [3], [4]. Those systems tackle the complexity of the underlying technologies, especially those pertaining to grid computing and distributed algorithms, and efforts are clearly made to improve accessibility through ease-of-use. Projects like myExperiment[1] aim to leverage the social web potential to ease and organize sharing and reuse of scientific workflows. We believe that while those portals surely help, they are hindered by the underlying scientific workflow models, too low-level to explicit the intentions of a workflow's author and thus burying the elements relevant to sharing, reuse and repurposing under techniques and technical considerations.

To improve and organize sharing and reuse of in-silico experiments, we need a process model meeting the following criteria:

1. **Flexibility** is made critical by the inherently exploratory nature of research: experiment protocols are constantly adapted to novel ideas, new results and fresh data.
2. **Modularity** is unavoidable, since share and reuse are otherwise near impossible.
3. **Abstraction levels** are important to cater to users of various technical proficiency levels.

There are plenty of options to model business processes [5]. Activity-based process models, such as BPMN [6], describe processes linearly as sets of predefined activities and relations between them to express data flow. Product-driven process models emphasize the results of performed activities and focus on the product's evolution throughout the process. Approaches like EPC [7] take into account the temporal and logical dimensions and highlight the control flow of the process. Decision-based process models regard product transformations as the consequences of decisions taken in a given execution context. Those models make the answer to "Why?" as explicit as the answer to "How?", and thus guide decision-making as well as reflexion on the process itself [5]. According to [5], business processes may be seen from three complementary viewpoints: (1) its objectives, (2) the organization in which it exists and (3) its technological infrastructure. Each family of business process models focuses on one or more of those aspects.

Our aim is to assist knowledge sharing between users of various technical and process modeling proficiency levels. It is therefore critical that we adopt a model that explicitly represents why and how a process is broken up from the highest to the lowest level of abstraction. For all the aforementioned reasons, we base our approach on the map model [8]: an intentional process model that sets apart goals to achieve (*Intentions*) from the ways to achieve them (*Strategies*).

[1] `http://www.myexperiment.org`

In Section 2, we briefly present maps then propose our own extensions for the model. In Section 3, we explain why and illustrate how we apply semantic web technologies to maps. Section 4 concludes this paper with research perspectives.

2 Map model extension

We propose two major extensions to the map model [8], [9]: on the one hand, we make the distinction clear and mandatory between AND, OR and XOR forks to allow automatic handling, on the other hand, we introduce the notion of parameterizable maps to ease and enhance reuse. Before we delve into those extensions, let us recall the basics of the original map model as well as justify some minor modifications we made to reflect the switch from control-driven to data-centered processes.

A *Map* is a directed graph whose vertices are *Intentions* - goals the user wants to achieve - and whose edges are *Strategies* - ways to achieve target intentions. Every *Map* has two special *Intentions* called *Start* and *Stop* and representing respectively the beginning and the end of the process. Triples made of a *Strategy* and its source and target *Intentions* are called *Sections*. A *Section* may be refined by a sub-*Map*. Through refinement, a *Map* author can separate abstraction levels or encapsulate sub-processes into larger processes while preserving the global *Map*'s understandability. Guidelines are provided in natural language to help the enactor:

- To every *Section* is associated an *IAG* (Intention Achievement Guideline) that guides the practical enactment of the *Section*.
- To every *Intention* is associated an *ISG* (Intention Selection Guideline) that directs the choice of the next target *Intention*.
- To every pair of source *Intention* and target *Intention*, i.e. linked by at least one *Strategy*, is associated a *SSG* (Strategy Selection Guideline) that directs the choice of the *Strategy* to employ to reach the target *Intention*.

Fig.1 illustrates all the aforementioned elements. Refer to [8] and [9] for more details about the map model itself.

Intentions and *Strategies* were originally expressed following a linguistic approach as such: 1 *Intention* +1 *Strategy* = 1 *Verb* +1 *Object* +n *Parameters*. This approach emphasizes the *Verb* and thus suits control-driven processes very well. In-silico experiments are most often data-centered: the *Subject* on which the *Action* is performed is generally much more important than the *Action* itself. *Parameters* previously only characterized *Strategies*, but with in-silico experiments, *Subjects* often also need to be characterized, e.g. a PNG image vs. a JPG image: the image format is a *Feature* of the image *Subject*. Therefore, we divided *Parameters* in two distinct groups: *Features* characterize *Subjects*, whereas *Attributes* characterize *Strategies*. There is also often the need to perform a given *Action* on many *Subjects* at once, for instance if we need to compare two different types of data. In practice, we express things thusly: 1 *Intention* = 1 *Action* +n(*Subject* [n *Feature*]) and 1 *Strategy* = n *Attributes*.

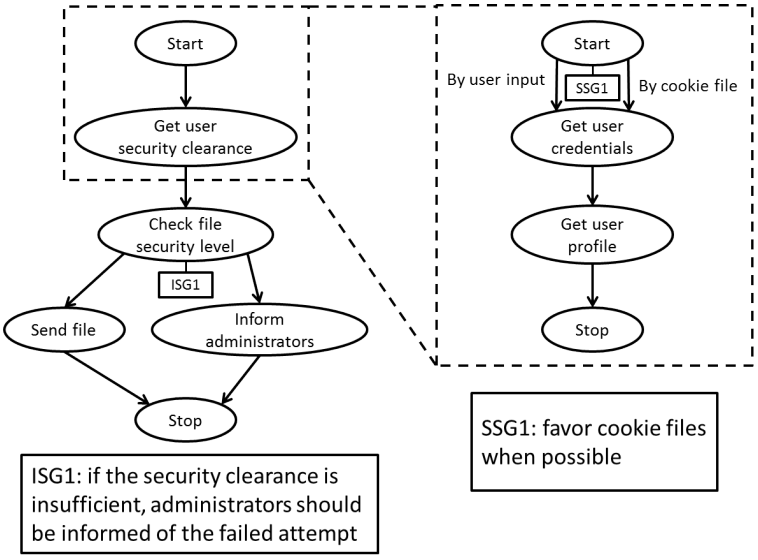

Fig. 1. Illustration of the original map model.

2.1 Explicit forks

Every time two or more *Sections* share the same source *Intention*, we call it a fork since it presents the user with a choice between multiple paths. There are three types of forks:

1. XOR forks are the mutually exclusive choices, e.g. you check the system, if something is wrong you stop everything and raise the alarm, else you proceed normally.
2. OR forks present alternatives that could be tried simultaneously, e.g. two distinct servers provide you with the service you need, you might want to use them both to compare them.
3. AND forks are related to task independence: more than one subtasks have to be done to achieve the global goal, but there is no order between the subtasks and you might as well perform them in parallel, if available resources are enough.

In maps as presented in [8], every fork is ambiguous in the sense that it is up to the user, to whom selection guidelines expressed in natural language are provided, to choose whether to reach more than one intention and/or employ more than one strategy at a given time. Similarly, it is up to the map author to specify the nature of a fork (i.e. AND, OR or XOR) in the associated guidelines. The notion of bundle was introduced in [9] and allows the representation of a specifically XOR fork. The complementary notion of thread is meant for OR forks. Obviously, the map author might specify in guidelines that a given fork is supposed to be an AND fork, but it is not mandatory in any way and is

certainly not apparent on the graphical representation. Such leniency in fork definition makes sense in the context of [8]: a human being can take advantage of such variability and dynamically adapt a given map to the situation at hand.

Problems arise when trying to process maps automatically. Obviously, with guidelines expressed in natural language that may or may not clearly state the nature of a fork, the enactor cannot determine what to do. Synchronization is a less obvious issue that cannot be solved simply with restrictive guideline specifications: if many paths can be taken simultaneously in a given map, how does the enactor track map completion? In other words, when the Stop intention is reached, is the map truly completed or should the enactor wait for possible other threads? To clear the synchronization concern and to make the distinction between the three types of forks both explicit and mandatory, we decided to confine both alternatives (OR) and independent tasks (AND) to submaps, through the refinement mechanism, thus restricting all forks in a given map to mutually exclusive choices (XOR).

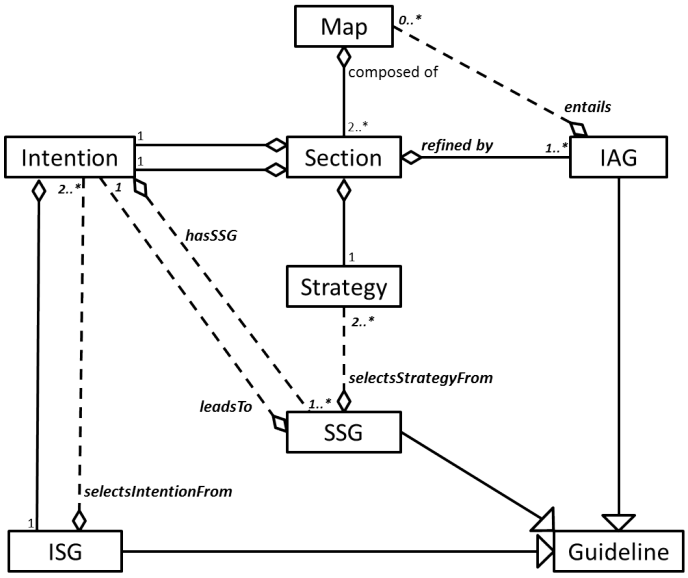

Fig. 2. Map meta-model: map-guidelines relationships. Names and cardinalities we modified are in bold and italic. Relationships we introduced are dashed.

To accommodate the uses we wanted to make of the refinement mechanism, for AND and OR forks in particular, we modified the original map meta-model (see Fig.2), of which we provide a graphical representation in Appendix (Fig.9): *Sections* are no longer *refined by* one *Map*, instead they are *refined by* any number of *IAGs*, each of which *entails* from one or more *Maps*.

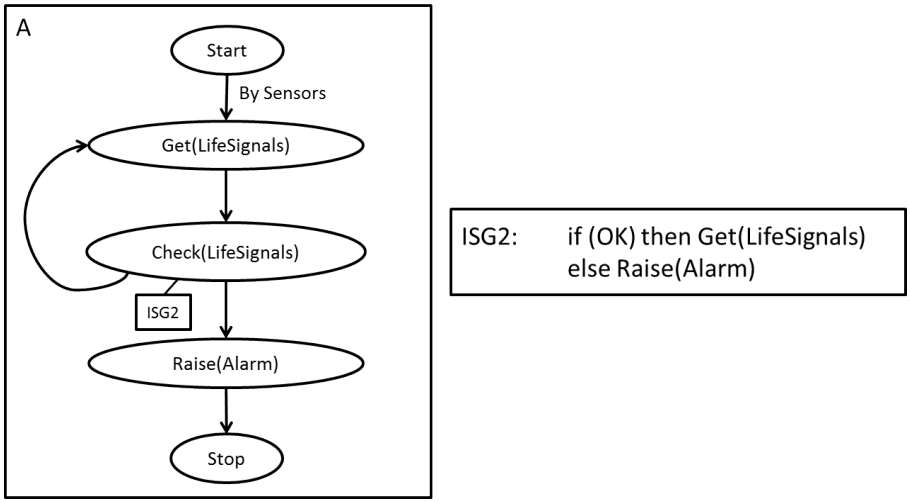

Fig. 3. A map (A) with mutually exclusive options (i.e. a XOR fork) and the corresponding ISG. It represents the process of monitoring the life signals of a patient.

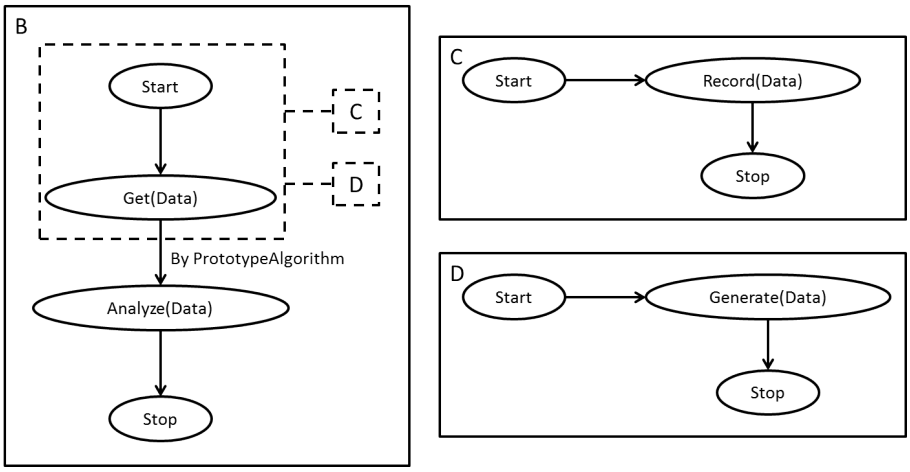

Fig. 4. A map (B) with alternative submaps (i.e. an OR fork) and the corresponding submaps (C and D). They represent the choice between synthetic and live data to feed a prototype algorithm.

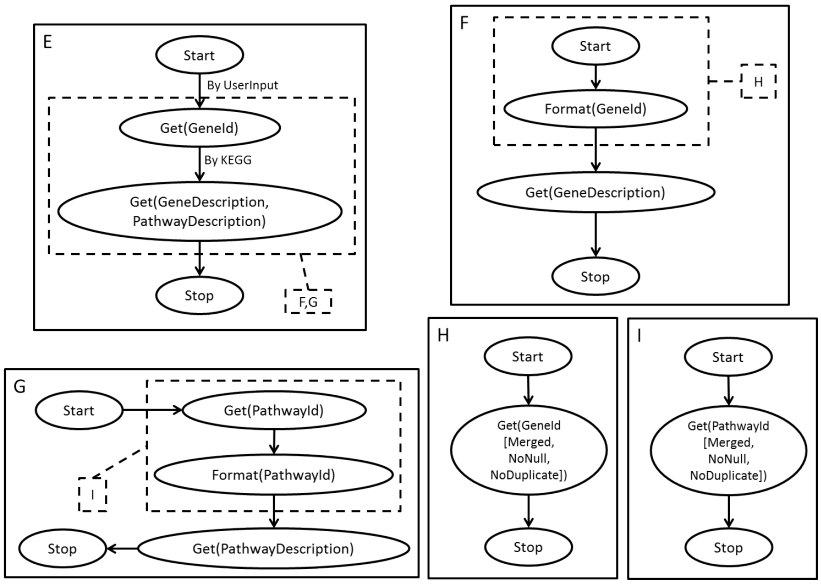

Fig. 5. A map (F) with independent submaps (i.e. an AND fork) that are themselves (G and H) refined to a lower level of abstraction (resp. I and J). This example was inspired by a Taverna scientific workflow found on myExperiment.org at the following URL: http://www.myexperiment.org/workflows/611.html.

- XOR forks are represented the way all types of forks were in the original model: by multiple *Strategies* sharing the same source *Intention*, as illustrated by Fig.3.
- OR forks are represented by multiple *IAGs refining* a given *Section*, as illustrated by Fig.4.
- AND forks are represented by multiple *Maps entailed* by a given *IAG*, as illustrated by Fig.5.

Our modifications make the map model more restrictive but no less expressive: the same scenarios can be modeled with both models. We merely transferred fork specification (and with it, part of the cognitive workload) from the enactor to the modeler, a critical step towards automatic processing.

2.2 Parameterizable maps

The other extension we propose addresses an issue that is not exclusive to maps but rather stems from the desire to share and reuse processes. Every now and then, a modeler finds a subprocess that almost suits the situation at hand or wants to use the same subprocess over and over in the same global process, but with different objectives, techniques or types of data. With a model that allows no genericity at all, modelers have to repurpose subprocesses by creating as many

different instances. On the one hand, it creates many more subprocesses than are meaningfully needed, on the other hand, the link between those subprocesses is inexistent, even though they are actually specific instances of a generic process.

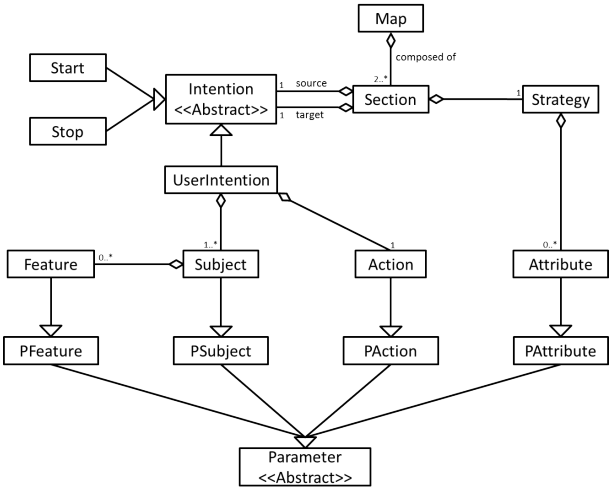

Fig. 6. Map meta-model: parameterizable maps.

We propose to alleviate those issues by enriching the map model with the notion of parameterizable maps (see Fig.6). *Parameters* may be used in four distinct places in a map: *Attributes*, *Actions*, *Subjects* and *Features*. Instantiation of a parameterized map is done through the refinement mechanism as in Fig.7. Whenever modelers feel the need to adjust a given map before including it as a submap in their approach, they may instead discover and formalize which parts of the process are actually generic. Once that modeling step is taken, repurposing becomes a simple matter of instantiation.

3 Automatic handling of maps

Our aim is to provide scientists with a platform dedicated to sharing and reuse of in-silico experiments. To actually ease and enhance sharing and reuse, we must provide our end users with features such as advanced searching, alternative suggestions and service discovery. To do so, we need to reason on maps and the processes they represent and we obviously need to transcribe our model in a computer-legible fashion. In a sense, we face the same problem that lead to the definition of the Semantic Web by Tim Berners-Lee et al. [10], albeit on a very different scale: we want to automatically integrate and combine data that are primarily designed for human use. It thus seems logical to turn to semantic web technologies such as RDF (Resource Description Framework) [11] and SPARQL (SPARQL Protocol and RDF Query Language) [12] to address our problem.

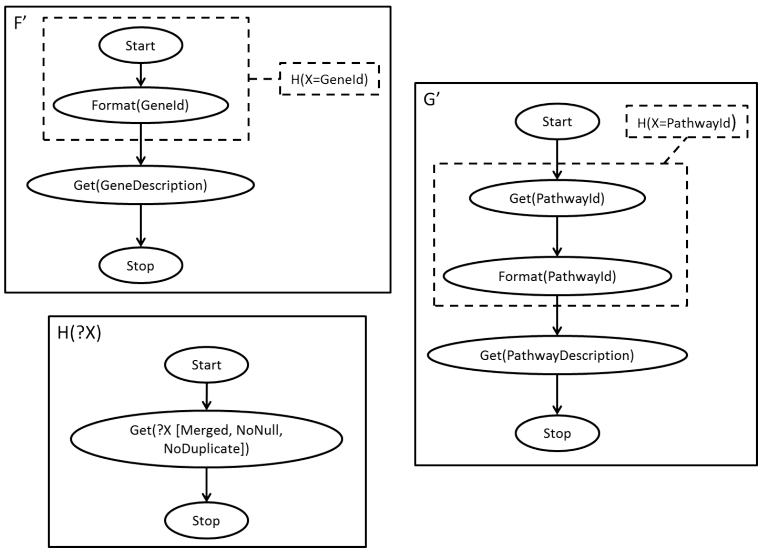

Fig. 7. A parameterized map (H) and two maps (F' and G') instantiating it with different parameters. Maps F' and G' are slightly modified version of maps F and G in Fig.5.

3.1 Intentional process ontology

To transcribe our meta-model in RDF, so as to reason on the model itself, we had to use an ontology language. We opted for RDFS (RDF Schema) [13].

Changes we made to the ontology since its last publication in [14] mostly reflect the changes we made to the meta-model, as detailed in section 2. The latest version, as of writing, is published online[2]. We provide a graphical representation in Appendix (Fig.10).

Modeling in-silico experiments implies handling concepts from three different worlds:

- process modeling (in this case map, e.g. section, intention, strategy)
- scientific domain (e.g. gene, MRI, epicenter)
- technical level (e.g. list, PNG format, upload)

Note that while the distinction between process modeling and the rest is pretty clear, the frontier between what is technical and what is not is highly subjective. We rely on existing domain ontologies. For instance, bioinformaticians might want to use the TAMBIS [15] or the GO (Gene Ontology) [16] ontologies, to model their in-silico experiments.

Let us examine how we describe various components of a map in RDF, in accordance with the following ontologies and with the corresponding namespaces:

[2] http://www.i3s.unice.fr/~cerezo/map/2010/10/20/map.rdf

- `map: <http://.../map.rdf#>` refers to the map ontology
- `bio: <http://.../bio.rdf#>` refers to an ad hoc bioinformatics ontology we created for the sake of our examples (upper part of Fig.11 in Appendix)
- `tech: <http://.../tech.rdf#>` refers to an ad hoc technical ontology we created for the sake of our examples (lower part of Fig.11 in Appendix)

Linking to domain and technical ontologies is done through instantiation, e.g. to handle KEGG gene ids, we declare an instance of both classes `map:Subject` and `bio:KEGGGeneId`. To create an AND fork, we declare an instance of `map:IAG` that `map:entails` more than one instance of `map:Map`. Both mechanisms are shown in the following RDF Sample, which is an excerpt of *Map E* from Fig.5:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    ... >
  <map:Map rdf:about="MapE">
    <map:composedOf>
      <map:Section>
        <map:hasTarget>
          <map:UserIntention>
            <map:hasSubject>
              <map:Subject>
                <rdf:type rdf:resource="bio:GeneDescription" />
              </map:Subject>
              <map:Subject>
                <rdf:type rdf:resource="bio:PathwayDescription" />
              </map:Subject>
            </map:hasSubject>
            <map:hasAction>
              <map:Action>
                <rdf:type rdf:resource="tech:Get" />
              </map:Action>
            </map:hasAction>
          </map:UserIntention>
        </map:hasTarget>
        <map:hasStrategy>
          <map:Strategy>
            <map:hasAttribute>
              <map:Attribute>
                <rdf:type rdf:resource="bio:KEGGService" />
              </map:Attribute>
            </map:hasAttribute>
          </map:Strategy>
        </map:hasStrategy>
        <map:refinedBy>
          <map:IAG>
```

```
            <map:entails
              rdf:resource="http://.../example.rdf#MapF" />
            <map:entails
              rdf:resource="http://.../example.rdf#MapG" />
          </map:IAG>
        </map:refinedBy>
        ...
      </map:Section>
      ...
    </map:composedOf>
  </map:Map>
</rdf:RDF>
```

3.2 Reasoning on processes

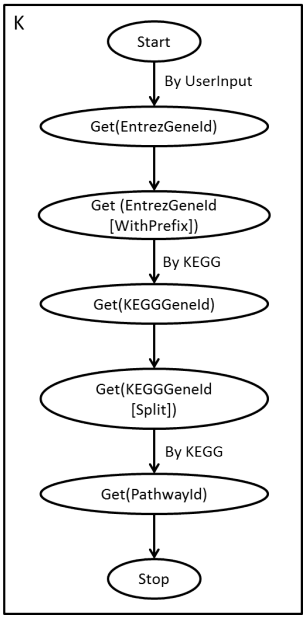

Fig. 8. A map example to illustrate advance searching. This example was inspired by a Taverna scientific workflow found on myExperiment.org at the following URL: http://www.myexperiment.org/workflows/15.html.

On most online sharing platforms, content sharing is done in three distinct ways:

1. automated content discovery through recommender systems,

2. direct recommendations from a user to another and
3. active content discovery a.k.a. searching

The first two types of content sharing are fairly low-level concerns. We intend to implement those features in our platform in due time. As for searching, we believe that to truly assist sharing and reuse, we have to provide not only keyword-based searching, but also much more advanced searching features. To illustrate such features, let us consider the maps represented in Fig.5, Fig.7 and Fig.8.

If we want to look for "all the processes where a gene description is obtained from a gene identifier", using keywords, we might well get false positives (e.g. processes where gene descriptions and gene ids are both inputs) and false negatives (e.g. processes where gene description is not mentioned in either keywords or tags). It is comparatively powerful and easy to express our query in SPARQL:

```
SELECT DISTINCT ?map WHERE
{
  ?map         map:composedOf  ?section            .
  ?section     map:hasSource   ?intention1         .
  ?intention1  map:hasSubject  ?subject1           .
  ?subject1    rdf:type        bio:GeneId          .
  ?section     map:hasTarget   ?intention2         .
  ?intention2  map:hasSubject  ?subject2           .
  ?subject2    rdf:type        bio:GeneDescription .
}
```

Given this query and the maps shown in this paper, a SPARQL engine would correctly retrieve the maps *E*, *F* and *F'*.

A benefit of linking to ontologies is that we can use the powerful reasoning mechanism of inference. For instance, our dummy technical ontology states that *tech:Split*, *tech:Merged*, *tech:NoNull* and *tech:NoDuplicate* are all subclasses of *tech:ListFeature*. In practice, it means that whenever a resource is declared to be of type *tech:Split* or another of those subclasses, the system can infer that it is also of type *tech:ListFeature*. Now, we want to find "all processes where we manipulate lists of gene identifiers", taking a look at our dummy technical ontology, we choose to use *ListFeature* to detect list manipulation. The query is again easy to express in SPARQL:

```
SELECT DISTINCT ?map WHERE
{
  ?map         map:composedOf  ?section            .
  ?section     map:hasSource   ?intention          .
  ?intention   map:hasSubject  ?subject            .
  ?subject     rdf:type        bio:GeneId          .
  ?subject     map:hasFeature  ?feature            .
  ?feature     rdf:type        tech:ListFeature    .
}
```

Given this query and the maps shown in this paper, a SPARQL engine would correctly retrieve the maps A, I, J and K.

Advanced search is but a small part of what we could do with semantically annotated maps and yet it already dwarfs keyword-based search in expressivity.

4 Conclusion and Future Work

In this paper, we presented an intentional approach to enhance sharing and reuse of in-silico experiments.

We extended the Map model in two ways: (1) we made fork specification both explicit and mandatory so as to allow automatic handling of maps and (2) we introduced the notion of parameterized maps to further ease and enhance reuse and repurposing. Both extensions could be useful outside the domain of in-silico experiments. We proposed a translation of the Map model into RDFS, in order to reason on maps, which would not be possible with graphical representations and is much more powerful than keyword-based comparison and lookup. We validated our version of the Map model by applying it to many in-silico experiments represented by scientific workflows on myExperiment.org.

We will delve into other faces of reasoning on maps, such as map comparison and combination, and work on a user interface to our SPARQL-based advanced search feature. Tasks of in-silico experiments are often operationalized by web services such as those provided by KEGG[3] and Entrez[4]. We aim to provide users with alternative suggestions and service discovery, thus we will investigate static and dynamic linking between maps and services.

References

1. Hollingsworth, D.: The workflow reference model. Technical report, Workflow Management Coalition (1993)
2. Yu, J., Buyya, R.: A taxonomy of scientific workflow systems for grid computing. J. ACM Sigmod Record 34, 44–49 (2005)
3. Taylor, I.J., Deelman, E., Gannon, D.B., Shields, M.: Workflows for e-Science: Scientific Workflows for Grids. Springer-Verlag, London, (2007)
4. Barker, A., Van Hemert, J.: Scientific workflow: A survey and research directions. In: 7th International conference on Parallel processing and applied mathematics, pp. 746–753. Springer-Verlag (2007)
5. Nurcan, S., Edme, M.H.: Intention-driven modeling for flexible workflow applications. J. Software Process: Improvement and Practice 10, 363–377 (2005)
6. Business Process Modeling Notation (BPMN) v.2.0 Specification, `http://www.omg.org/spec/BPMN/2.0/`
7. Van der Aalst, W.M.P.: Formalization and verification of event-driven process chains. J. Information and Software technology 41, 639–650 (1999)
8. Rolland, C., Prakash, N., Benjamen, A.: A multi-model view of process modelling. J. Requirements Engineering 4, 169–187 (1999)

[3] `http://www.genome.jp/kegg/soap/`
[4] `http://www.ncbi.nlm.nih.gov/entrez/query/static/esoap_help.html`

9. Rolland, C.: Capturing system intentionality with maps. In: Krogstie, J., Opdahl, A.L., Brinkkemper, S.: Conceptual modelling in information systems engineering. pp. 141–158. Springer-Verlag, New York. (2007)
10. Lee, T.B., Hendler, J., Lassila, O.: The semantic web. J. Scientific American 284, 34–43 (2001)
11. Resource Description Framework (RDF) Primer, `http://www.w3.org/TR/rdf-primer/`
12. SPARQL Query Language for RDF, `http://www.w3.org/TR/rdf-sparql-query/`
13. RDF Vocabulary Description Language 1.0: RDF Schema, `http://www.w3.org/TR/rdf-schema/`
14. Corby, O., Faron-Zucker, C., Mirbel, I.: Démarches sémantiques de recherche d'information sur le Web. In: IC 2009: 20es Journés Francophones d'Ingénierie des Connaissances. Hammamet, Tunisia (2009)
15. Baker, P.G., Goble, C.A., Bechhofer, S., Paton, N.W., Stevens, R., Brass, A.: An ontology for bioinformatics applications. J. Bioinformatics 15, 510–520 (1999)
16. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T.: Gene ontology: tool for the unification of biology. J. Nature genetics 25, 25–29 (2000)

Appendix

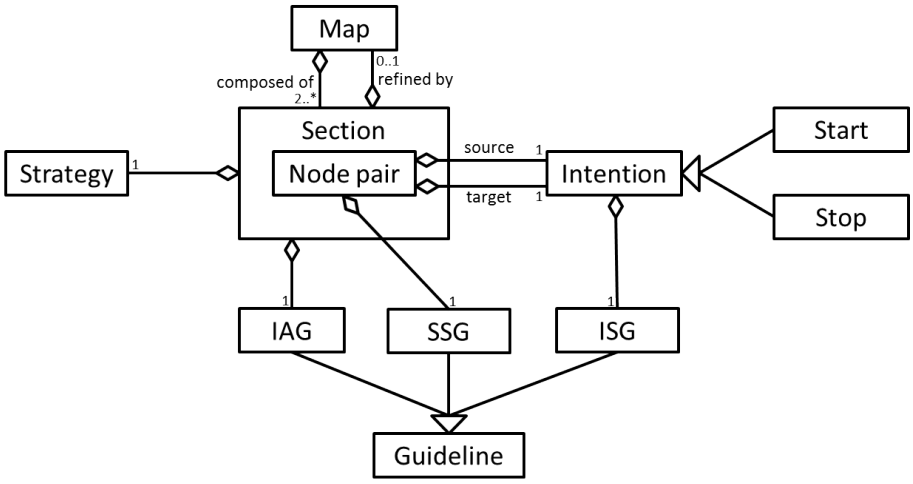

Fig. 9. A graphical representation of the original map meta-model, cf. [8] and [9] for more details.

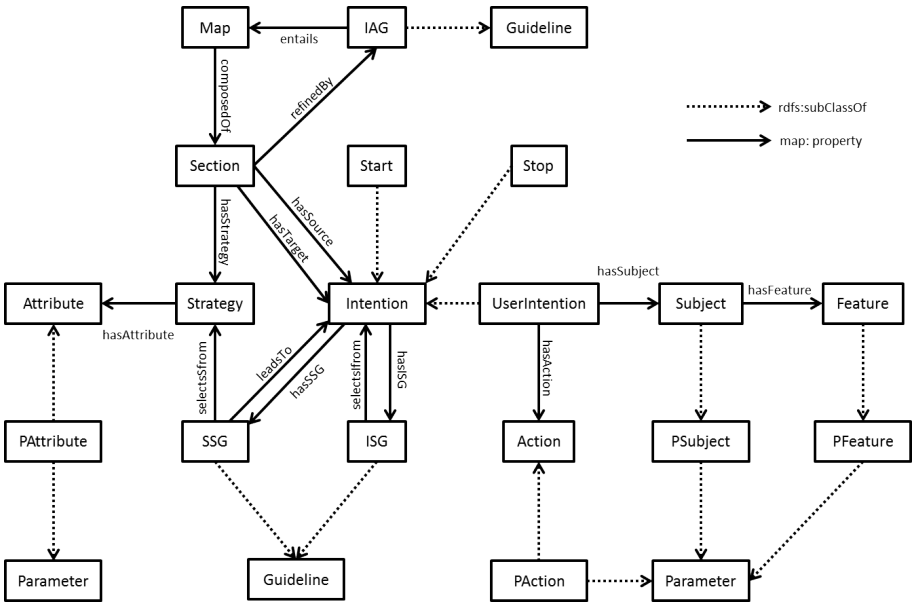

Fig. 10. Graphical representation of our current map ontology.

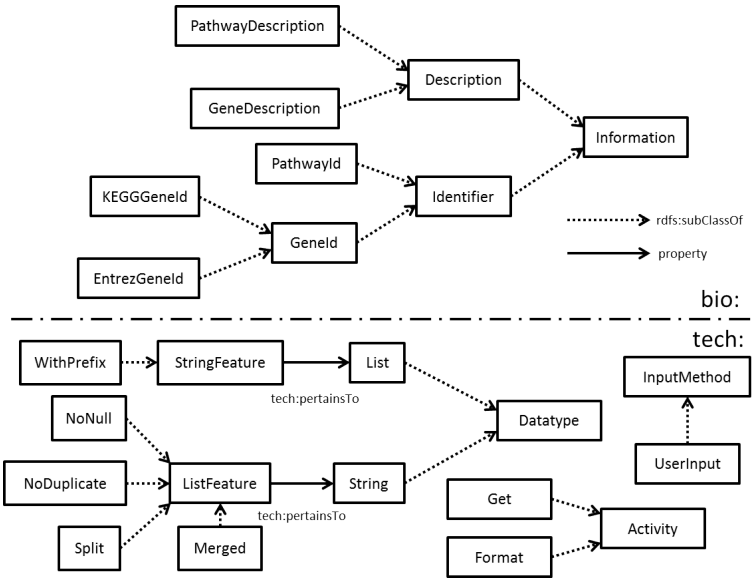

Fig. 11. Graphical representation of our ad hoc ontologies.