vldb manuscript No.
(will be inserted by the editor)

POESIA: An Ontological Workflow Approach for Composing Web Services in Agriculture

Renato Fileto[1,3], Ling Liu[2], Calton Pu[2], Eduardo Delgado Assad[3], Claudia Bauzer Medeiros[1]

[1] Institute of Computing, University of Campinas
Caixa Postal 6176, Campinas, SP, 13081-970 - Brazil
e-mail: {fileto|cmbm}@ic.unicamp.br

[2] College of Computing, Georgia Institute of Technology
801 Atlantic Drive, Atlanta, GA, 30332-0280 - USA
e-mail: {lingliu|calton}@cc.gatech.edu

[3] Embrapa - Brazilian Agricultural Research Corporation
Av. Dr. Andre Torsello, 209, Campinas, SP, 13083-886 - Brazil
e-mail: {fileto|assad}@cnptia.embrapa.br

The date of receipt and acceptance will be inserted by the editor

Abstract This paper describes the POESIA approach for systematic composition of Web services. This pragmatic approach is strongly centered in the use of domain specific multidimensional ontologies. Inspired by applications' needs, and founded by ontologies, workflows and activity models, POESIA provides well-defined operations (aggregation, specialization, and instantiation) to support the composition of Web services. POESIA complements current proposals for Web services definition and composition by providing a higher degree of abstraction with verifiable consistency properties. We illustrate the POESIA approach using a concrete application scenario in agro-environmental planning.

1 Introduction

Web services [23] are components for constructing next generation Web applications. These composite Web applications are built by establishing meaningful data and control flows among individual Web services. These data and control flows form *workflows* connecting components distributed over the Internet. However, there has been very limited research on the composition of Web services using workflow concepts and techniques. This

is partially due to the limitations of centralized control in traditional workflow management systems, which are inadequate for the scalability and versatility requirements of Web applications (e.g., dynamic restructuring of processes [16] and activities [14]).

This paper bridges this gap by applying advanced workflow and activity concepts in the composition of Web services, towards the construction of sophisticated Semantic Web applications. Our approach is called POESIA (Processes for Open-Ended Systems for Information Analysis), an open environment for developing Web applications using metadata and ontologies to describe data processing patterns developed by domain experts. These patterns specify the collection, analysis, and processing of data from a variety of Internet sources, thus providing building blocks for next generation Semantic Web applications.

The main contribution of the paper is POESIA's support of Web services composition, using domain ontologies with multiple dimensions (e.g., space, time, and object description). Tuples of terms taken from these ontologies, called *ontological coverages*, formally describe and organize the utilization scopes of Web services. An *utilization scope* is a specific context in which different data sets and distinct versions of a repertoire of services can be used. In POESIA, Web services are composed under these scopes, through well-defined operations such as specialization and aggregation. Rules based on the correlation of utilization scopes and their ontological relationships enable systematic means to verify the semantic and structural consistency of Web services compositions. POESIA ontologies are furthermore used in the determination of the granularities for selecting and integrating data and processes, as well as helping to describe their semantics.

The second main contribution consists in showing how POESIA solves some open issues in Web services composition. This is done through the modeling of a substantial application of practical impact using POESIA. Our application is in the area of environmental information systems, specifically, agricultural zoning – the determination of lands' suitability for important crops. Agricultural zoning is a challenging application for several reasons. First, several kinds of heterogeneous scientific data streams, such as meteorological measurements, are gathered continuously in large volumes and correlated for specific temporal and spatial conditions. Second, these data sources are distributed over the Web, increasingly through Web services. Third, agricultural zoning is a cooperative (distributed) decision making process, involving experts from several fields. Finally, it requires continuous processing since the situation is frequently reevaluated depending on temporal (seasonal) changes.

POESIA is a contribution towards the realization of the vision of the semantic Web for scientific applications. It allows the partial automation of some expert reasoning for organizing, reusing and composing not only data, but also the Web services that provide access to and process these data.

The remainder of this paper is organized as follows. Section 2 describes our example application. Section 3 defines the domain ontologies and on-

tological coverages, which are the basis of our approach. Section 4 presents the POESIA approach to specify and reuse Web services. Section 5 outlines the main technical issues in the implementation of POESIA environment. Section 6 discusses related work and Section 7 concludes the paper.

2 Application Scenario

2.1 Agricultural Zoning

Agricultural zoning is a scientific process to determine lands suitability, in a geographic region, for a collection of crops. This process classifies the land into parcels, according to their suitability for a particular crop, and the best time of the year for key cultivation tasks (such as planting, harvesting, pruning, etc). The goal of agricultural zoning is to determine the best choices for a productive and sustainable use of the land, while minimizing the risks of failure. However, some constraints may impose inevitable trade-offs that lead to compromises (e.g., short term productivity versus long term sustainability). Typically, agricultural zoning requires looking at many factors such as regional topography, climate, soil properties, and crop requirements. Additional concerns include interactions with wildlife, environmental preserves, as well as social and market impact.

As illustrated in Section 2.2, agricultural zoning is a complex process consisting of intricate interactions among a variety of data sources. The process is built by cooperation of experts from many scientific and engineering disciplines. For example, agronomists contribute with planting techniques and crop management models. Biologists provide crop growth and nutrient requirements. Statisticians provide risk management analysis for potential crop failures (e.g., due to severe weather). Environmental scientists analyze the impact of crop selection over the environment for both short and long term. These and other scientists and engineers bring together their expertise and a variety of computational and data analysis tools to build an agricultural zoning model.

At run-time, an agricultural zoning process obtains relevant data from a variety of heterogeneous sources, primarily sensors that collect data on physical and biological phenomena (e.g., weather stations, satellites, and laboratory automation equipment). Since gathering and processing real-time data can be costly, database systems and existing documents in different formats are frequently used as alternative sources. In any case, large amounts of fine grained data are usually required for extracting the needed information. Both data and data processing tools can be encapsulated and provided through Web services. In summary, agricultural zoning combines tools and services developed by a diverse set of scientists and integrates data from many heterogeneous sources through coordinated activities, as described by POESIA.

Agricultural zoning has been a labor-intensive process that is both expensive and slow to develop, due to the complexities enumerated above.

This is a serious problem, since it is an extremely important problem for a country with many commercial crops such as Brazil. Suppose we want to produce an agricultural zoning model for the top 20 crops for each region. Let us consider the 10 major varieties of each crop (these varieties usually have different weather and soil requirements). Simply dividing Brazil according to state boundaries (27 states) will result in more than 5000 models. It is clear that we need a systematic way to develop and maintain these models since manual processes will be too expensive and error-prone.

2.2 Case Study

Figure 1 illustrates a specific agricultural zoning process, namely, the land suitability for *Coffea arabia* in the Center-South region of Brazil. *Coffea arabia* is the main species of coffee produced by Brazil. Although coffee is no longer the country's number one export product, it remains one of the major farm export products due to the high commercial value of good coffee. The zoning process for *Coffea arabia* is composed of several distributed and cooperating activities, represented by ellipses. Data from several sources are processed by these activities and the results generated by each activity are transferred to other activities or data repositories.

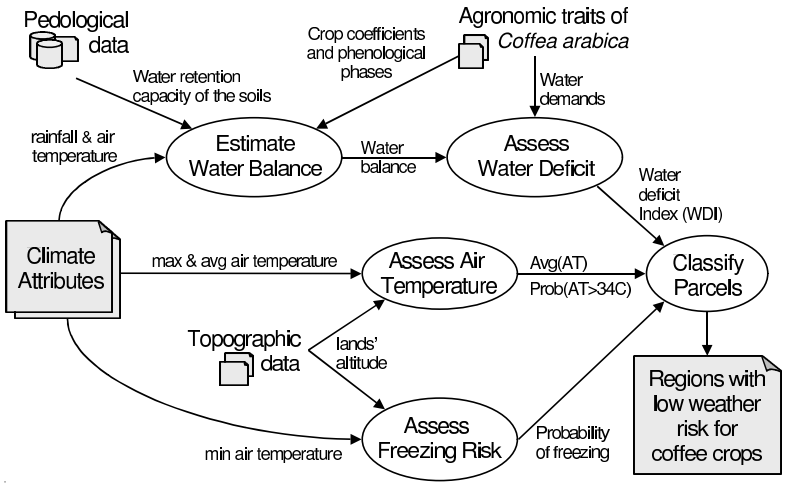


Fig. 1 Determining land suitability for *Coffea arabia* in Brazil's Center-South

According to domain experts [6,28], the most influential environmental factors for *Coffea arabia* are: (1) water available in the soil, (2) air temperature and (3) the risk of freezing. These factors are reflected in the structure of the land suitability process in figure 1, which relies on a data warehouse of climate attributes to obtain aggregated values of measurements, such as

maximum, minimum and average temperature and total rainfall, in appropriate time granularities. This warehouse is a composite Web service encompassing resources for collecting and maintaining climate data from several regions and institutions. It serves as input to three activities which can be executed in parallel – *Estimate Water Balance*, *Assess Air Temperature* and *Assess Freezing Risk*. Activity *Estimate Water Balance* takes the expected rainfall and the average air temperature for each month of the year, the water retention capacity of the soils and some phenological coefficients of coffee plants (collected from legacy database systems and scientific publications in agronomy, respectively), to calculate the water balance – a measurement of the expected amount of moisture available in the ground through the year. *Estimate Water Balance* is followed by *Assess Water Deficit*, which compares the data from water balance with the water demands of the plants during their successive phenological stages, producing the water deficit index (`WDI`) – a measurement of the expected deficit of water for the crop throughout the year.

In a similar way, activities *Assess Air Temperature* and *Assess Freezing Risk* use other climate data and topographic data, to produce the average air temperature, the probability of air temperature exceeding 34 degrees Celsius, and the probability of freezing. These partial results (indices and probabilities) are visualized as maps, showing the distribution of the relevant measurements or estimations across the region. When all these activities finish and deliver their results, activity *Classify Parcels* fuses these partial results to determine the suitability of the expected environmental conditions across the lands for the crop.

The data sources and activities for agricultural zoning may be dispersed across different sites over the Internet. Furthermore, these processes are sensitive to crop, location and time, i.e., they depend on the species and variety of the crop, the environmental characteristics of the region, and the opinion of the experts involved. The granularities for which these processes are defined are usually not uniform. Indeed, for some crops it is possible to devise a generic zoning process, while other crops require specific processes for each plant variety. Similarly, certain zoning processes are defined for vast regions and others for specific land parcels.

The map of figure 2, borrowed from [6], shows the land suitability results for *Coffea arabia* in the state of Paraná. It shows, for instance, that in the southern area of the state, one freezing event happens every 2 years in average. Freezings can impair the productivity and even kill coffee trees, rendering that area unsuitable for coffee cultivation. Governments and financial institutions rely on this kind of information, for instance, to define and enforce adequate loan granting policies. These policies direct farmers to choices and practices that contribute to lessen risks and increase the productivity of their enterprises. Experiences in sectors of Brazilian agriculture [18] in the last few years corroborate the economic advantages of adopting this scientific approach to agricultural zoning.

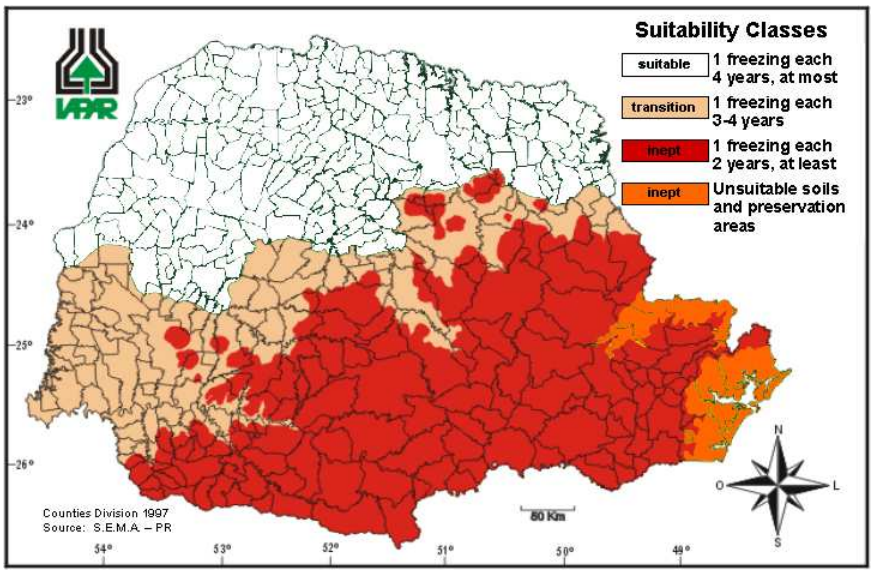


Fig. 2 Land suitability map for *Coffea arabia* in Paraná State

2.3 Technical Challenges

In our application example, the semantics of data are interrelated with the processes which manipulate them, so that data and processes cannot be completely decoupled. Interconnected activities cooperate with each other to process data collected from several heterogeneous distributed sources, giving rise to distributed processes whose complexity requires their organization in several abstraction levels. The outputs of a process can contribute to the inputs of other processes. The data sources to be taken into account and the resulting information, for each specific application, are dynamically defined by user requirements and contingent on climatic conditions. The analysis of the results gives feedback to improve the process or devise new ones. However, despite the numerous variants of these processes, some patterns can be recognized.

These scientific processes are in fact vast and distributed efforts for data integration and fusion. By *data integration*, we mean the transformations applied to heterogeneous data so that they can be analyzed together for some specific purpose. It does not imply that data must be coerced and congealed into a global schema. What matters is the correct interpretation and use of the data. *Data fusion* consists in applying some function to a collection of data values in order to produce other meaningful values (e.g., fuse the expected environmental conditions of a land parcel to determine its suitability for a crop). Our experience with scientific applications shows that data integration and fusion are scattered across the constituent activities of complex processes at distinct abstraction levels. Experts in this kind of context face many challenges, some of which are described below.

Identifying Resources Lack of catalogs and inspection mechanisms to find and reuse available Web resources to solve each particular problem.

Systems Interoperability Domain experts and technicians waste time converting data among formats of different tools. This effort should be spent on application specific issues.

Data Traceability There is no means to track data provenance, i.e., their original source and the way they were obtained and processed. This hampers the evaluation of whether the quality of a data item satisfies the requirements of a particular application.

Process Documentation and Execution Processes are rarely documented. When this is done, the specifications produced are either not broad enough for giving a general view of the processes or not formal enough to allow the automatic repetition of the process with different data sets.

Process Versatility There should be schematic means to reformulate processes on the fly. This kind of decision support systems rely on continuous feedback to improve the processes – as data keep arriving and results are produced, the processes may evolve.

Adaptation and Reuse Mechanisms for adaptation and reuse of Web services could boost productivity and enhance the quality of the results.

These issues are common to several kinds of applications involving distributed processes over the Web. The following sections describe the POESIA approach for handling some of these issues.

3 Ontological Delineation of Utilization Scopes

Ontologies [10] describe the meaning of terms used in a particular domain, based on semantic relationships observed among these terms. In the POESIA approach, they play a crucial role in composing Web services. Concretely, ontologies delineate the utilization scopes of data sets and processes, and orient the refinement and composition of Web services. An *utilization scope*, or *scope* for short, is a context in which different data sets and distinct versions of a repertoire of services can be used. In this section, we describe the structure of our multidimensional ontologies, and how they delineate and correlate utilization scopes. These are the foundations of our scheme to catalog and reuse components, and ensure the semantic consistency of the resulting Web services compositions.

3.1 Semantic Relationships between Words

Let Ω be a set of simple and/or composite words referring to objects or concepts from a universe of discourse U. *Objects* are specific instances (e.g. `Brazil`). *Concepts* are classes that abstractly define and characterize a set of instances (e.g. `Country`) or classes. The *universe of discourse* gives a context where the meaning of each word $w \in \Omega$ is stable and consistent.

The study of linguistics defines several semantic relationships between words. We consider the following subset in this work:

Synonym Two words are *synonym* of each other if they refer to exactly the same concepts or objects in U.

Hypernym/hyponym A word w is *hypernym* of another word w' (conversely w' is *hyponym* of w), if w refers to a concept that is a generalization of the concept referred to by w' in U. Hyponym is the inverse of hypernym.

Holonym/meronym A word w is *holonym* of w' (conversely w' is *meronym* of w) if w' refers to a concept or object that is part of the one referred to by w in U. Meronym is the inverse of holonym.

Roughly speaking, synonym stands for equivalence of meaning, hypernym for generalization (IS_A) and holonym for aggregation (PART_OF). For example, in the agriculture realm, `Cultivar` is a *synonym* of `Variety of Plant` and `Crop` is *hypernym* of `Cultivar`.

A set of words Ω is said to be *semantically consistent* for the universe of discourse U and a set of semantic relationships Υ, if at most one semantic relationship of Υ holds between any pair of words in Ω. This ensures some coherence for the meanings of the words in Ω for U.

The semantic relationships defined above preserve certain properties. Let w, w' and w'' be any three words and θ denote one of the semantic relationships considered. Then, for a given universe of discourse U, the following conditions hold:

- w *synonym* w (reflexivity)
- $w\ \theta\ w' \ \wedge\ w'\ \theta\ w'' \ \Rightarrow\ w\ \theta\ w''$ (transitivity)
- w *synonym* $w' \ \wedge\ w'\ \theta\ w'' \ \Rightarrow\ w\ \theta\ w''$ (transitivity wrt synonyms)

These properties enable the organization of a set of semantically consistent words Ω according to their semantic relationships in a given universe of discourse U. The *synonym* relationship partitions Ω in a collection of subsets such that the words of each subset are all synonyms. The transitiveness of the *hypernym* and *holonym* relationships correlates the semantics of words from different subsets of synonyms, inducing a partial order among the words of Ω. The resulting *arrangement of semantically consistent words* is a directed graph G_Ω which expresses the relative semantics of the words of Ω for the universe of discourse U [8]. The nodes of G_Ω are the subsets of synonyms of Ω. The directed edges of G_Ω represent the semantic relationships among the words of different subsets. There is a directed edge from vertex $\Re$ to vertex $\Re'$ of G_Ω if and only if each word of $\Re$ is *hypernym* of all the words of $\Re'$ or each word of $\Re$ is *holonym* of all the words of $\Re'$.

Consider the case where all the words of Ω represent concepts. Then, an arrangement of semantically consistent words is called an arrangement of semantically consistent concepts. Figure 3 illustrates an arrangement of concepts for territorial subdivisions. It is an extract from a very large set of ontological concepts used by experts for developing agricultural applications.

The concepts appear in the rectangles. The edges representing hypernym relationships are denoted by a diamond close to the specific concept and the edges representing holonym are denoted by a black circle close to the component concept. This graph denotes that a `Country` is composed of a set of `States` or, alternatively, a set of `Country Regions`. A `Country Region` may be a `Macro Region`, an `Official Region` or another kind of region. `Macro` and `Official Regions` are composed of `States`, but a region of type `Metro Area` is composed of `Counties`. `Eco Region` and `Macro Basin` define other partitions of space, based on ecological and hydrological issues, respectively. There is no constraint on the geometry of the land parcels modeled according to these concepts, except the containment relationships implied by the *hypernym* and *holonym* relationships (e.g., each `state` must be inside one `country`).

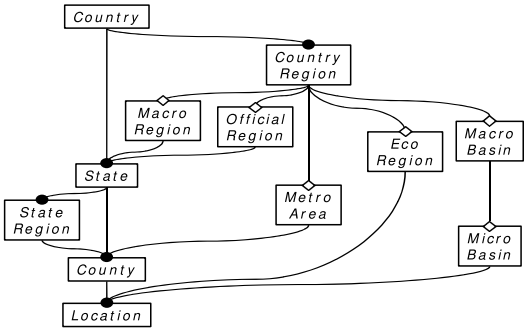
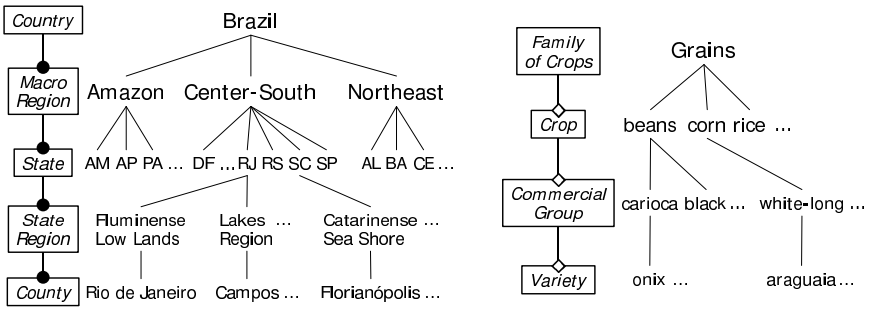


Fig. 3 An arrangement of concepts relative to territorial subdivisions

Given an arrangement G_Ω for a semantically consistent set of words Ω we say that a word $w \in \Omega$ *encompasses* another word $w' \in \Omega$, denoted by $w \models w'$, if and only if w and w' are in the same vertex of G_Ω (i.e., $w = w'$ or w *synonym* w') or there is a path in G_Ω leading from the vertex containing w to the vertex containing w' (i.e., there is a sequence of *hypernym* and/or *holonym* relationships relating the meaning of w to the more restricted meaning of w'). The encompass relationship is transitive [8]. According to figure 3, `Country` $\models$ `State`, `Country` $\models$ `County`, and so on.

Now, consider the instantiation of the concepts from figure 3. For example, the concept `Country` can be instantiated to `Brazil`, `State` to its states and so on. Let us call the instances of concepts *terms*. If there is a semantic relationship between two concepts of an arrangement of concepts, the same relationship holds between terms instantiated from these concepts. Therefore, the arrangement of semantically consistent concepts plays a role like that of a schema for the corresponding set of terms, inducing a similar structure (direct graph) to arrange the semantically consistent terms. Figure 4(a) illustrates a subgraph of the arrangement of concepts from figure 3, and one corresponding arrangement of terms, referring to Brazilian regions, states, and so on.

Terms are not restricted to instances of objects. Figure 4(b) illustrates an arrangement of concepts and one corresponding arrangement of terms, referring to crops and their varieties. Grains, beans, rice, corn, etc. do not refer to specific objects, but to concepts (or classes). This is an example of a specialization relationship between the terms and respective concepts. Further formalization of these notions is outside the scope of this paper, and appears in [8].



(a) Territorial Subdivisions (b) Agricultural Products

Fig. 4 Arrangements of semantically consistent terms

3.2 POESIA Ontologies and Ontological Coverages

A *POESIA ontology* is a collection of arrangements of semantically consistent terms. Each arrangement describes a particular dimension of the domain. For instance, figure 4 presents fragments of arrangements of terms for (a) the space and (b) the product dimensions, with the respective arrangement of concepts on the left of each hierarchy. On referring to a term of such a hierarchy, one must qualify the term with the corresponding concept of the respective arrangement of concepts, by using the expression *concept(term)*, in order to avoid ambiguity. Thus, `State(RJ)` refers to the Brazilian `state` called `Rio de Janeiro` (`RJ` is an acronym), while `County(RJ)` refers to the `county` of the same name.

An entire path in the hierarchy may be required to precisely indicate a term (e.g., if the same `county` name appears in different `states`). An *unambiguous reference to a term* of an ontology Σ is a path in one of the arrangements of terms of Σ. This path is expressed by the concatenated sequence of *concept(term)* vertices visited within it. This sequence, when taken as a string, must be unique across all the dimensions of the ontology. For instance, `State(RJ).County(Campos)` is an unambiguous reference to the `county` called `Campos` in the `state` called `Rio de Janeiro`. The term `Crop(beans)` is an unambiguous reference too, because there is only one `crop` called `beans`.

Finally, we are ready to define ontological coverages and their properties. An *ontological coverage* is a tuple of unambiguous references to terms defined

in a POESIA ontology. For example, `[Country(Brazil)]`, `[Crop(beans)]` and `[Country(Brazil),Crop(beans)]` are three different ontological coverages. Each of these ontological coverages expresses one *utilization scope*, or *scope* for short, i.e., a context in which a data set or service can be used.

An individual term of an ontological coverage expresses an utilization scope in a particular dimension. For instance, the term `Country(Brazil)`, defined in the space dimension, expresses the utilization scope "the whole `country` called `Brazil`". The universal coverage (denoted by ∞) is the empty tuple. It does not restrict the utilization scope in any dimension. The scope expressed by terms referring to the same dimension is a restriction of the universal scope to the union of the scopes expressed by the individual terms. For instance, the ontological coverage `[State(RJ),State(SP)]` expresses a scope which is obtained by the union of the scopes individually expressed by the terms `State(RJ)` and `State(SP)`. The scope expressed by terms referring to different dimensions restricts the universal scope to the intersection of the scopes expressed by the individual terms. For example, `[State(RJ),Crop(orange)]` restricts the scope to the intersection of the scopes defined by the spatial dimension term `State(RJ)` and the agricultural product dimension term `Crop(orange)`. To narrow the scope in a particular dimension one has to choose a more specific term (e.g., go from `State(RJ)` to `County(Campos)`). The absence of terms for a particular dimension means that the scope is not restricted for that dimension.

The semantic relationships among the terms of a POESIA ontology induce semantic relationships among ontological coverages. Given two ontological coverages, C and C', defined with respect to the same ontology Σ, C *encompasses* C', denoted by $C \models C'$, if and only if for each term $w \in C$ there is another term $w' \in C'$, such that $w \models w'$ (where w and w' are in the same dimension of Σ). For example, `[Country(BR)]` $\models$ `[Country(BR).Region(CS)]`, i.e., the whole country encompasses its Center-South region.

The encompass relationship between ontological coverages is transitive, inducing a partial order among coverages referring to the same ontology [8]. The universal coverage encompasses any other. Thus, $\infty \models$ `[Country(BR)]`, `[Country(BR)]` $\models$ `[Country(BR),Crop(beans)]` and so on. One can also evaluate the *equivalence of ontological coverages*. Two ontological coverages C and C' are equivalent (denoted by $C \equiv C'$), if and only if they encompass each other (i.e., $C \models C'$ and $C' \models C$). This occurs if each term in C has a synonym in C' and vice-versa. For example `[Country(Brazil)]` $\equiv$ `[Country(BR)]`, because `BR` can be used as a synonym of Brazil.

Figure 5 presents an entity-relationship diagram for POESIA ontologies and the ontological coverages defined according to such ontologies. It shows that a POESIA ontology has one or more dimensions. The domain specific terms for each dimension are organized in an arrangement of semantically consistent terms. The qualifiers of these terms, i.e., the concepts defining the classes of terms, are organized in an arrangement of semantically consistent

concepts for each dimension. An ontological coverage is a tuple of terms taken from one or more dimensions of an ontology.

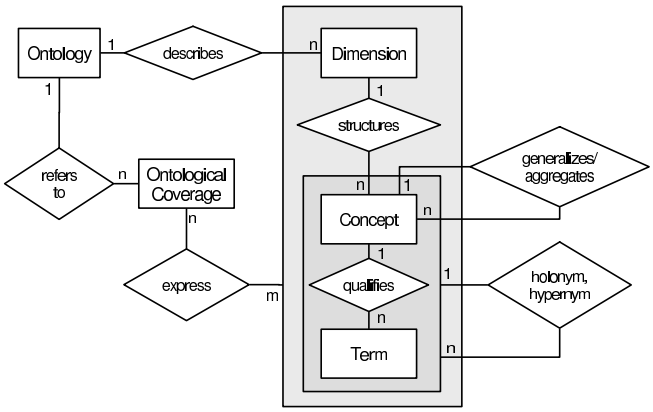


Fig. 5 A schema for POESIA ontologies and ontological coverages

4 The POESIA Activity Model

4.1 Overview

The basic construct of the model is the *activity pattern*. It may refer to any kind of data processing task – computational and/or manual. These tasks are performed in an open environment, comprehending several platforms. In POESIA, activity patterns are implemented as Web services.

An activity pattern has a set of communication ports, called *parameters*, to exchange data with other activity patterns and data repositories. Each parameter of an activity pattern refers to a Web service encapsulating a data source or sink for that particular pattern. Each input parameter is associated with outputs of another activity pattern or with a data repository. Conversely, each output parameter is associated with inputs of another activity pattern or with a data repository.

POESIA employs aggregation, specialization and instantiation of activity patterns to organize and reuse the components of processes as proposed in [14,13]. These mechanisms determine how processes can be composed and adapted. Activity pattern composition is depicted by a hierarchical graph, where intermediate nodes are composite patterns and leaves are atomic or simple patterns. The latter must be specialized before they are decomposed.

A hierarchy of activity patterns, i.e., of Web services, is called a *process framework*. Each activity pattern of a process framework is associated with an ontological coverage, that expresses its utilization scope, in order to drive the selection and reuse of components. A process framework must

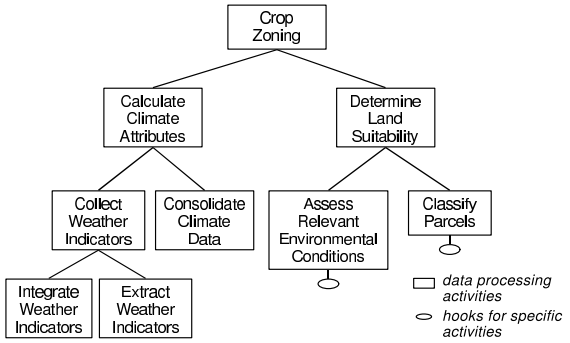


Fig. 6 Process framework for agricultural zoning

be refined, adapted to a particular situation, and instantiated before execution. POESIA provides some rules to check the semantic consistency of process frameworks and instantiated processes, based on correlations of the ontological coverages of their constituents. For example, the ontological coverages of all the components of a process framework must be compatible with (encompass or be encompassed by) the ontological coverage of the highest activity in the hierarchy.

Let us illustrate these notions with a simple example. Figure 6 presents a simplified framework for agricultural zoning. It shows that the major components of *Agricultural Zoning* are *Calculate Climate Attributes* and *Determine Land Suitability*. The former, which is composed of *Collect Weather Indicators* and *Consolidate Climate Data*, collects weather data from a variety of Web services, and consolidates them into the Web service of lands' climate attributes. The activity pattern *Determine Land Suitability* takes the climate attributes, along with other data relevant for one specific crop, to determine the most appropriate lands for that crop.

This framework applies to the zoning of any crop. In order to obtain instantiated processes for specific crops, one must adapt the constituent activities to the peculiarities of that crop. For example, the relevant environmental conditions for zoning coffee (discussed in section 2.1) are different from those for zoning rice. Thus, *Determine Land Suitability* and its two constituents must be specialized for each crop. In addition, an specific activity must be defined to asses each relevant environmental condition for each crop. On the other hand, the activities that calculate climate attributes do not require adaptation, as one general Web service can supply climate data to several specific services for determining land suitability for different crops. The ontological coverages associated with the Web services enable automated means to check their compatibility for composition, with respect to their utilization scopes. This helps domain experts to organize and compose the services necessary for their applications and factor their solutions to reduce costs, according to domain specific concepts and reasoning.

4.2 Activity Pattern

An *activity pattern* is an abstraction that defines the structure and behavior of a collection of instances of data processing activities implemented as Web services, in a similar way as a class does for instances of objects [13]. Activity patterns also resemble software design patterns [9], in the sense that each activity pattern is designed to solve a well defined category of problems, in a particular utilization scope. Definition 1 depicts the structure of an activity pattern.

Definition 1 *An* **activity pattern** α *is a five-tuple:*

$$(NAME, COVER, IN, OUT, TASK)$$

where:

$NAME$	*is the string used as the name of* α.
$COVER$	*is the ontological coverage of* α, *i.e., expresses its utilization scope.*
IN	*is the list of input parameters of* α.
OUT	*is the list of output parameters of* α.
$TASK$	*describes the processing chores that* α *does.*

$NAME$, $COVER$, IN and OUT represent the *external interface* or *signature* of the pattern. $TASK$ specifies the behavioral semantics of the activity pattern, including the composition semantics and the execution dependencies between component patterns.

Figure 7 presents the textual specification of an activity pattern to determine land suitability for an arbitrary crop, whose $NAME$ is `DetLandSuitability`, $COVER$ is `[Country(BR), Cons(RNA)]`, i.e., Brazil, according to the methodology of RNA[1], the IN and OUT parameters are specified as $INPUTS$ and $OUTPUTS$, and $TASK$ is composed of two activity patterns – `AssessEnvCond` and `ClassifyParcels` – invoked within `DetLandSuitability`. These component patterns are assumed to be declared elsewhere. Figure 7 also shows a few special keywords. The `#DEFINE` clause specifies an alias for a URL that is frequently used in the pattern specification. `LOCAL` declares the internal variables of the pattern. The delimiters `BEGIN TASK` and `END TASK` enclose the specification of the $TASK$. `COMPOSITION` enumerates the constituent patterns of a composite pattern. `EXECUTION DEPENDENCIES` establishes the relative order of execution of the constituent patterns. `EXECUTION DEPENDENCIES` and `TASK DESCRIPTION` are optional. Another example of task description is provided in Section 4.4.

An activity pattern, which is implemented as a Web service, is uniquely identified by the URL of the site holding it, its name and its ontological coverage. All the data exchanged by activity patterns can be viewed in XML.

[1] RNA stands for National Meteorological Network, a consortium of Brazilian institutions linked to agricultural research.

```
#DEFINE RNA "http://www.agric.gov.br/rna/pub_docs"

ACTIVITY_PATTERN DetLandSuitability [Country(BR), Cons(RNA)]

  INPUTS
    ClimAttr:  "RNA/clim_info.wsd";
    LandsInfo: "RNA/lands_info.wsd";
    CropInfo:  "RNA/crops_info.wsd";
  OUTPUTS
    Zoning: "RNA/agric_zoning.wsd";
  LOCAL
    EnvCond: "RNA/env_cond.wsd";

  BEGIN TASK
    COMPOSITION
      AssessEnvCond (IN: ClimAttr, LandInfo, CropInfo;
                     OUT: EnvCond);
      ClassifyParcels(IN: EnvCond; OUT: Zoning);
    EXECUTION DEPENDENCIES
      AssessEnvCond PRECEDES ClassifyParcels;
  END TASK;

END ACTIVITY_PATTERN;
```

Fig. 7 Activity pattern to *Determine Land Suitability* for an unspecified crop

Each parameter is associated with some description of the capabilities of the corresponding Web service – like the `.wsd` (Web Service Description) files referenced in figure 7. The service descriptions must provide links to DTD or XML-Schema specifications that define the types of all data elements that can be exchanged via the respective parameters. Links are defined as URLs.

The description of each activity pattern parameter includes the description of the interface of the services that can be bound to that parameter, in order to support more sophisticated communication than just transferring packets of semi structured data. For example, the service that supplies climate data to `DetLandSuitability`, denoted by the parameter `ClimAttr`, allows the target to pose queries (e.g., OLAP operators) specifying filters and granularities for the data to be transferred (e.g., in order to get the average temperature in a certain region for each month). Notice that data filters and granularities can also be expressed by ontological coverages. This makes POESIA ontologies central not only as a means to organize data and services, but also to define the communication interfaces for Web services. The designer of a process can refer to published Web service and schema descriptions, or develop his own descriptions to fulfill specific demands. This encourages standardization, at the same time that confers flexibility to Web services and data representation.

The following subsections present the operations for composing activity patterns (implemented as Web services), and some rules to check the semantic consistency of these compositions. The emphasis is on the correlation of the utilization scopes of the services, expressed by their ontological coverages. Some aspects of our workflow specification language, such as synchronizing mechanisms, are outside the scope of this work. They are still

under development, with the incorporation of more elaborate constructs borrowed from other works such as WSFL [26] and BPEL4WS [24].

4.3 Activity Pattern Aggregation

In POESIA, a complex activity pattern is defined as an aggregation of a set of component activity patterns. A component activity pattern can itself be a complex activity pattern or an elementary activity pattern. Figure 8 shows the activity pattern *Determine Land Suitability*, which is an aggregation of the activity patterns *Assess Environmental Conditions* and *Classify Parcels*.

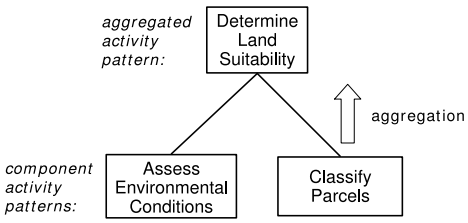


Fig. 8 An aggregation of two activity patterns

When decomposing an activity pattern in its constituents (or conversely, composing an activity pattern from the components) we have to make sure that there is no conflict among names and ontological coverages of the activity patterns involved, and all parameters are connected.

Definition 2 *Activity pattern α is an* **aggregation** *of the activity patterns $\beta_1, \cdots, \beta_n$ $(n \geq 1)$ iff the following conditions are verified (let $1 \leq i, j \leq n; i \neq j$ for each condition):*

1. $\forall\ \beta_i : NAME(\alpha) \neq NAME(\beta_i)\ \vee\ COVER(\alpha) \neq COVER(\beta_i)$
2. $\forall\ \beta_i, \beta_j : NAME(\beta_i) \neq NAME(\beta_j)\ \vee\ COVER(\beta_i) \neq COVER(\beta_j)$
3. $\forall\ \beta_i : COVER(\alpha) \models COVER(\beta_i)\ \vee\ COVER(\beta_i) \models COVER(\alpha)$
4. $\forall\ p \in IN(\alpha) : \exists\ \beta_i$ *such that* $p \in IN(\beta_i)$
5. $\forall\ p \in OUT(\alpha) : \exists\ \beta_i$ *such that* $p \in OUT(\beta_i)$
6. $\forall\ \beta_i,\ p' \in IN(\beta_i) : p' \in IN(\alpha)\ \vee\ (\exists\ \beta_j$ *such that* $p' \in OUT(\beta_j))$
7. $\forall\ \beta_i,\ p' \in OUT(\beta_i) : p' \in OUT(\alpha)\ \vee\ (\exists\ \beta_j$ *such that* $p' \in IN(\beta_j))$

We call α an aggregated (or composite) activity pattern and each β_i a constituent (or component) activity pattern.

Definition 2 states that an activity pattern α is defined as an aggregation of n component activity patterns $\beta_1, \ldots, \beta_n$, if they satisfy the above mentioned seven conditions. Condition 1 says that the name and the ontological coverage of each constituent pattern β_i must be different from the name and the coverage of the aggregated activity pattern. Condition 2 specifies that the name and the coverage of a constituent activity pattern can

uniquely distinguish itself from other constituent patterns of α. Condition 3 states that the ontological coverage of the composite pattern α must encompass the coverage of each constituent pattern β_i or vice-versa, i.e., the intersection of their utilization scopes is not null. Condition 4 ensures that every input parameter of α is connected to an input parameter of some constituent β_i. Similarly, condition 5 ensures that each output parameter of α is connected to an output parameter of some β_i. Finally, conditions 6 and 7 state that all parameters of constituent patterns must be connected to a parameter of other constituent or the aggregated pattern.

4.4 Activity Pattern Specialization

The descriptors of an activity pattern can be refined when specializing that activity pattern for a particular situation. Figure 9 illustrates an specialization of the activity pattern *Classify Parcels* for the crop *Coffea arabia*.

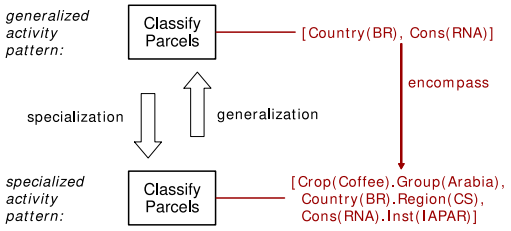


Fig. 9 A specialization of *Classify Parcels*

The specialization of an activity pattern can be formally defined by relationships similar to those used to define the aggregation abstraction.

Definition 3 *Activity pattern β is a* **specialization** *of the activity pattern α (conversely α is a generalization of β) iff the following conditions are verified:*

1. $NAME(\alpha) \neq NAME(\beta) \ \vee \ COVER(\alpha) \neq COVER(\beta)$
2. $COVER(\alpha) \models COVER(\beta)$
3. $\forall\ p \in IN(\alpha) : \exists\ p' \in IN(\beta)$ *such that* $p \vdash p'$
4. $\forall\ p \in OUT(\alpha) : \exists\ p' \in OUT(\beta)$ *such that* $p \vdash p'$

We call α the generalized activity pattern of β, and β a specialized activity pattern (version) of α.

Condition 1 of definition 3 states that the name and/or the ontological coverage of the generalized activity pattern α must be different from those of its specialized version β. Condition 2 states that the ontological coverage of α must encompass that of β. The notation $p \vdash p'$ in conditions 3 and 4 means that each parameter p' of β must refer to a Web service which is a

```
#DEFINE IAPAR "http://www.pr.gov.br/iapar/pub_docs"

ACTIVITY_PATTERN
ClassifyParcels [Crop(Coffee).Group(arabia),
                 Country(BR).Region(CS).State(PR),
                 Cons(RNA).Inst(IAPAR)]

  REFINES ClassifyParcels [Country(BR), Cons(RNA)]

  INPUTS
    EnvCond->WDI:          "IAPAR/wdi.wsd";
    EnvCond->AvgAT:        "IAPAR/avg_at.wsd";
    EnvCond->ProbHeat:     "IAPAR/prob_heat.wsd";
    EnvCond->ProbFreeze:   "IAPAR/prob_freeze.wsd";
  OUTPUTS
    Zoning->Zon_Coffee: "IAPAR/zoning_coffee.wsd";

  BEGIN TASK
    DESCRIPTION
      OVERLAY
        IF WDI <= 150 THEN "OK" ELSE "Water restriction";
        IF ProbHeat <= 30 THEN "OK" ELSE "Thermal restriction";
        IF AvgAT <= 24 THEN "OK" ELSE
           IF WDI <= 100 THEN "OK" ELSE "Thermal restriction";
        IF ProbFreeze <= 25 THEN "Low risk of freeze" ELSE
           IF ProbFreeze <= 50 THEN "Medium risk of freeze";
              ELSE "High risk of freeze";
  END TASK;

END ACTIVITY_PATTERN;
```

Fig. 10 *Classify Parcels* for *Coffea arabia* in Paraná

refinement of the Web service referred to by the corresponding parameter p of α. This refinement of Web services can refer to their capabilities or data contents. The exact relationship between the generic and the refined parameters is defined in the description of the corresponding Web services. Ontological coverages can be associated with these Web services, in order to express and correlate their utilization scopes.

Figure 10 shows the specialized version of the activity pattern *Classify Parcels* for *Coffea arabia*, according to the methodology of Paraná Agricultural Institute (IAPAR) [6], a member of RNA. The clause `REFINES` tells that this pattern is one specialization of the pattern `ClassifyParcels` with a wider utilization scope expressed by `[Country(BR),Cons(RNA)]`. Each parameter declared in the specialized version is explicitly related to the corresponding one of the generalized pattern. The notation `EnvCond->WDI` indicates that the parameter `WDI` of the specialized version is derived from the parameter `EnvCond` (the expected environmental conditions) of the generalized version of `ClassifyParcels`. The other input parameters of the specific version of `ClassifyParcels` also derives from the generic parameter `EnvCond`. The output parameter `ZonCoffee` of the specialized version is a refinement of the parameter `Zoning` of the generalized activity pattern. The `TASK DESCRIPTION` clause overlays logical conditions involving the measurements of the relevant environmental conditions for the crop.

4.5 The Combined Refinement Mechanism

The aggregation and specialization of activity patterns can be combined to define a complex activity pattern, whose constituents depend on the utilization scope to which the complex pattern is specialized. The definition of such a complex activity pattern must conform both the conditions of aggregation and the conditions of specialization. Figure 11 illustrates a refinement of the activity pattern *Assess Environmental Conditions* for *Coffea arabia*.

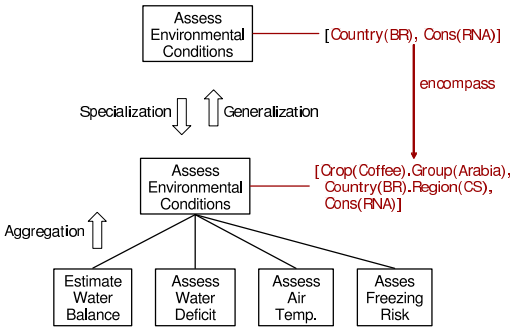


Fig. 11 Combining specialization and aggregation

Specialization and aggregation of activity patterns are intertwined. The specialization details the parameters and constituents of a pattern for a particular utilization scope, establishing a flat view, at a particular abstraction level, to express the cooperation of the constituent patterns. Problems related to parameter passing – type checking, parameter uniqueness and disambiguation – are solved by defining parameters' scopes just as in programming languages: a parameter's scope is local to the specification of activity pattern where it is defined.

Figure 12 shows the specialized version of `AssessEnvCond` (*Assess Environmental Conditions*). The input parameter `ClimAttr` appears in both the generalized and the specialized version. The `LandsInfo` parameter of the generalized version unfolds in `Relief` and `WaterRetSoil` in the specialization. `CropInfo` unfolds in `CropCoef` and `WaterDemands`. The output `EnvCond` of the generalized version unfolds in `WDI`, `AvgAT`, `ProbHeat` and `ProbFreeze`. The `LOCAL` parameter `WaterBal` is used to transfer data between `EstWaterBal` and `CalcWaterDeficit`. The binding of these parameters expresses the data flow illustrated in figure 1. The clause `EXECUTION DEPENDENCIES` states that `EstWaterBal` precedes `CalcWaterDeficit` and `ClassifyParcels` initiates after all the other constituents have finished.

4.6 Process Framework

In POESIA, activity patterns can be defined in terms of other activity patterns through activity patterns aggregation and specialization. As a result,

```
ACTIVITY_PATTERN
AssessEnvCond [Crop(Coffee).Group(Coffea arabia),
               Country(BR).Region(CS), Cons(RNA)]

  REFINES AssessEnvCond [Country(BR), Cons(RNA)]
  INPUTS
    ClimAttr:                "RNA/clim_info.wsd";
    LandsInfo->Relief:       "RNA/relief.wsd";
    LandsInfo->WaterRetSoil: "RNA/water_ret_soil.wsd";
    CropInfo->CropCoef:      "RNA/coffee_water_coef.wsd";
    CropInfo->WaterDemands:  "RNA/coffee_water_dem.wsd";
  OUTPUTS
    EnvCond->WDI:          "RNA/wdi.wsd";
    EnvCond->AvgAT:        "RNA/avg_at.wsd";
    EnvCond->ProbHeat:     "RNA/prob_heat.wsd";
    EnvCond->ProbFreeze:   "RNA/prob_freeze.wsd";
  LOCAL
    WaterBal: "RNA/water_bal.wsd";

  BEGIN TASK
    COMPOSITION
      EstWaterBal (IN: ClimAttr,WaterRetSoil,CropCoef;
                   OUT: WaterBal);
      CalcWaterDeficit (IN: WaterBal,WaterDemands;
                        OUT: WDI);
      AssessAirTemp(IN: ClimAttr; OUT: AvgAT,ProbHeat);
      AssessFreezeRisk (IN: ClimAttr,Relief; OUT: ProbFreeze);
    EXECUTION DEPENDENCIES
      EstWaterBal PRECEDES CalcWaterDeficit;
      (CalcWaterDeficit AND AssessAirTemp AND AssessFreezeRisk)
                                      PRECEDES ClassifyParcels;
  END TASK;
END ACTIVITY_PATTERN;
```

Fig. 12 *Assess Environmental Conditions* for *Coffea arabia* in Brazil's Center-South region

a hierarchy of activity patterns can be formed. We call such a hierarchy a process framework of the root activity pattern. Figure 13(a) shows a process framework to determine land suitability for *Coffea arabia* presenting only compositions of activity patterns. Figure 13(b) extends 13(a) by adding the hierarchies of specializations of some activity patterns in the hierarchy. We say that a hierarchy as that shown in figure 13(b) is multi-fold, because each of its activity patterns (nodes) can have two kinds of immediate subordinates: its constituent patterns and its specialized versions.

Definition 4 *A* **process framework** *is a directed graph* $\Phi(V_\Phi, E_\Phi)$ *satisfying the following conditions:*

1. V_Φ *is the set of vertices of* Φ
2. E_Φ *is the set of edges of* Φ
3. $\forall\ v \in V_\Phi : v$ *is an activity pattern*
4. $(\overrightarrow{v, v'}) \in E_\Phi \ \Leftrightarrow\ v'$ *constituent* $v\ \vee\ v'$ *specialization* v
5. Φ *is acyclic*
6. Φ *is connected*

Definition 4 establishes the structural properties of a process framework – a directed graph $\Phi(V_\Phi, E_\Phi)$ whose nodes represent the activity patterns

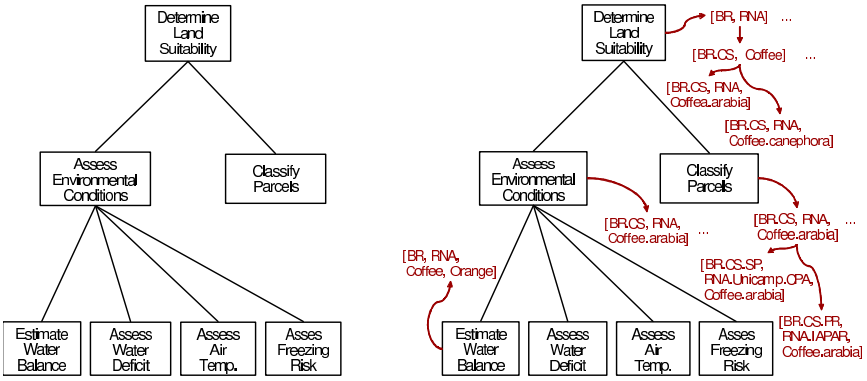


Fig. 13 Hierarchies of activity patterns for determining land suitability for *Coffea arabia*: (a) decomposition hierarchy; (b) multi-fold hierarchy or process framework

and the directed edges correspond to the aggregation and specialization relationships among these patterns. Condition 4 states that there is a directed edge (v, v') from vertex v to vertex v' in Φ if and only if v' is a constituent of v or v' is a specialization of v. Condition 5 states that no sequence of aggregations and/or specializations of patterns in Φ can lead from one pattern to itself. This restriction is necessary because aggregation and specialization can intermingle. In such a case, an aggregation may break the gradual narrowing of the utilization scopes done by specialization. Condition 6 guarantees the connectivity of the activity patterns participating in the process framework Φ.

Adaptation of a Process Framework

A process framework captures the possibilities for reusing and composing Web services to build consistent processes for different situations, in terms of utilization scopes, data dependencies and execution dependencies among components. The adaptation of a process framework for a particular scope consists in choosing (and developing if necessary) components to compose a process tailored for that scope.

Definition 5 *A* **process specification** $\Pi(V_\Pi, E_\Pi)$ *associated with an utilization scope expressed by an ontological coverage* C *is a subgraph of a process framework satisfying the properties:*

1. $\forall\ (v, \vec{v'}) \in E_\Pi : v'$ *constituent* v
2. $\forall\ v \in V_\Pi$:
   $(\nexists\ v' \in V_\Pi$ *such that* $(v, \vec{v'}) \in E_\Pi) \Rightarrow v$ *is atomic*
3. $\forall\ v \in V_\Pi : COVER(v) \models C$

Definition 5 states that a process specification Π is a subgraph of a process framework. Condition 1 states that Π is a decomposition hierarchy, i.e., all its edges refer to aggregations of activity patterns. Condition 2 states that all the leaves of Π are atomic patterns, otherwise Π would be missing some constituents for its execution. Condition 3 ensures that the ontological coverage of each pattern participating in Π encompass the coverage C associated with Π, i.e., the intersection of the utilization scopes of all the constituents of Π are equivalent or contain the utilization scope of Π.

Refinement and adaptation of process frameworks can alternate in practice. Frameworks, specific processes or individual activity patterns can always be reused to produce new or extended frameworks. Additionally, the development of activity patterns to contemplate specific needs, when adapting a framework, also contributes to enrich the repertoire of specialized patterns of a framework.

Process Instantiation

Note that all the elements of the POESIA model presented above are at the conceptual level. Thus, after adapting a process framework to produce a process specification for a particular situation, this process has to be instantiated for execution. Instantiating a process specification Π consists in assigning concrete Web services to handle the inputs and outputs of each activity pattern of Π, allocating sites where to execute the corresponding tasks and designating agents (humans or programs with the appropriate abilities and roles) to perform them.

The location of the concrete resources assigned to execute a process is independent of the locations of their descriptions. The selection of the concrete resources to perform the process, during its instantiation, confers an extra level of execution independence to POESIA. Once particular resources have been assigned, the specific formats and protocols used to connect them can be defined. This may be done by using the binding mechanisms of Web service specification languages like WSDL [25].

POESIA Meta Model

Figure 14 shows the POESIA meta model, which is an extension of the workflow reference model of the WfMC [11]. It summarizes, in bold, our extensions: (1) associate an ontological coverage with each activity pattern, and (2) associate Web services descriptions to the interfaces (parameters) of the activity patterns. This allows the organization of a repertoire of activity patterns according to their utilization scopes and helps to determine the services for reusing in specific situations and the rules to connect them.

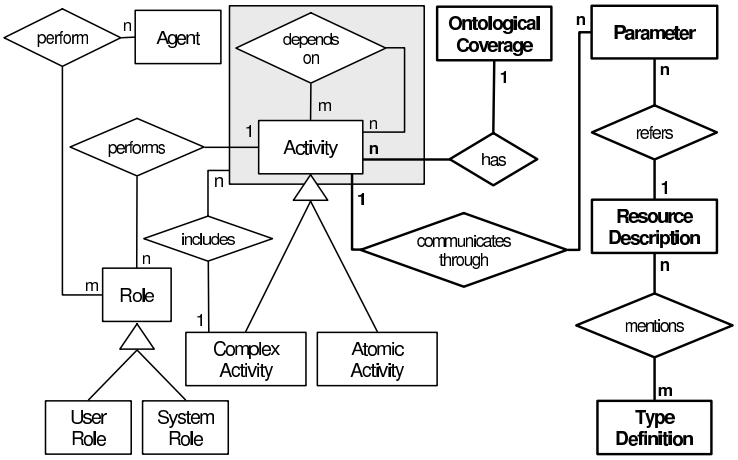


Fig. 14 The POESIA process definition meta model

5 Implementation Issues

A number of issues are important in the implementation of the POESIA approach to Web services composition. First, checking the correctness of the composition semantics. Second, mechanisms for composing Web services through ontology construction and ontology reasoning. Third, an efficient and scalable implementation architecture. In this section, we discuss how POESIA handles these issues.

5.1 Checking Specifications

Hierarchy of Activity Patterns

The aggregations and specializations of activity patterns must be checked for the properties expressed in definitions 2 and 3. The direct graphs corresponding to process frameworks must be acyclic and connected as stated in definition 4. Furthermore, the conditions expressed in definition 5 must be checked when adapting a framework for a particular utilization scope.

Figure 15 illustrates a process for zoning *Coffea arabia* in Paraná State. All the activity patterns in this structure, starting by its root, have compatible ontological coverages. The ontological coverage of *Agricultural Zoning* encompasses that of *Calculate Climate Attributes*, *Determine Land Suitability*, and so on. The activity pattern *Calculate Water Balance*, has a wider coverage including coffee and orange, i.e., the same pattern for calculating the water balance is used for both crops.

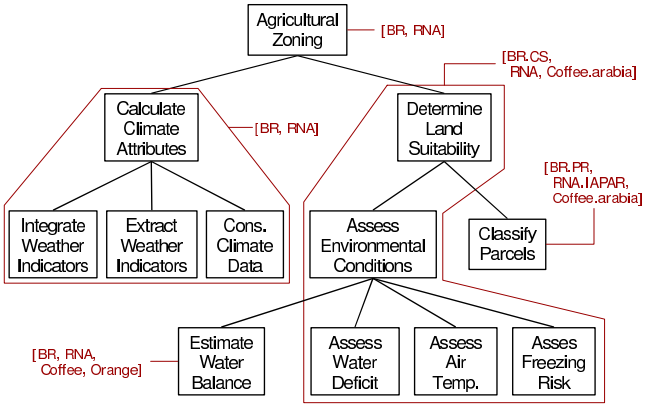


Fig. 15 Zoning *Coffea arabia* in Paraná State

Execution and Data Dependencies

The collection of execution dependencies among activity patterns, can be represented in a dependencies graph. Figure 16 presents the dependency graph for the process framework for zoning *Coffea arabia*. It shows that the execution of the activity pattern *Consolidate Climate Attributes* can be initiated only after successfully finishing the execution of *Integrate Weather Indicators* or *Extract Weather Indicators*, which provide data (from weather stations or remote sensing, respectively) for updating the climate attributes. When *Consolidate Climate Data* has done its work, *Estimate Water Balance*, *Assess Air Temperature* and *Assess Freezing Risk* can execute in parallel. The conclusion of *Estimate Water Balance* triggers the execution of *Assess Water Deficit*. *Classify Parcels* can only start executing after a successful execution of all the previous activities.

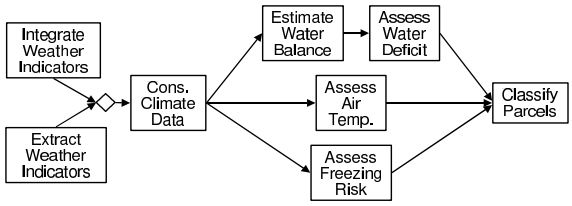


Fig. 16 Execution dependencies among activity patterns for zoning *Coffea arabia*

A similar dependency graph for the data dependencies is inferred from the connection of parameters amid process frameworks. These two graphs must be compatible. Individually, these graphs must be acyclic and connected. Properties relative to the structure and the dynamics of the execution and data dependencies among activity patterns can be evaluated

with algorithms based on Petri nets formalisms. For example, [22] proposes an algorithm to translate workflow graphs into WF-Nets, a class of Petri nets tailored towards workflow analysis. The verification of the properties of WF-Nets, allows the automatic detection of design errors in the corresponding workflow specifications. The absence of deadlocks in a workflow, for instance, is associated with the soundness property of the corresponding Petri net. Roughly speaking, the soundness property states that for every reachable state of the Petri net there must be a sequence of steps leading to the final state.

5.2 Composing Web services: An Implementation Perspective

A POESIA Web service can access a collection of existing Web services functioning as data sources for its processes and publish its own processes and data sets as Web services. Each POESIA-enabled Web site organizes its services description, composition and interconnection apparatus, according to the representation layers of the Semantic Web [7,19]. In the bottom layer, XML wrapping, source data are converted into XML, thus providing a syntax standard for semi structured data in the extensional level. The XML related standards confer versatility and expression power for representing and interrelating documents in the Internet. The second layer is the schemas and processes layer. It uses DTDs or XML-Schema to represent data sets in the intentional level, in order to factor the problems related with data heterogeneity. POESIA frameworks appear at the top of the second layer, and provide specific criteria based on utilization scopes to select services and check the semantic consistency of their connections. The third layer is the semantic description layer which describes the services, in a higher abstraction level, using RDF statements and process description standards like DAML-S [5,1]. These resource descriptions must conform to metadata standards and vocabularies, including domain specific ones. The vocabulary used in the first, second and third layers is defined in the fourth layer, which maintains a dictionary. The top layers of the Semantic Web infrastructure – namely logic, proof and trust – are not contemplated at this moment.

POESIA services in different sites can be logically arranged in successive abstraction levels. Figure 17 illustrates such a situation. The process specification stored in server *"A"* is composed of two cooperating activity patterns *"X"* and *"Y"*. Activity pattern "X" access the Web services described by *"B1"* and *"B2"* to take its inputs, process them and pushes its outputs into the Web service described by *"B3"* (consider that *"B1"*, *"B2"* and *"B3"* are published in server *"B"*). Then *"Y"* takes its data inputs from the Web services described by *"C1"*, *"C2"*, *C3* (all published in *"C"*) and *"B3"*, to generate the outputs pushed in the Web service described by *"A2"* (maintained and published by *"A"* itself).

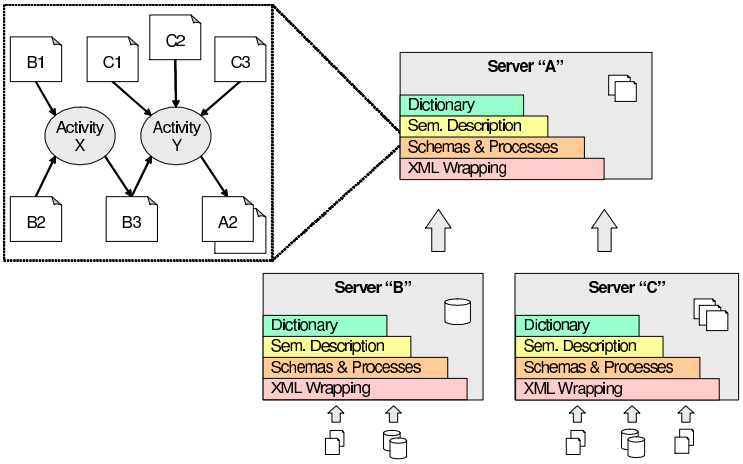


Fig. 17 The multi-tier distributed infrastructure for composition of Web services

5.3 Architecture

Figure 18 presents the architecture of a peer to peer site supporting POESIA services, outlining the communication with external sites and service brokers. The *Services Specification Tool* allows the domain expert to build solutions for particular needs. This tool supports browsing the resources available locally or remotely, in order to discover components to reuse. The descriptions and formal specifications of the local services are stored in the *Local Services* repository. One service may encapsulate one or more data sets. The *Local Data* repository maintains the data and metadata associated with local services. All the constituents of a service specification stored in the site are indexed by one ontology of the *Local Ontologies* repository. The *External Resources Locator* provides access to the descriptions of external resources. The *Catalog of External Resources* functions as a cache for the descriptions of external resources frequently accessed. Each local service and ontology can be published and used by external Web services.

The *Services Execution Engine* interprets the service specifications to properly manage the corresponding fragments of distributed processes. A service can be activated locally or by some external connection. A locally running service can also activate remote services to obtain its inputs or send its outputs. The *External Connections Manager* controls the communication with remote components and users at run time. It relies on the *External Resources Locator* to retrieve the descriptions of external resources whenever necessary. The thicker double arrows connecting the *Local Data* repository with the *Services Execution Engine*, and the latter with the *External Connections Manager*, which is linked to the *External Resources Gateway*, represent the data exchange between a local service and remote resources, during the execution of the distributed processes. A POESIA site also has two kinds of human-computer interfaces. The *User Interface* allows the do-

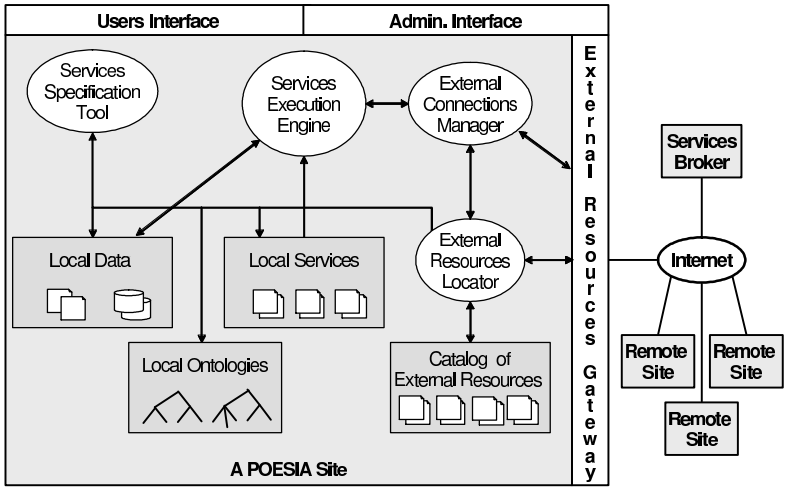


Fig. 18 The architecture of a POESIA-enabled peer to peer Web site

main experts to specify and activate services. The *Administration Interface* serves configuration purposes.

The architecture of a POESIA-enable Web site contemplates two types of external resources: *Remote Sites* and *Service Brokers*. Though it does not rule out connections with other kinds of resources. A *Remote Site* has the internal structure described for our POESIA site. *Service Brokers* are special sites that catalog the descriptions of the resources available across the Web to support the discovery and selection of resources.

6 Related Work

The Semantic Web [7,19] intends to extend the capabilities of the current Web to cope with problems such as finding precise information in the vast amount of resources available and supporting inter-institutional applications like electronic commerce. The means for achieving this are: standards for expressing machine-processable meta information (e.g., RDF, DAML+OIL); development and dissemination of terminologies using these standards (e.g., domain ontologies); and new tools and architectures based on this apparatus to build applications empowered with semantics and automated reasoning capabilities. POESIA relies on the infrastructure of the Semantic Web to implement some techniques, based on domain expertise, to organize, select and reuse data and services in the Web.

The POESIA approach to compose Web services through activity aggregation and specialization was inspired by the needs of our application domain, and founded by earlier work done in transactional activity modeling by Liu [14,13], where a set of mechanisms are proposed and formalized for specification and reuse of activities. Other research areas that are di-

rectly related to POESIA are the use of metadata and ontologies for Web services description, discovery and composition [1,5,17,4,15,3] and workflow techniques for scientific processes and Web services composition [11, 21,22]. Descriptions of the meaning, properties, capabilities and ontological relationships among Web services, expressed in languages like DAML services [1,5], support mechanisms to discover, select, activate, compose and monitor Web resources. Related work covers different aspects, ranging from theoretical studies to implementation efforts, from architecture issues to conceptual models [12,27].

Concretely, Paolucci *et al.* [17] show that the capabilities of registries such as UDDI and languages like WSDL are not enough to support services discovery. They employ DAML-S for this purpose and present an algorithm to match service requests with the profile of advertised services, based in the minimum distance between concepts in a taxonomy tree. Cardoso and Sheth [4], on the other hand, present metrics to select Web services for composing processes. These metrics take into account functional and operational features, such as the purpose of the services, quality of service (QoS) attributes and the resolution of structural and semantic conflicts. McIlraith *et al.* [15] use agent programming to define generic procedures involving the interoperation of Web services. These procedures, expressed in terms of concepts defined with DAML-S, do not specify concrete services to perform the tasks neither the exact way to use available services. Such procedures are instantiated by applying deduction in the context of a knowledge base, which includes properties of the agent, its user and the Web services. Finally, Bussler *et al.* [3] sketches an architecture for Web services attaining the Semantic Web aspirations.

The grounding of Web services involves several abstraction layers between the semantic specification and the implementation [20]. Nowadays, there is a myriad of proposals for specifying Web services composition in intermediate layers, such as WSFL (IBM), BPML (BPMI), XLANG (Microsoft), BPEL4WS (BEA, IBM, Microsoft), WSCI (BEL, Intalio, SAP, Sun), XPDL (WfMC), EDOC (OMG) and UML 2.0 (OMG). These proposals concern the orchestration of the execution of Web services in processes running across enterprise boundaries [21,2]. They build on top of standards like XML, SOAP, WSDL and UDDI, providing facilities to interoperate and synchronize the execution of Web services, which can use different data formats (e.g., heterogenous XML schemas) and communication protocols (HTTP, XMTP, etc.). Some challenges for these technologies are (i) reduce the amount of low level programming necessary for the interconnection of Web services (e.g., through declarative languages); (ii) provide flexibility to establish interactions among growing numbers of continuously changing Web services during run time; (iii) devise mechanisms for the decentralized and scalable control of cooperative processes running on the Web.

In order to illustrate the differences between our approach and Web services orchestrating languages, let us consider two of them: WSFL and BPML. The Web Services Flow Language (WSFL) [26] is an XML language

for the description of Web services compositions. WSFL considers two types of Web services compositions: Flow models specifies the appropriate usage pattern of a collection of Web services and how to choreograph the functionality provided by a collection of Web services to achieve a particular business need. Global models specify the interaction pattern of a collection of Web services, describing how a set of Web services interacts with each other. POESIA can be seen as a value-added method with an emphasis on using domain-specific ontologies to guide and facilitate the interaction among a set of Web services in terms of services utilization scopes.

BPML is specialized in supporting control flows of business process patterns. BPML and POESIA share the same objectives of supporting Web services composition. The main differences, however, lie in the mechanisms and methodology used in the underlying framework. BPML promotes the use of control constructs such as merge, split, multi-merge, exclusive choice, synchronization, and so forth to facilitate the composition of services, whereas POESIA combines the control logic with domain-specific ontologies, with emphasis on complex composition semantics at both data level and workflow activity level.

In summary, to the best of our knowledge, current proposals are mostly focused on business processes and there is a lack of research on supporting semantic consistency for Web services refinement and reuse. The POESIA approach contemplates the demands of some scientific applications. Furthermore, it addresses the semantic consistency issue, by using domain ontologies. POESIA complements the current technologies for Web services description, discovery and composition (including approaches based on ontologies for describing services, like DAML-S) in two aspects. First, it provides mechanisms to select Web services according to their utilization scopes (e.g., services intended for particular regions and classes of products). Second, it enables automated means to check if compositions of Web services are semantically correct with respect to these scopes (e.g., to determine if a Web service for calculating the water balance of lands covered with bushes can be properly incorporated in a process to determine land suitability for coffee).

7 Conclusions

Many scientific applications, including agro-environmental applications such as agricultural zoning are built by composing heterogeneous data sources and services. Large data sets are organized according to time and space dimensions, e.g., climate data rely on time series of weather data, and expected water content in soil is measured in spatial terms. Well-defined metadata precisely describing the meaning of these data sets are required for their correct composition. Agricultural zoning is an application built on scientific models (e.g., the matching of weather data with the plant model of growth and water requirements over time) and has very high economic impact.

For example, government agencies and financial institutions use agricultural zoning to make decisions on policies and loan approvals for farmers that want to plant specific crops.

In this paper, we introduced the POESIA approach to support the systematic composition of Web services. It is founded by domain ontologies, in which the properties of the semantic relationships between terms induces a partial order among the terms for each dimension of a reality (e.g., space, time, product). Current ontology engineering tools, such as Protégé and OntoEdit, can help to develop such ontologies. Using tuples of terms from these ontologies to express and correlate the utilization scopes of data and services, the POESIA activity model defines activity patterns that specify the Web services composition and the communication channels that link these services together.

POESIA complements current proposals for Web services' description, selection and composition, by using domain ontologies to (i) conceptually organize vast collections of services; (ii) uncover and select data and services according to their utilization scopes; (iii) check semantic and structural consistency properties of compositions of Web services. We illustrated the POESIA approach through a real application scenario: the agricultural zoning of *Coffea arabia* in the Center-South region of Brazil.

On top of this foundation, we are investigating further extensions of POESIA. Knowledge management and keeping track of data provenance in distributed processes can be more easily supported when Web services are built from well-defined ontologies and through well-defined operations based on activity patterns composition. Precise documentation of the data provenance will be useful in the evaluation of the quality and suitability of results for many applications. A richer set of semantic relationships can also be considered, in order to enhance POESIA capabilities for expressing and managing the utilization scopes of data and services. Another concern are aspects of the synchronization of Web services. These issues are being considered by several Web services orchestration languages (e.g., WSFL, BPEL4WS, XPDL). POESIA's strength is handling semantic aspects of Web services composition using domain ontologies. We are investigating extensions to its activity model to incorporate synchronization mechanisms using an existing proposal. On the one hand, our research will continue to be guided by real world application such as agricultural zoning. On the other hand, the generality and abstraction of POESIA approach makes it useful to many next generation Web service-based applications.

Acknowledgments

The first author is partially supported by Embrapa, CAPES and the Finep-/Pronex/IC/SAI95/97 project. The authors from Georgia Tech are partially supported by two grants from the Operating Systems and ITR programs (CISE/CCR division) of NSF, by a contract from the SciDAC program

of DoE, and a contract from the PCES program (IXO) of DARPA. All agriculture data used in this paper were provided by Brazilian experts. Thanks to the anonymous reviewers who contributed to improve this work.

References

1. A. Ankolekar, M. H. Burstein, J. R. Hobbs, O. Lassila, D. Martin, D. V. McDermott, S. A. McIlraith, S. Narayanan, M. Paolucci, T. R. Payne, and K. P. Sycara. DAML-S: Web service description for the semantic web. In *ISWC*, volume 2342 of *LNCS*, pages 348–363. Springer, 2002.
2. B. Benatallah, Q. Z. Sheng, and M. Dumas. The self-serv environment for web services composition. *IEEE Internet Computing*, 7(1):40–48, 2003.
3. A. Maedche C. Bussler, D. Fensel. A conceptual architecture for semantic web enabled web services. *SIGMOD Record*, 31(4), 2003.
4. J. Cardoso and A. Sheth. Semantic e-workflow composition. Report, LSDIS Lab, Computer Science Dep., Univ. of Georgia, 2002.
5. The DARPA Agent Markup Language (DAML). `http://www.daml.org/`.
6. P. H. Caramori et al. Climatic risk for coffee in Paraná state. *Brazilian Journal of Agrometeorology*, 9(3), 2001. (in Portuguese).
7. D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster (editors). *Spinning the Semantic Web*. MIT Press, 2003.
8. R. Fileto. *POESIA: An Ontological Approach for Data and Services Integration on the Web*. PhD thesis, Institute of Computing, University of Campinas, Brazil, 2003. (in preparation).
9. E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, Massachusets, 1995.
10. N. Guarino. Formal ontology and information systems. In *FOIS*, pages 3–15. IOS Press, 1998.
11. D. Hollingsworth. *The Workflow Reference Model*. Workflow Management Coalition, January 1995.
12. I. Horrocks and J. Hendler, editors. *Intl. Semantic Web Conf.(ISWC)*, number 2342 in LNCS, Sardinia, Italy, June 2002. Springer.
13. Ling Liu and R. Meersman. The building blocks for specifying communication behavior of complex objects: An activity-driven approach. *TODS*, 21(2):157–207, 1996.
14. Ling Liu and Calton Pu. A transactional activity model for organizing open-ended cooperative activities. In *HICSS*, 1998.
15. S. A. McIlraith, T. C. Son, and H. Zeng. Semantic web services. *IEEE Intelligent Systems*, 16(2):46–53, 2001.
16. C. B. Medeiros, G. Vossen, and M. Weske. WASA - a workflow-based architecture to support scientific database applications. In *DEXA*, volume 978 of *LNCS*, pages 574–583. Springer, 1995.
17. M. Paolucci, T. Kawamura, and K. P. Sycara T. R. Payne. Semantic matching of web services capabilities. In *ISWC*, volume 2342 of *LNCS*. Springer, 2002.
18. L. A. Rossetti. Agricultural zoning: Lessening the risks of agriculture and providing sustainable regional development. In *Intl. Symp. on Making Sustainable Regional Development Visible*, 2000.
19. W3C's Semantic Web Activity. `http://www.w3.org/2001/sw/`.

20. T. Sollazzo, S. Handschuh, S. Staab, and M. Frank. Semantic web service architecture – evolving web service standards toward the semantic web. In *FLAIRS Conf. – Special Track on Semantic Web*, 2002.
21. W. M. P. van der Aalst. Don't go with the flow: Web services composition standards exposed. *IEEE Intelligent Systems*, 18(1), 2003.
22. W. M. P. van der Aalst, A. Hirnschall, and H. M. W. Verbeek. An alternative way to analyze workflow graphs. In *Advanced Information Systems Engineering (CAiSE)*, volume 2348 of *LNCS*, pages 535–552. Springer, 2002.
23. The W3C web services activity. `http://www.w3.org/2002/ws/`.
24. P. Wohed, W. M. P. van der Aalst, M. Dumas, and A. H. M. ter Hofstede. Pattern based analysis of BPEL4WS. Report FIT-TR-2002-04, Queensland University of Technology, Queensland, Australia, 2002.
25. Web services description language (WSDL) 1.1. `http://www.w3.org/TR/wsdl`.
26. Web Services Flow Language (WSFL 1.0). `http://www.ibm.com/software/solutions/webservices/pdf/WSFL.pdf`.
27. N. Zhong, J. Liu, and Y. Yao (eds.). Special Issue – In Search of the Wisdom Web. *IEEE Computer*, 35, 2002.
28. Zoning coffee in Brazil. `http://orion.cpa.unicamp.br/cafe` (in Portuguese).