Towards Interactive Composition of Semantic Web Services

Jihie Kim and Yolanda Gil
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292, U.S.A.
jihie@isi.edu, gil@isi.edu

Abstract

We are developing a framework for interactive composition of services that assists users in sketching their task requirements by analyzing the semantic description of the services. We describe the requirements that an interactive framework poses to the representation of the services, and how the representations are exploited to support the interaction. We also describe an analysis tool that takes a sketch of a composition of services and generates error messages and suggestions to users to help them complete a correctly formulated composition of services.

1 Introduction

Existing approaches to generate compositions automatically are limited in their use when explicit goal descriptions are not available and when users want to drive the composition process, influencing the selection of components and their configuration. The goal of our work is to develop interactive tools for composing web services where users sketch a composition of services and system assists the users by providing intelligent suggestions.

Interactive service composition poses additional challenges to composing services. Users may make mistakes and the system needs to help fix them. Also, user's input is often incomplete and may even be inconsistent with existing service descriptions. In order to help users in this context, we have developed a framework for providing strong user guidance by reasoning on the constraints associated with services. The framework is inspired by our earlier work in KANAL to help users construct process models from pre-defined components that represent objects and events [Kim and Gil, 2001]. In our previous work, we have built a tool that performs verification and validation of user entered process models by exploiting domain ontologies and event ontologies. In this work, we take simple service descriptions (in WSDL) and augment them with domain ontologies and task ontologies that address various constraints in the domain. Our analysis tool then use these ontologies in examining user's solutions (i.e., composition of services) and generating error messages and suggestions to correct the errors. We believe that as ontologies become richer, the tool can provide more direct and focused suggestions.

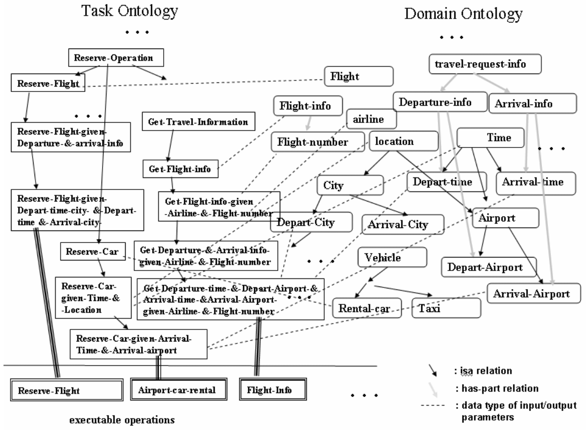

Task Ontology
Domain Ontology
Reserve-Operation
Reserve-Flight
Reserve-Flight-given-Departure-&-arrival-info
Reserve-Flight-given-Depart-time-city- &-Depart-time &-Arrival-city
Reserve-Car
Reserve-Car-given-Time-&-Location
Reserve-Car-given-Arrival-Time-&-Arrival-airport
Get-Travel-Information
Get-Flight-info
Get-Flight-info-given-Airline-&-Flight-number
Get-Departure-&-Arrival-info-given-Airline-&-Flight-number
Get-Departure-time-&-Depart-Airport-&-Arrival-time-&Arrival-Airport-given-Airline-&-Flight-number
Flight
Flight-info
Flight-number
airline
location
travel-request-info
Departure-info
Arrival-info
Time
City
Depart-City
Arrival-City
Depart-time
Arrival-time
Airport
Depart-Airport
Arrival-Airport
Vehicle
Rental-car
Taxi
Reserve-Flight
Airport-car-rental
Flight-Info
executable operations
: isa relation
: has-part relation
: data type of input/output parameters

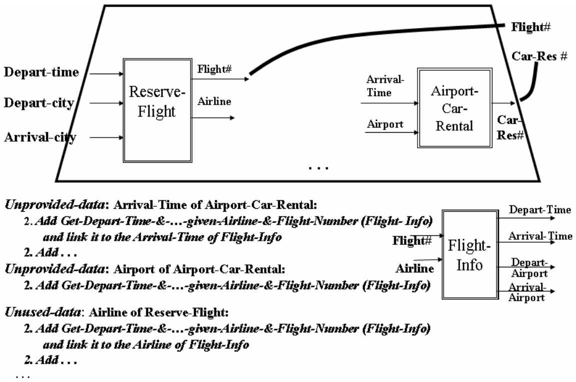
Figure 1: Task Ontology and Domain Ontology.

2 Approach

Our approach is to provide strong user guidance through constraint reasoning, as described above. First we take definitions of services and analyze relations between service operations in the composition sketch based on their input and output parameters. We then detect gaps and errors from the analysis including missing steps, missing connections, incomplete steps, etc. Finally we produce suggestions based on the problem type and context. In performing the analysis, we assume a knowledge rich environment where services and their operations are described and related in terms of domain objects. (We are investigating some ways to exploit existing ontologies that are available on-line.) Currently we are exploiting two types of ontologies: domain term ontology and task ontology. That is, data types are represented using domain objects, and their task types are defined in terms of their input and output data types. Figure 1 shows such ontologies that we are using in a travel planning domain. For example, a task type Reserve-Car-given-Arrival-Time-&-Arrival-Airport represents a service operation that has Arrival-Time and Arrival-Aiport as the input and Flight-Info as the output. Its parent Reserve-Car-given-Time-&-Location represents a more general class of operations including Reserve-Car-given-Arrival-Time-&-Arrival-Airport. Note that because the system has an ontology of operation types that describes high-level task types as well as specific operations that are mapped to actual operations, users can start from a high-level description of what they want without knowing the details of what operations are available. We often find that users have only partial description of what they want initially, and our tool can help users find appropriate service operations by starting with a high-level operation type and then specializing it.

The tools we built is called CAT (Composition Analysis Tool). CAT's analysis is driven by a set of desirable properties of composed services. Given a sketch of a service composition and a user task description (i.e., a set of initial input and expected results), CAT checks if (1) all the expected results are produced, (2) all the links are consistent, (3) all the input data needed are provided, and (4) all the operations are executable (there are actual operations that can be executed). In addition, it generates warnings on (5) unused data and (6) unused operations that don't participate in producing expected results. Given any errors detected, CAT generates a set of specific fixes that can be potentially used by the user. The following shows the general algorithms.

- **Checking Unachieved Expected Results**:
  Detect problem: for each expected result, check if it is linked to an output of an operation or directly linked to any of the initial input (i.e., the result is given initially).
  Help user fix problem:
  1. find any available data (initial input or output from introduced operations) that is subsumed by the data type of the desired result, and suggest to add a link
  2. find most general operation types where an output is subsumed by the data type of the desired result, and suggest to add the operation types.
- **Checking Unprovided Data**:
  Detect problem: for each operation introduced, for each input parameter of the operation, find if it is linked to any (either to the initial input or to some output from introduced operations).
  Help user fix problem:
  1. find any initial input data or output of operations that is subsumed by the desired data type, and suggest to add a link.
  2. find most general operation types where an output is subsumed by the desired data type, and suggest to add the operation types.
- **Checking Inconsistent Links**:
  Detect problem: for each link between data types, find if the type of the data provider is subsumed by the type of the consumer.
  Help user fix problem:
  1. find most general operation types where an output is subsumed by the type of the consumer and an input subsumes the the type of the provider, and suggest to add the operation types.
- **Checking Unexecutable Operation**:
  Detect problem: for each operation type introduced, check if there is an actual operation of that type that can be performed.
  Help user fix problem:
  1. find a set of qualifiers that can be used to specialize it and suggest to replace the operation type with a more special one base on the qualifiers.
  2. find the subconcepts of the task type in the task ontology and suggest to choose one of them.
- **Checking Unused Data**:
  Detect problem: for each initial input data type and the output from the introduced operations, check if it is linked to an operation or an expected result.
  Help user fix problem:
  1. find any unprovided data or unachieved results that subsumes the unused data type, and suggest to add a link.
  2. find most general operation types where an input subsumes the unused data, and suggest to add the operation types.
- **Checking Unused Operation**:
  Detect problem: for each operation introduced, check if its output or any output from its following operations is linked to an expected result.
  Help user fix problem:
  1. suggest to add a link to connect the operation



Unprovided-data: **Arrival-Time of Airport-Car-Rental:**
2. ***Add Get-Depart-Time-&-…-given-Airline-&-Flight-Number (Flight- Info) and link it to the Arrival-Time of Flight-Info***
2. ***Add . . .***
Unprovided-data: **Airport of Airport-Car-Rental:**
2. ***Add Get-Depart-Time-&-…-given-Airline-&-Flight-Number (Flight-Info)***
Unused-data: **Airline of Reserve-Flight:**
2. ***Add Get-Depart-Time-&-…-given-Airline-&-Flight-Number (Flight-Info) and link it to the Airline of Flight-Info***
2. ***Add . . .***
. . .

Figure 2: Travel Planning: CAT finds errors and help users fix them.

Figure 2 shows a process of composing services for a travel planning. The user wants to reserve a flight first and then reserve a car based on the reserved flight. Currently two input parameters of Reserve-Car operation, Arrival-Time and Airport, are not linked yet. CAT points that both of them can be potentially linked if the Flight-Info operation is added in between, since it produces data on Arrival-Time and Airport (Depart-Airport and Arrival-Airport) given an Airline and a Flight-number. This addition will also resolve the warning of unused data (Airline of Reserve-Flight). In this case, as the system has richer ontology of trips so that the airport of the Airport-Car-Rental actually means the Arrival-Airport, then the suggestions will become even more specific.

3 Current Status

The current implementation of CAT has a text-based interface for reporting errors and suggestions. We have applied CAT in composing computational pathways to put together end-to-end simulations for earthquake scientists where the problem is to analyze the potential level of hazard at a given site. The preliminary tests show that CAT can help users formulate correctly formulated pathways by pointing specific ways to fix errors. Our plans for future work include development of graphical user interfaces for CAT, dynamic generation of task ontologies from service descriptions, and incorporation of automatic service composition approaches.

References

[Kim and Gil, 2001] Jihie Kim and Yolanda Gil *Knowledge Analysis on Process Models*. Proceedings of IJCAI-2001.