Analyzing Service-Oriented Systems Using Their Data and Structure

Dragan Ivanović  
Universidad Politécnica de Madrid (UPM)  
Boadilla del Monte, Madrid, Spain  
idragan@clip.dia.fi.upm.es

Manuel Carro and Manuel Hermenegildo  
UPM and IMDEA Software Institute  
Boadilla del Monte, Madrid, Spain  
manuel.carro@{imdea.org,upm.es}  
manuel.hermenegildo@{imdea.org,upm.es}

Abstract—We describe our approaches to enhance consistency validation and QoS prediction in service-oriented systems by making use of data properties and the structure of workflows and service compositions.

I. CONSISTENCY IN SERVICE ORCHESTRATIONS

Real workflows can be deployed across organizations and it is then paramount to have clear rules regarding which information reaches which activity (due to, for example, privacy concerns): a health-related workflow, where patient data is retrieved from a database or even bundled in a physical folder, may need data tagged as “medical” to be accessible to the doctor and require administrative data not to be accessible to the doctor.

That situation can be described by attaching properties to data and activities, but atomic tags are not rich enough for many situations: for example, they cannot automatically describe data items which are generated by grouping together other data items. Organizing properties in a (finite) lattice (as in formal concept analysis [1]) enhances their descriptiveness by making it possible to capture properties which are the lowest upper bound and greatest lower bound of other properties.

By marking input data items with their properties, describing how activities examine and reassemble data, and which properties have the data generated by activities, it is possible to infer safe approximations of the properties of data reaching every activity by analyzing the workflow. In our approach (Figure 1) we use sharing analysis in a logic program [2] which captures the relevant characteristics of the workflow. The result of the analysis is an assignment of activities and data items to lattice points (and hence to properties) which is a safe approximation of actual behavior.

This assignment can be compared with the actual workflow to ensure that activities receive data with the required properties and that no data is delivered to activities with insufficient privileges to read it. This is obviously useful to suggest and verify, e.g., fragmentation policies [3], [4]. We note that this analysis is non-trivial in the presence of loops and sophisticated data handling, which we treat.

II. PREDICTION OF SLA VIOLATIONS

Statistical and probabilistic principles have been commonly used to predict QoS. However, they usually do not take into account the actual data sent to service-oriented systems. This is depicted in Figure 2, top left: a single point, surrounded by an area of (increasing) uncertainty, is used to predict future behavior. This is unsatisfactory but can be improved by making the prediction depend on the data (top right). However, this cannot be used to ensure that

Figure 1. Minimal assignment of descriptive concepts.

Figure 2. Combination of data awareness and safe bounds.
the QoS stays within certain limits (upper and lower bounds – bottom), which can increase prediction precision [5].

In order to improve this situation, we take into account both the structure of the orchestration, including loops, and the (expected) upper and lower bounds of QoS of activities, including invocations to external services. With this information, we generate dynamically a system of constraints (Figure 3) which expresses the possible upper and lower bounds according to the structure of the process [6] and the known QoS limits of the activities. As an example, the constraints for the structure marked 1 (an and-split) include bounds for the activities and for the structure itself: the components may run in parallel or, if only one thread or one CPU are available, sequentially.

The constraints for the different structures are recursively combined upwards to give the total QoS, $T$. By constraining $T$ to meet or not the SLA conditions ($T \leq T_{\text{limit}}$ and $T > T_{\text{limit}}$, resp.) and solving the equation system under these boundaries, we derive conditions for the intermediate $T_{ai}$ under the assumption of success or failure. These conditions are compared at runtime with the actual value for $T_{ai}$ to decide whether violations can be expected to happen or not. Additionally, and since constraints are generated dynamically, we get progressively simpler and more accurate constraints as the execution proceeds. Besides, we also use techniques derived from automatic cost/complexity analysis to derive bounds to the number of loop iterations ($k$, in our case) as functions of the input data, which greatly improves prediction accuracy.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Community’s Seventh Framework Pro-

gramme under the Network of Excellence S-Cube (Grant Agreement no 215483). The authors were also partially supported by Spanish MEC project 2008-05624/1TN DOVES and CM project P2009/TIC1465 PROMETIDOS.

REFERENCES


