Haley: An End-to-End, Scalable Web Service Composition Tool

Haibo Zhao
LSDIS Lab, Dept. of Computer Science
University of Georgia
Athens, GA, 30602
zhao@cs.uga.edu

Prashant Doshi
LSDIS Lab, Dept. of Computer Science
University of Georgia
Athens, GA, 30602
pdoshi@cs.uga.edu

1. MOTIVATION AND SIGNIFICANCE

A major benefit of service oriented architectures (SOAs) is the potential to automatically compose Web services into business processes thereby promoting flexibility and reusability. The business process execution language for Web services (WSBPEL) provides a standard way to represent the Web service compositions. However, manually designing BPEL processes is cumbersome and tedious. While visual BPEL editing tools such as the ActiveBPEL editor provide some ease, it quickly becomes impractical as the number of Web services increases.

In this regard, automated service composition using machine understandable and semantically-annotated service descriptions is promising. Existing composition approaches utilize AI planning but suffer from the following limitations: (1) Tools such as Astro and SHOP2 do not operate directly on Web service descriptions in WSDL; (2) many techniques do not scale due to the inherent computational complexity of their planning algorithms; (3) both Astro and SHOP2 assume that services are determinstic and fail to capture the uncertainty; and (4) most approaches primarily focus on functional satisfiability and do not account for key non-functional requirements such as cost, response times and reliability of the generated processes. Consequently, few comprehensive tools exist for automatically composing Web services. A notable freely available tool is the Astro suite, which allows for WSBPEL based automated business process composition. However, it requires that the Web services be described using abstract BPEL – a surprising and unintuitive choice for describing services. In particular, the design of the abstract BPEL itself is time-consuming and complicated.

Our tool suite, Haley[1] is an end-to-end scalable solution for automating Web service composition. Haley accepts Web service descriptions in SA-WSDL, which is a new W3C recommended language for annotating and describing Web services. For this, we provide an Eclipse plug-in to view and edit the SA-WSDL descriptions of the component services. Haley promotes large compositions by exploiting the hierarchical decomposition that often exists naturally in many real-world processes. Additionally, Haley offers a way to mitigate the problem of large state spaces by composing at the logic level and provides the expressive power of first order logic.

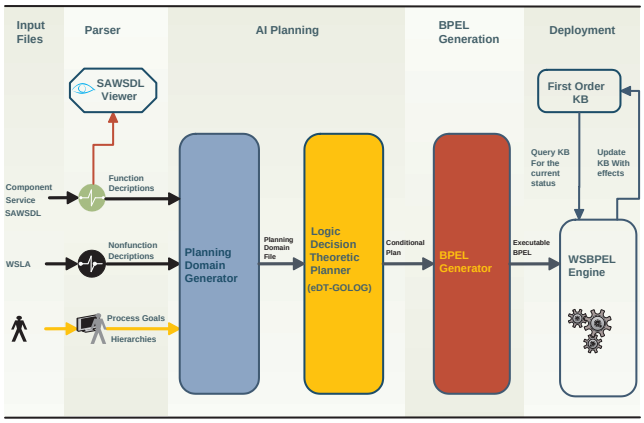

Figure 1: Architectural details of Haley.

At its core, Haley utilizes a first order decision theoretic planner (semi-Markov decision process) to compose the services. The planner is stochastic thereby explicitly modeling the uncertain behaviors of Web services and provides guarantees of expected optimality with respect to cost and other QoS metrics. Haley produces a composition specified using standard WSBPEL, which may be executed in any BPEL implementation.

2. ARCHITECTURAL DETAILS

Haley is an end-to-end technology solution for Web service composition providing an Eclipse plug-in for viewing input service descriptions in SA-WSDL to automatically generating an executable BPEL composition. We show the architecture in Fig. 1 and describe the main components in the tool suite below:

•**SA-WSDL Parser and Viewer** SA-WSDL (semantic annotations for WSDL) extends WSDL by allowing semantic annotations in the form of model references and schema mappings. In addition to specifying the inputs and outputs of a service, SA-WSDL also allows the specification of preconditions and effects, which are useful for composing services. However, the current SA-WSDL specification does not ground preconditions and effects using any language. We therefore extend SA-WSDL to support preconditions and effects specified using SWRL, a popular semantic Web rule language.

[1]Haibo Zhao and Prashant Doshi, "Haley: A Hierarchical Framework for Logical Composition of Web Services", in Intl. Conf. on Web Services (ICWS), 2007, pp.312-319

WWW2008, April 21–25, 2008, Beijing, China.

In addition to parsing SA-WSDL files using the SAWSDL4J API, Haley provides a new Eclipse plug-in for graphically viewing SA-WSDL based services descriptions. We show a snapshot of the viewer in Fig. 2.

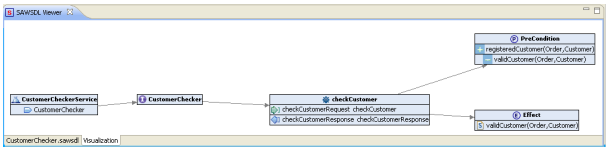

Figure 2: Eclipse plug-in for viewing SA-WSDL files.

•**WSLA Parser** Web service level agreements (WSLA) specify nonfunctional quality of service (QoS) parameters of Web services such as response times, costs and availability percentages. Haley is not limited to any particular service agreement specification and can be easily extended to support WS-Agreement or other agreement specifications. QoS considerations are often neglected while manually designing BPEL processes as well as by many other automated composition techniques. Haley uses a decision-theoretic planner for the composition that provides an intuitive way to model the QoS parameters and optimize them.

•**Process Hierarchy Modeler** Service composition methods often do not scale well to many services, which makes it difficult to use them in real-world applications. Haley promotes scalability by exploiting the hierarchies usually found in real-world processes. To facilitate this, Haley provides an intuitive GUI (see Fig. 3) to construct process hierarchies by importing component Web services at each level. The modeler may also be used to specify the start states, multiple goals and associated priorities at each level of the hierarchy. All of this information is written into an XML file for input to the planner.

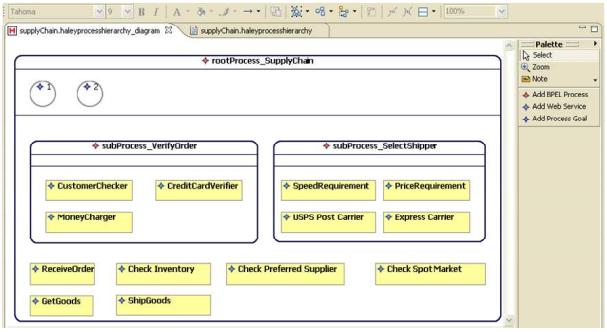

Figure 3: GUI for intuitively formulating the process hierarchy.

•**Planning Domain Generator** Given the functional and nonfunctional descriptions of individual Web services and goal descriptions for the target process, we automatically generate a corresponding planning domain file. The planning domain contains a first order logical description of the operations and their inputs, outputs, preconditions and effects as well as the goals.

•**eDT-GOLOG Planner** Haley uses a first-order Prolog based planner that has been extended to make it applicable to situation calculus and decision theory. In addition to being expressive, eDT-GOLOG allows us to model the uncertainty of Web service operations, QoS measures and provide guarantees of optimality while preserving speed of planning. We have empirically evaluated the performance of eDT-GOLOG in comparison with two other well-known WS composition techniques – HTNs augmented with information gathering actions and MBP (used in the Astro project). Our results have shown the improved performance in terms of optimality and speed obtained by the planner. The planner takes as input the planning domain file and produces a policy or a conditional plan for the composition.

•**BPEL Generator** Haley transforms the conditional plan output by the planner into an executable WSBPEL file. Manual BPEL process design requires designers to specify name spaces, variables, partner links, and the control flow of the activities. Haley programmatically generate an executable BPEL and then deploys it in an ActiveBPEL engine. This saves time and effort, and avoids common grammatical and logical errors while designing BPEL processes.

• **KB based Process Monitor** In order to determine which branches to take while executing the BPEL, service operations once performed are asserted in a first order knowledge base (KB). The KB is implemented using an embedded Prolog engine wrapped in a Web service. The KB is updated with the effects of the operations and queried for the next operation to perform.

In summary, Haley automatically composes an executable WSBPEL process given component services described using SA-WSDL and service agreement files, using a first-order logic based planner that is both scalable and expressive.

3. IMPLEMENTATION DETAILS

Haley is implemented as a suite of freely-available Eclipse plug-ins and a stand-alone Eclipse RCP application. It is provided under the EPL license v1.0. Some of the technologies used in developing Haley are Draw2d, EMF, GMF, Prolog and ActiveBPEL API. Haley also contributes several independent tools and experiences to the SOA community: (1) SA-WSDL viewer is a complete and independent Eclipse plug-in and is the first viewer for SA-WSDL files. (2) eDT-GOLOG may be used as a powerful stand-alone decision-theoretic planner. The system is compliant to the Eclipse plug-in standard and can be integrated with other Eclipse-based tools like WTP, WSDL editor, ActiveBPEL Simulator and ActiveBPEL designer in case process designers want to customize the generated BPEL code.

4. DISCUSSION

We believe that automated Web service composition is both promising and practical, but few tools exist in support due to inherent scalability problems. Haley provides a state of the art, end-to-end and scalable solution for Web service composition, and an interface that hides the complexity of AI planning and BPEL from process designers. Haley offers unique advantages over manual BPEL process design and other automated approaches to composition. Development of Haley represents a significant milestone in the research on Web service composition, and we are excited to bring a significant contribution to application development in SOA.

We plan to integrate service discovery with the composition framework, and test Haley with additional real-world large scale scenarios. We will also continue to improve the usability and reliability of Haley.