Tailor: Interactively Acquiring and Modifying Procedures for a Plan Execution System

Jim Blythe
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292
blythe@isi.edu

Overview

The Calo project is designing an office assistant intended for a wide range of users, using a PRS-like tool called SPARK to track the various tasks the user may need to perform, e.g. setting and attending meetings or arranging trips. The office domain has a rich set of tasks associated with it, and the Calo system will learn from the user in a variety of ways. One important way is to learn from instruction: to be able to modify its behavior based on one or two sentences from the user that describe the required change. We are developing the Tailor system to interpret a user sentence about a desired behavior change based on an understanding of the tasks, a model of possible changes, a constructive search process to interpret the user's sentence and an explicit analysis to check that the proposed change leads to a correct, working system. This demo shows our initial work on providing this capability.

We illustrate Tailor on an example task of purchasing a laptop within an organization. This requires finding laptops that meet the user's specifications, choosing a model, submitting an application to purchase the laptop, receiving authorization, and finally placing the order and overseeing the purchase. A set of PRS procedures to handle and track these tasks was designed independently from our group. Although they were designed to be general and include several alternative procedures that can be chosen using advice or through state variables, inevitably there are cases where the procedures themselves need to be modified to capture the preferences of a particular user or group.

Users face a number of hurdles when trying to modify procedure knowledge directly. It can be hard to choose a procedure or set of procedures to modify to produce some desired change in a large system. Making the modification requires understanding the syntax of the task language and the meaning of the action terms and state predicates used in the system. The consequences of a local change may be far-reaching and hard to predict in all cases.

Tailor addresses these problems in some cases. It allows a user to enter a short sentence describing a desired change in the system's behavior and combines several techniques to interpret the sentence as a modification to SPARK's task KB. First, Tailor makes a global analysis of the how the procedures are linked in order the solve a problem, including the information they use, to understand which procedures can be modified to create some effect and what consequences this might have on other tasks. The analysis creates a graph structure called the global procedure analysis (GPA) that tracks information use and maintains type information about variables. Second, a search is made based on both tasks and the relations in SPARK's knowledge base to find candidate procedures to be changed and conditions based on the user's sentence. This builds on earlier work in search-based annotations applied to purely declarative domains [Blythe 2001; Blythe and Gil 2004]. Once the user agrees to a modification, the new procedures are sent to SPARK and can change its behavior even on tasks that are already under way.

As an example, suppose the user finds that Calo is asking for authorization to make a laptop purchase, when it is not necessary. Perhaps in this organization, authorization is only required when the laptop cost is above a certain threshold. If such a test had been built in to SPARK's procedure, the user could set a variable to change the threshold. However, the current procedure has no such test and always makes a request for authorization. The user issues Tailor the sentence "You don't need authorization when the cost is less than $2000". Tailor should change the procedure that issues this request so that it depends on the appropriate condition. Tailor avoids showing code when communicating with the user by translating from the SPARK code into an English format to describe the current procedure and the suggested changes, as shown below.

Global procedure analysis

When the user gives advice to the system to produce a change to the procedure knowledge base such as a new or modified procedure, follow-on changes may be needed to produce a coherent set of procedures that have the effect the user intended. For example, if the user relaxes the preconditions for a procedure, this may have no effect unless the preconditions of the sub-procedures are also relaxed. The tool analyzes the global effects of a change to ensure that the new set of procedures can be combined to solve the goal and to suggest remedies if this is not the case. This helps the user define a coherent set of modifications to the task KB.

The current version builds a partial GPA to compare the information use in the planner before and after the user gives advice. Tailor suggests options to the user to help repair potential problems. For example, it warns the user and suggests fixes if a task is made optional but produces information required by a task in a subsequent procedure that has not been made optional. We are currently extending the cases tested by the GPA.

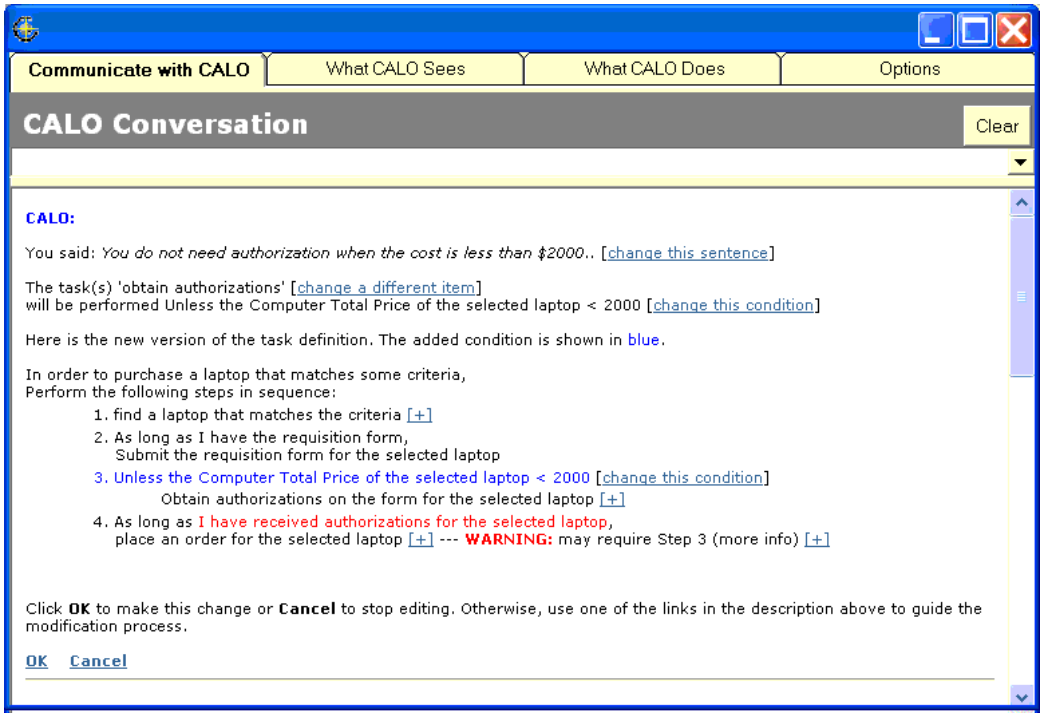
Operationalizing the user's advice

The user describes a desired change in Calo's behavior in their own terms, not those in the Calo ontology or conforming to the structure of the procedure knowledge. First, Tailor must operationalize the user's advice into specific modifications to procedures and choose the modification that best captures the user's intention. This depends on the current active tasks as well as existing advice. Second, Tailor must map the user's description of tasks and conditions into its formal representation based on the situation and its currently active tasks.

The constructive search approach currently finds mappings for conditions that are added to a procedure definition. It searches for compound terms from the ontology, allowing for synonyms and using the partial GPA as context. For example, if the user says "don't buy if the screen is less than 15 inches", Tailor reasons that the user could be referring to a computer, that computers have displays that are also called 'screens' and that displays have physical sizes that can be compared in order to map the condition to a valid expression from Calo's ontology, in this case

(< (Unit_Size (Computer_Display $selection)) (length_value 15 inch))

This requires not only knowledge of the predicates and concepts in the knowledge base but also the analysis of the global set of procedures provided by the GPA, for example to know that $selection will be bound to a computer in the procedure that is being modified.



. We are currently extending the scope of the mapping to cover advice that refers to tasks as well as factual knowledge (for example, "make the purchase through the business office unless the cost is below $5000"). When several inferences are combined, as in this approach, there is often more than one possible match, and Tailor should ask more detail from the user in order to make the right change to the KB. Currently, it indicates that there are several potential matches and allows the user to choose.

Our initial work shows that some modifications can be made by users with Tailor that would otherwise be out of reach. It may not be possible to handle all modifications in this way, because many assumptions and pieces of background or common sense information are left unstated in a SPARK KB, as in any body of code. However Tailor shows how explicit reasoning about the task knowledge can significantly broaden the applicability of a fielded reactive planning system such as SPARK.

References

Blythe, J. Integrating Expectations to Support End Users to Acquire Procedural Knowledge, IJCAI 2001

Blythe, J. and Gil, Y. Incremental Formalization of Document Annotations through Ontology-Based Paraphrasing, WWW 2004