Agent Wizard: Building Information Agents by Answering Questions

Rattapoom Tuchinda and Craig A. Knoblock
Information Sciences Institute
University of Southern California
Marina Del Rey, CA 90292 USA
+1 310 448 8786
{pipet, knoblock}@isi.edu

ABSTRACT
We present a question-answering approach where a user without any programming skills can build information agents by simply answering a series of questions. These resulting agents can perform fairly complex tasks that involve retrieving, filtering, integrating and monitoring data from online sources. We evaluated our approach to building agents, which is implemented in a system called the Agent Wizard, by re-implementing a set of agents for monitoring travel that originally took four programmers roughly four days to implement. Using the Agent Wizard, the entire set of agents can be implemented in under 35 minutes.

Categories and Subject Descriptors
H.5.2 [**User Interfaces**]: *Graphical user interfaces (GUI), Interaction styles*

General Terms: Algorithms, Design, Human Factors

Keywords
Information Integration, Information Agent, User Interface

1. INTRODUCTION
As more information becomes available and more business transactions are performed on the Internet, users often need to compare and monitor information from multiple web sources. Unfortunately, users are stuck with the interface provided by the web sites. For users with no programming experience, the option to view integrated data and monitor information is available on a very limited set of websites.

Our paper introduces the concept of guiding users with no programming experience to build an information agent by answering a set of questions. The original idea of our work comes from commercial tax preparation software, such as TurboTax, where users "build" complicated tax forms by answering a list of questions. To create a tax form, tax software guides users through a set of questions. As the user provides answers, the information in the answer is propagating through workflows that compute taxes based on choices and information given by the user.

2. MOTIVATING EXAMPLE
We consider the example of purchasing an airplane ticket online. A user wants to build an information agent (we will name it the PriceMonitor Agent) that monitors the price of the flights that depart from LAX on 10-10-03 to BOS, and return on 10-13-03. Assume that the user usually checks prices on Orbitz and Expedia, so the agent should monitor prices from both sources every three hours over the period of 10 days. Also, the user wants to view only flights that belong to the airline that he has frequent mileage program with; let assume that it is United Airlines. If the price for a specific flight changes, or if new flights are available, the user wants the agent to notify him by email with a list of updates. Assuming that we already have agents that extract information from Orbitz and Expedia (i.e., flightnumber, airlines, price), Figure 1 shows the workflow of our new PriceMonitor agent.

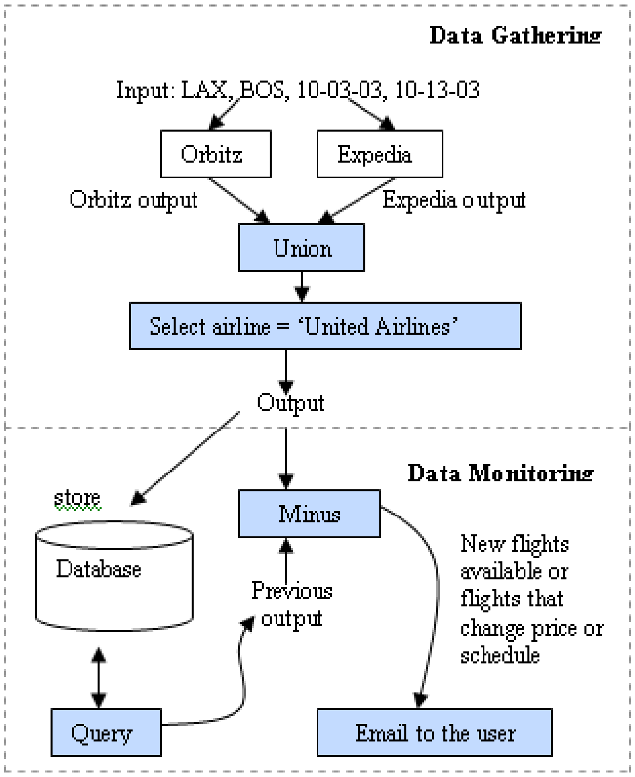


Figure 1: The workflow for our PriceMonitor Agent

3. BUILDING INFORMATION AGENTS BY ANSWERING QUESTIONS

The goal of the question-answering approach is to create the workflow in Figure 1, which can later be converted into an executable agent, by asking a user some simple questions. The challenge is to decide what questions to ask and the order of the questions that we should ask the user. Our approach is to impose a hierarchical structure on the web sources in a form of a tree as shown in Figure 2. The lowest level is the agent level where we have agents that can extract the data from the web site. Domain level and service level are abstract levels that we introduce so we can group agents based on domains and services. The output level can be mapped to the output node in Figure1.

Based on this structure, we can derive the set of questions to ask on each node based on the level of the tree. To determine the order of the questions, we use the post order traversal of the node in the tree.

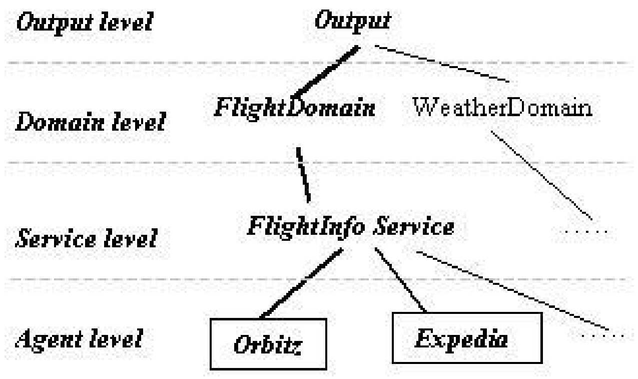


Figure 2: The hierarchical organization of the web sources

3.1 Building Agents

We have implemented the Agent Wizard, the web interface that allows users to build information agents by answering questions. Figure 3 through Figure 6 show the screen shot of the questions that the Agent Wizard presents to the user. To build agents, we first ask the user to select domain(s) (shown in Figure 3), service(s), and agent(s) that the user wants to work on. Based on these answers, we can build a tree similar to Figure 2, but this tree will only include selected domain (FlightDomain), service (FlightInfo Service), and agents (Orbitz and Expedia). This tree captures the structure of web sources for agents that the user wants to use to build a new agent. We translate this tree into an intermediate workflow using post-order traversal. We will traverse these nodes in the following order: Orbitz, Expedia, FlightInfo Service, FlightDomain, and Output. Based on the level of the node as shown in Figure 2, different questions will be asked.

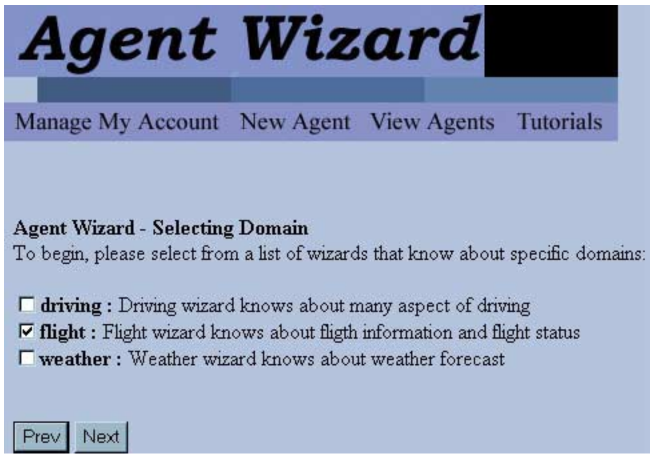
Agent Level (Orbitz, Expedia): the Agent Wizard will ask the user to specify the required input for each of the agent.

Service Level (FlightInfo): when a node in the service level is reached, all of its children have already been given input by the user. In this level, the Agent Wizard will ask users to specify how to combine the result from each of its children (Figure 4).

Domain Level (FlightDomain): in this level, the Agent Wizard will ask the user to integrate the result from each service. Since in our example, we only select one service, no question will be asked.

Output Level (Output): in this level, the user will have a chance to filter the data, so that only flights from United Airlines will be selected (Figure 5).

As the user answers each question, the workflow in the data gathering part in Figure 1 will be generated incrementally. By the time we reach the output node of the post order traversal, the workflow in the data gathering part will be completed. For the data monitoring part, the user will be asked to select one of the seven available monitoring conditions (Figure 6). Based on the user choice, the workflow in the data monitoring part will be generated automatically, so the user will be sheltered from details of the query and the configuration of the database.

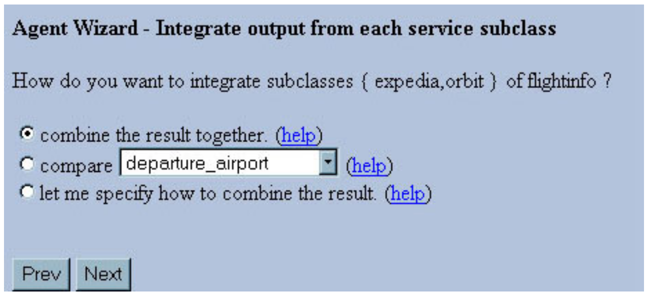


Agent Wizard - Selecting Domain

To begin, please select from a list of wizards that know about specific domains:

- ☐ **driving :** Driving wizard knows about many aspect of driving
- ☑ **flight :** Flight wizard knows about fligth information and flight status
- ☐ **weather :** Weather wizard knows about weather forecast

Figure 3: Selecting domain(s)

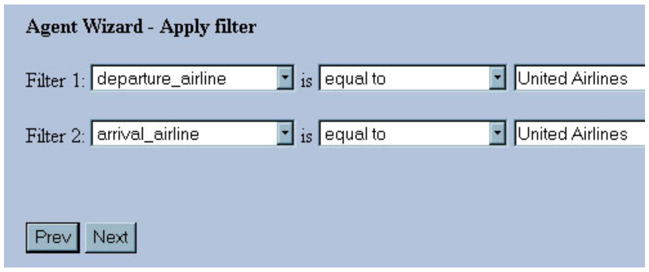


Agent Wizard - Integrate output from each service subclass

How do you want to integrate subclasses { expedia,orbit } of flightinfo ?

- [✓ combine the result together.] (help)
- compare departure_airport (help)
- let me specify how to combine the result. (help)

Figure 4: Combine the result from Orbitz and Expedia

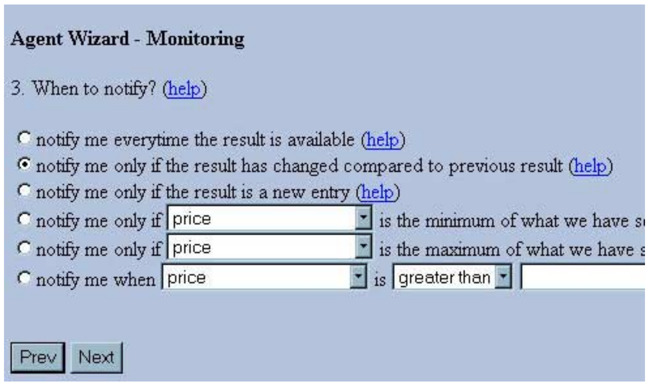


Agent Wizard - Apply filter

Filter 1: departure_airline is equal to United Airlines

Filter 2: arrival_airline is equal to United Airlines

Figure 5: Specifying filters


Agent Wizard - Monitoring

3. When to notify? (help)

( ) notify me everytime the result is available (help)
(•) notify me only if the result has changed compared to previous result (help)
( ) notify me only if the result is a new entry (help)
( ) notify me only if price is the minimum of what we have s
( ) notify me only if price is the maximum of what we have s
( ) notify me when price is greater than

Figure 6: Specifying a condition that will prompt the notification.

4. EVALUATION

We evaluate how well the Agent Wizard works by using it to build a set of agents in the flight travel domain based on the Travel Elves [1]. The Travel Elves is an application suite that let users search and monitor for information about flights that most air travelers find useful. The Travel Elves contains nine agents and it took four programmers roughly four days to implement the whole suite. To evaluate the Agent Wizard, we had two users build nine agents using the Agent Wizard that are functionally equivalent to the nine agents in Travel Elves. The first user is an expert user who knows the Agent Wizard well. The second user is one of the four programmers who implemented the Travel Elves. Table 1 shows the experimental results. Using the Agent Wizard, the entire set of agents can be implemented in under 35 minutes.

	Total time	Average time per agent	Total questions answered	Average questions answered per agent
Expert	27:00	3:00	164	18
Programmer	33:52	3:45	152	17

Table 1. Experimental results

5. RELATED WORK

Many research projects also address the problem of letting users who have no programming experience create complicated programs or agents. Programming by demonstration [2, 4] use machine learning to help constructing agents, web pages, or teach robots to perform tasks from users' actions. By learning from user demonstrations, users have to understand what they are trying to do, and the systems that implemented the approach have to understand what users want based on an observation. In our approach, users do not need to fully understand how to perform the tasks because the Agent Wizard would guide them through the task.

Expert systems [3] use knowledge bases compiled from experts and ask questions to users to help identify problems in specific areas, such as identifying disease or troubleshooting engineering problems. Our approach differs from the expert systems because we exploit what users already know through questions, while the expert system tries to elicit the knowledge so it can map user's knowledge with its internal knowledge base.

6. CONCLUSION

In this paper, we presented an approach to building information agents by asking questions. The question-answering approach lets a user who does not know how to program to build agents that integrate and monitor information from multiple web sources. We evaluate how well the approach works by using the Agent Wizard to rebuild the flight travel application in less than 35 minutes.

7. ACKNOWLEDGMENTS

We would like to thank Parag Samdadiya for implementing the original version of the Agent Wizard.

This material is based upon work supported in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under Contract No. NBCHD030010, and in part by the Air Force Office of Scientific Research under grant number F49620-01-1-0053.

8. REFERENCES

1. Ambite, J., Barish, G., Knoblock, C.A., Muslea, M., Oh, J., and Minton, S. *Getting from Here to There: Interactive Planning and Agent Execution for Optimizing Travel*. in *The Innovative Application of Artificial Intelligence Conference (IAAI)*. 2002. AAAI Press, Menlo Park, CA.

2. Cypher, A., *Watch what I do: Programming by demonstration*. 1993: MIT Press.

3. Giarratano, J., *Expert Systems: Principles and Programming*. 3 ed. 1998: PWS Publishing Company.

4. Tessa, L., *Programming by Demonstration: Z Machine Learning Approach*, in *Computer Science*. 2001, University of Washington.