User modelling in the car

Niels Ole Bernsen

Natural Interactive Systems Laboratory, University of Southern Denmark,
DK 5230, Odense, Denmark
nob@nis.sdu.dk
http://www.nis.sdu.dk

Abstract. The paper presents work on user modelling of car drivers. The paper presents an implemented solution to user modelling in the car, which includes an aspect of location-based user modelling.

1 Introduction

The proliferation of mobile systems in the future offers new challenges for user model-based adaptation technologies. Mobile systems will communicate with their users in an increasing number of modalities [1], such as input/output speech, in addition to the customary GUI input/output modalities. Mobile systems may offer location-based and situation-based information. Coupled with the user-awareness provided by user model-based adaptation, opportunities are huge indeed.

This paper describes work on user model-based adaptation in a large mobile system under development. The system is called VICO or Virtual Intelligent CO-driver. Section 2 describes the system's functionality. Section 3 presents some findings on generic system tasks for user modelling, user identification, user modelling information, and criteria for selecting the information to model. Section 4 and 5 describe the VICO user modelling module. Section 6 describes the next step in our work.

2 The VICO system

Spoken language dialogue systems (SLDSs) are migrating into mobile environments, such as the car. VICO is such a system, addressing some next-step challenges in the context of supporting car drivers whilst driving [2]. In brief, the challenges include: (1) ease of use by a large and heterogeneous user population; (2) processing of fully spontaneous spoken input, English, German, and Italian; (3) multiple-task assistance: with navigation to addresses and points of interest in Italy, hotel reservation, restaurant reservation, and VICO system information; (4) confidence-score adaptive dialogue; (5) GPS-based location-awareness; (6) multimodal interaction through push-to-talk button and spoken input, and spoken and display (text) output; and (7) integration of adaptive user models built on-line.

3 General findings

At the start of VICO, we analysed the scope of user modelling for in-car use, taking into account the particular tasks of VICO (see Sect. 2).

3.1 Generic system tasks

VICO's *generic* UM-related tasks are [3]: (1) identify the present driver; (2) retrieve the present driver's user model; (3) optionally: create a new user model UM(Dx) for a new driver, Dx; (4) make appropriate on-line use of the present driver's user model during the driver's dialogue with VICO; (5) collect new information on the driver during the driver's dialogue with VICO; (6) update the present driver's user model with the new information gathered; and (7) store the user model whenever it has been updated with new information.

3.2 Driver identification

VICO must determine the car's current driver since cars often have several different drivers. Driver identification must be made with *near-certainty*. If it is uncertain that VICO has correctly identified the driver, misidentification will happen too often. In such cases, the driver is likely to be "mistreated" because VICO will adapt to the driver based on a wrong user model. Similarly, the modelled behaviour of the misidentified driver will tend to fudge up the misallocated user model with misleading information. Since the driver's user model cannot be invoked before identification, VICO must identify the driver *up front*, i.e. as soon as that driver starts the dialogue. Later identification means less support for the driver, and the updated user model runs the risk of having missed to collect important information on the driver's behaviour.

In SLDSs, driver identification design is a non-trivial problem. We have considered (a) voice identification, (b) a driver's code, however input to VICO, (c) driver's spelled first name, and (d) combinations of (a) through (c). We prefer to use voice identification-cum-spelling of their first names by first-time users. This combines the unobtrusive elegance of voice identification with (non-coded) first-name feedback and avoids having to remember and use a code or spell one's name on each occasion of use. This option also allows passengers to speak to the system without being registered in the UM database – passengers just have to avoid giving the system their first names.

3.3 Which type(s) of information to model?

Based on analysis of some 25-30 candidate kinds of information about driver behaviour which VICO might collect and use adaptively, we distinguish between: (T1) information on the driver's task objectives due to task goals, preferences, habits, etc.; (T2) information on the driver's communication with VICO; and (T3) information on

the driver's experience of various kinds. This information typology helps generate a structured space of candidates for observation-based adaptive user modelling, each generic type of information subsuming several specific information types, such as the driver's: preferences for hotels, restaurants, points of interest, petrol brands, or address locations; native language, communication difficulties due to strong accent or dialect, speech disorders, extreme talkativeness, elaborate politeness, frequent cross-talk with passengers, unusual speaking style, etc.; and experience in using VICO itself.

We identified a set of criteria which should be satisfied by the driver information to be modelled. These are: (C1) *universality:* unless other factors advocate in their favour, the chosen information should be top quality in terms of usefulness to all or most drivers; (C2) *quality:* the information should provide genuine driver adaptivity without significant drawbacks; (C3) *feasibility:* the functionality should be implementable without extreme or unpredictable effort, the latter being due to, e.g., a needed research breakthrough; and (C4) *verifiability:* the functionality must be based on clearly verifiable information about the driver.

An example of T1 in the information typology is: store the driver's past hotel preferences, such as number of stars, price, location (city centre, countryside), hotel chain, etc. Even if not presently told about them by the driver, VICO could offer to use those constraints as selection criteria when looking for a suitable hotel.

Let us evaluate the hotel preferences user modelling functionality using the selection criteria C1 through C4. C4 is met because the driver's hotel preferences become apparent during dialogue with VICO and they do not appear to have any significant drawbacks (C2). The functionality can be implemented without extreme or unpredictable effort (C3). Whether C1 is met depends on, at least, (i) how many users of VICO will need to book hotels, (ii) how many users will do so *en route*, and (iii) how many users have systematic hotel preferences. We do not know the answers at this point.

It is harder to identify suitable T2 information candidates. An example is a system which adapts its dialogue to drivers having a strong dialect or accent. A major problem is that any solution may be at risk as long as we do not have efficient ways of diagnosing different *possible causes* of recognition problems. Low confidence scores, many out-of-vocabulary words, or multiple error corrections, for instance, cannot tell if the cause of repeated recognition problems is a strong dialect or accent or something different, such as a driver who regularly talks to passengers during dialogue (C3, C4). T2 solutions might well satisfy C2. And even if not benefiting all or most drivers, they might benefit large fractions of those drivers who have great difficulty using spoken language dialogue systems at all, illustrating the exception clause in C1.

T3 information includes at least one obvious candidate, i.e. the driver's experience with VICO itself. The idea is to offer up-front information on VICO to all new drivers independently of whether or not a new driver asks for it. Provision of this information would seem to rather easily meet C1 through C4.

Guided by the above analysis, we have implemented a general-purpose UM module which currently facilitates drivers' hotel selection dialogue through knowledge of their hotel preferences in the past.

4 Location-dependent and location-independent adaptivity

The following driver hotel preference behaviour feature-value pairs are collected and used by the UM:

- type [VALUE = HOTEL]
- hotel name [VALUE = NAME]
- hotel address [VALUES = ADDRESS ITEMS]
- number of stars [VALUES = 1, 2, 3, 4, 5]
- hotel chain [VALUE = NAME]
- hotel location [VALUES = TOWN, OUTTOWN, COUNTRYSIDE]
- max. prices for single (S)/double (D) rooms [VALUES = S: X and D: Y Euros]
- restaurant in hotel [VALUE = TRUE, FALSE]
- protected parking [VALUE = TRUE, FALSE]

These attribute-value pairs are of two very different kinds, i.e. the generic hotel properties 1, 4, 5, 6, 7, 8, and 9, and the specific hotel properties 2 and 3. *Generic* hotel properties may belong to any particular hotel. *Specific* hotel properties imply all properties of a particular hotel: once you choose a *particular* hotel, e.g. by its name, you choose all its properties. The UM thus cannot use specific hotel properties to support the driver's hotel selection task independently of where the driver happens to be. Only generic hotel properties can be used for *location-independent* user model-based driver support. Thus, the hotel selection UM has two distinct adaptive functionalities. The first, *location-independent*, functionality offers hotels having the generic properties of hotels which the driver has preferred in the past. The second, *location-dependent*, functionality offers the specific hotels which were preferred in the past if and only if the driver is in the area in which those hotels are located.

5 User model update and use

A crucial design issue is how to *update* the UM with new information. If, e.g., the update algorithm averages over the past, and if there is a long UM record of staying in inexpensive hotels, then the UM may never fully realise that the driver has changed hotel preferences. As they are, our update algorithms for *generic* hotel properties take the two latest hotel reservations into account. As for the driver's previous choices of *specific* named hotel/location pairs, the VICO UM preserves and uses all of them, no matter how long ago it was when the driver stayed in a particular hotel.

The context of use of the hotel selection UM is that the driver asks VICO to help book a hotel, possibly adding some selection constraints, saying, e.g., "VICO, please find a three-star hotel." At this point, the hotel preferences UM is being applied.

VICO UM application raises several design issues likely to strongly affect user acceptance. Firstly, it seems clear that the UM should never override the driver's stated hotel selection constraints. Thus, if the three-star constraint in the example above conflicts with the UM, the former prevails by applying conflict resolution. By implication, if the driver's stated selection constraints suffice for uniquely identifying a hotel through querying the hotels database, the UM is not applied. Secondly, when UM

hotel selection constraints are being applied in querying the database, possibly as complements to the driver-provided selection constraints and following conflict resolution, the driver is informed that the returned hotel suggestions are the results of UM application. This allows the driver to decide whether the suggestions match current preferences. Adaptation should be made behind the driver's back.

In the case of *location-based* selection support, the UM provides a list of past selected hotel/location pairs. If the driver provides a location corresponding to a location on the list, s/he is offered the corresponding hotel(s). If no location is provided, VICO assumes that the driver wants a local hotel. If the car's current location matches a location in the hotel/location pairs list, the driver is offered the corresponding hotel(s).

VICO's historical UM consists of a managing module, a reasoning module for updating the user model of each driver, and a database. Only the reasoning module is UM task-specific. The database has (i) a record of the drivers known to VICO, (ii) a record of all observed, UM-relevant past driver behaviour organised per driver and sub-divided into the different UM tasks performed by VICO, and (iii) a record of updated UMs per driver, sub-divided into the UM tasks performed by the system.

6 Conclusion and future work

The VICO UM module has been integrated into the hotel reservation task domain agent module. Our next task is to test with real users in order to evaluate the UM's update algorithms and the principles behind its on-line use.

7 Acknowledgement

The VICO support by the EU HLT Programme is gratefully acknowledged. I would like to thank Aziz Joumady and Dymitro Kupkin who implemented the VICO UM module, and Laila Dybkjær who specified the hotel reservation task.

References

1. Bernsen, N.O.: Multimodality in Language and Speech Systems - from Theory to Design Support Tool. In: Granström, B., House, D., Karlsson, I. (eds.): Multimodality in Language and Speech Systems. Dordrecht: Kluwer Academic Publishers (2002a) 93-148.
2. Bernsen, N.O., Dybkjær, L.: A Multimodal Virtual Co-Driver's Problems with the Driver. In: Dybkjær, L., André, E., Minker, W., Heisterkamp, P. (eds.): CD-ROM Proceedings of the ISCA Tutorial and Research Workshop on Spoken Dialogue in Mobile Environments, Irsee, Germany. Bonn, Germany: International Speech Communication Association (2002).
3. Bernsen, N.O.: Report on User Clusters and Characteristics. VICO report D10, NISLab, August (2002b).