Changing machine decisions

Shapira Oz

Abstract

Introduction

User modeling has made considerable progress during the last decade, particularly in the last few years, user application expand their domain from the personal computer to smart phone, tablet computers and home devices (TV, cable modem, streamers…) <insert sentence> and change our live. today application become cross platform and cross devices , therefore the need to adaptive software is increase every day , software today are need to know every changes in their users , learn and analyze user changes in fast way through variety of information sources.

The classic user modeling approach consider the complex of the model itself, some researchers are working on creating <בכל מקום > for improving the basic design of user model systems <לתת דוגמאות למאמרים שעוסקים במבנה > in the aspect of data bases, high level design, deployment, software structure,ontology design, …<וכולי> they main mission is to contribute dynamic data base how needed in UM systems, but additional essential approach is the software adaption.

Adaption of software is the key for managing a dynamic application how reflects and respond to user behavior.

Therefore some we can catalog them to tree crude type:

1. Application with dynamic data base founded on anthology which “knows” all user stereotypes.
2. Adaptive application with learning abilities.
3. Combination of 1 & 2.

<להכניס שורת מעבר....>

According to Alfred Kobsa “User modeling research has spread into many disciplines which are concerned with the development of computer systems that are to be used by heterogeneous user populations. These fields include:

* 1. Human-Computer Interaction.
  2. Intelligent Interfaces.
  3. Adaptive Interfaces.
  4. Cognitive Engineering.
  5. Intelligent Information Retrieval.
  6. Intelligent Tutoring .
  7. Active and Passive Help Systems.
  8. Guidance Systems .
  9. Hypertext Systems and Expert Systems.”

[Alfred Kobsa “**User Modeling:Recent Work, Prospects and Hazards**”,1993]

In this lecture review will focus on the adaptive aspect, we examine if software can analyze itself by user behavior, change her code, structure, data base and follows.

**Know your user**

For creating adaptive software the first step to adopt the ability of “recognizes” users.

Recognition of user by application it’s the first step for adaptive software need overcome.

The meaning of recognition covers these fields:

* **User identification –** adaptive application need to identify her user according to the base identification parameters , according to Kobsa [“**User Modeling:Recent Work, Prospects and Hazards”,**1993] the identification process must fulfill tree tasks:
  + **User subgroup identification.** – Application must containing data base on users subgroups and to identify this subgroup from expected user population.
  + **Identification of key characteristics. –** Each subgroups has is unique characteristics (for example in music subgroup have characteristics as music type (rock,pop,juzz ….) ), the adaptive application need to draw relevant data for the identification process (without those characteristics computer system couldn’t identify the user )
  + ***Representation in (hierarchically ordered) stereotypes*** – The application-relevant characteristics of the identified user groups must be formalized in an appropriate representation system. The collection of all represented characteristics of a user subgroup is called a **stereotype** for this subgroup. The stereotype represent the order of characteristics by those stereotype , we can contribute to our application the ability of analyze user identification
* **User prior knowledge -** today most users had prior knowledge, some are first time users with minimum background knowledge while some are more experience, but even so adaptive application must obtain relevant past knowledge, filter it and analyze. This process is complex and not homogeneous; developer must consist his program to the user knowledge, save relevant data and use it when the application need to.

**Arrange your data**

Since most the user model data is not hard coded, and it can change form user to user, the adaptive application must been establish generic data structure that can handled dynamic code changes, use different data bases and add new model characteristics.

This aspect has been research by many agents:

Shlomo Berkovsky ,Dominikus Heckman and Tsvi kuflik[“**Addressing challenges of Ubiquitous user model :between mediation and semantic integration”,** 2009] have been working on integrated system between semantic standardization of user model and build hybrid system - in their research they suggest to create internal semantic language combine with generic dynamic structure, with the ability to change according to language interrupter.

Other approach is to creating a collective software with the ability to collect data from different applications or for the web Francesca carmagnole [“**Handling semantic heterogeneity in interoperable distributed user models”,** 2009] insinuating to used semantic language between application and servers for data exchange.

k.vab der skuijs and G- j houben[**“automatic generation of semantic metadata as basis for user modeling and adapation”**,2009] take this approach even further more by creating frameworks with internal ontology base on semantic command language this frameworks will establish metadata for each object in the user model with the ability to read/write is metadata .

Federica cena and Roberto Furnanri[“**a model for feature base user model interoperability on the web”,2009**] suggest on using web data collector for finding requested data ,in their approach they suggest to used SOA (service oriented architecture) for creating dialog frameworks between application/servers Although this API approach is more healthy of using semantic language since it’s can spare misunderstand between applications , it’s hard to obtain foreign frameworks in each application , it’s also obliged web interface.

There few researches who suggest complex structure for user model – not all of them suitable for the adaptive approach.

António Constantino Martins, Luíz Faria, Carlos Vaz de Carvalho and Eurico Carrapatoso [“**User Modeling in Adaptive Hypermedia Educational Systems**”,2008] summarize from kobsa (2001), Brusilovsky (2001) ,carrilho (2004), Benyon (1993) and Vassileva (1998) a generic approach for designing adaptive user model . in their paper they concentrate on educational system but they mention and used the concept of DDD and DID a structural design guideline for creating adaptive application :

**Domain Dependent Data (DDD) –**

The components of the Domain Dependent Data correspond to the Domain Model with a three-level functionality.

• Task level, with the objectives / competences of the domain that the user will have to master. In this case, the objectives or intermediate objectives can be altered according to the evolution of the user changes process.

• Logical Level, which describes the user knowledge of the domain and is updated during the user task and changes .

• Physical Level that registers and infers the profile of the user knowledge.

**The Domain Independent Data (DID) –**

are composed of two elements: the Psychological Model and the Generic Model of the user Profile, with an explicit representation (Kobsa, 2001). The psychological data are related with the cognitive and affective aspects of the user. Some studies have demonstrated that the difference between the cognitive capacities and personality aspects affects the quality of some models or styles of interaction (Kobsa, 2001; Carrilho, 2004).

These data are more permanent which allows the system to know beforehand which are the 196 characteristics that it must adapt to. (Benyon, 1993, Vassileva, 1998). The data related to the user interests, common knowledge and background are kept in the Generic Model of the Student Profile.”

The **DID** include following aspects: (Benyon, 1993; Kobsa, 2001; Carrilho, 2004):

• Initial user knowledge.

• Objective and plans.

• Cognitive capacities.

• Learning styes.

• Preferences.

• Age and type of User (Kobsa, 1997).

• Cognitive style (affective, impulsive, etc.) personality aspects (introverted, extroverted, etc.) (Laroussi, 2001).

This two main guideline are suggestion for obtaining flexible structure with the ability of adjusting user changes in the user mode.

**Predict user action**

Ones you obtain user knowledge and storage his data, you need predict is next plans and actions, this is the key element in adaptive software, a good prediction can case for good software respond while bad prediction can case for bad respond ,software crashes and user frustration.

In the resent years much emphasis has been recently put on the recognition of users plans where a plan is a sequence of user actions that achieve a certain goal.

Adaptive application need Plan recognition element which observe user input action and try to determine the all user possible plans.

According to Kobsa **[“User Modeling:Recent Work, Prospects and Hazards”,1993]** basic algorithm will not sufficient for prediction ,for good prediction the adaptive application need advance algorithm combination for the following reasons :

“1 .it is often unclear when the user commences a new plan.

2 . Actions and short action sequences may often be part of more than one plan.

3. Users may interrupt or suspend the execution of their current plans (for various

reasons, such as when issuing the ‘date’ command or when replying to an email message which they just received);

4. there is often more than one action sequence for achieving a (sub-)goal (i.e., there can be variations of user plans).”

Kobsa summarize two kind of technique for handling user prediction:

**Plan libraries -** in the approach all possible user plan already coded in pre-stored libraries, the user input action sequence will compare with the pre-stored libraries application respond will according to matching library.

**Plan construction-** in this approach the application constructs the reaction according to user input action sequence