

Adaptation in Multimedia Systems

P.t.de Vrieze (pauldv@cs.kun.nl), P.van Bommel (pvb@cs.kun.nl)
and Th.van der Weide (tvdw@cs.kun.nl)

University of Nijmegen

J.Klok (klok@oce.nl)

Océ technologies

Abstract. Multimedia systems can profit a lot from personalization. Such a personalization is essential to give users the feeling that the system is easily accessible especially if it is done automatically. The way this adaptive personalization works is very dependent on the adaptation model that is chosen.

We introduce a generic two-dimensional classification framework for user modeling systems. This enables us to clarify existing as well as new applications in the area of user modeling. In order to illustrate our framework we evaluate push and pull based user modeling in user modeling systems.

Keywords: multimedia adaptation, user modeling, adaptive systems, adaptation models, hybrid adaptation

1. Introduction

Multimedia documents are often time based and mono-directional. As such it can be hard to isolate those parts from the multimedia document which are relevant to the user. This in contrast to text documents in which one can easily scan the document for the interesting parts.

To alleviate these problems one can identify scenes in the document. This allows searching on a higher granularity level than time-unit based searching. Without a scene description however it is still hard to know which scene a user is interested in. We believe that user modeling can help here by offering a user-adapted path through the document. This belief is supported by [2] and other publications.

This paper focuses on the ways one can perform the modeling and personalisation. Both modeling and personalisation are described by the adaptation model. This adaptation model can be seen as describing how the user models need to be created, maintained and used.

We distinguish two kinds of adaptation models: a push adaptation model and a pull adaptation model. Those models are based on the direction of inference in the system. Further it is possible to combine both models into a hybrid adaptation model that combines aspects of both models.



© 2004 Kluwer Academic Publishers. Printed in the Netherlands.

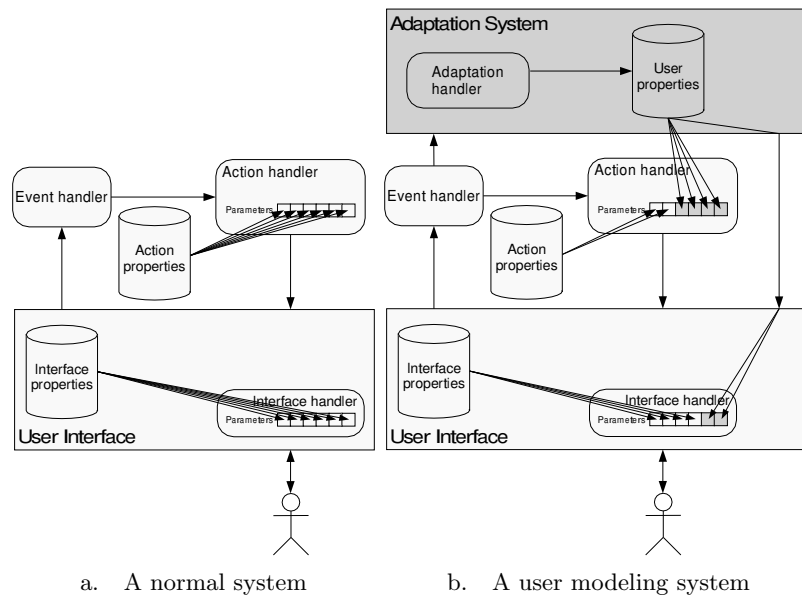


Figure 1. Comparison of normal and user modeling interactive systems

While publications have described the use of both kinds of models and combinations of them, they have not explicitly evaluated the advantages and disadvantages of those models. We believe that this is important to be able to design user modeling systems better.

In this paper we analyse the differences between the push and pull adaptation models. For that it is important to first define what a user modeling system actually is, and which parts of a system can be seen as a part of the adaptation system.

Multimedia case: In this paper we aim to explain the theories based on the following case: We have a multimedia document that is composed of various media components. Those components are for example video or audio fragments. For some fragments there are also alternatives. Those alternatives can be used to personalize the presentation. In this case we suppose that we have much more material than can reasonably be presented to any one person. It is also unlikely that every user is interested in all fragments.

2. Overview of User Modeling Systems

A user modeling system is a system that shows adaptive behaviour concerning its interaction with the user. For explaining the difference between conventional systems, i.e interactive systems that do not em-

ploy user modeling, (see Figure 1.a) and user modeling systems (see Figure 1.b) we first need to describe conventional systems in a suitable way. Then we need to describe user modeling systems, and compare them. In the next two sections we will describe both conventional and user modeling systems.

Advanced multimedia documents can be seen as interactive systems that have internal (e.g. timer) events and external (user) events (see Figure 1.a). Each event can induce a state change, after which new user actions are possible.

In designing a user interface several choices have to be made concerning the looks and behaviour of the interface. Many of these choices are implicit or given by default choices from guidelines. For the sake of being able to compare a conventional system with a user modeling system we assume that the choices are explicit. We call those choices interface properties. The interface properties determine both the behaviour and looks of the user interface.

In a conventional system user actions induce events. These events trigger system actions and interface changes. These actions and interface changes can differ based on the interface properties

In a system based on user modeling (see Figure 1.b), the behaviour of the various handlers may be affected by user properties in addition to the handler specific properties. See e.g. [6] and [7] for systems that show such a change of behaviour. Those user properties are supplied by the adaptation system and can be seen as questions that can be asked by the system about a specific user property. As the adaptation system can be seen as the authority on the user, the questions should be in such a way that all inference happens inside the adaptation system.

As a consequence of the user properties influencing the handlers the user interface now takes into account the user model as its behaviour is determined by the user interface handler. The same goes for the action handler.

The user properties are provided by the adaptation handler. The adaptation handler generates these properties based on events fed to it by the event handler. The main point of user modeling is about how to go from these events to the user properties.

For our case this would mean that the user has the ability to choose between fragments or has functions like fast-forward or next-chapter to his disposal. Pushing a button or choosing for a fragment would be seen as an event, and is used to model the user.

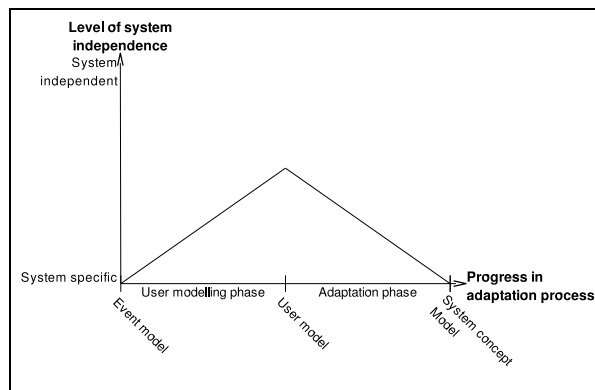


Figure 2. A two-dimensional user modeling classification framework

3. Further Analysis of User Modeling Systems

To evaluate user modeling systems it is very useful to have a clear method for comparing them. For this purpose we have developed a two-dimensional classification framework. Our framework aims towards all kinds of user modeling systems from a theoretical point of view. In this it differs significantly from the framework in [10].

Figure 2 presents the proposed framework. Along the horizontal axis is the inference process. This inference process works on transforming the event model into the user model and the user model into the system concept model. The event model consists of the actual events generated by the system. The user model consists of the most system independent user properties, and the system concept model consists of all the questions about the user that can be asked by the system.

For certain user properties many derivation steps are necessary, and for others only a few. Because of this reason we model the progress in that process, not the steps. Further, we define the model that is least system specific to be in the middle. For that reason all systems will have their highest point in the middle.

On the vertical axis we model system independence. At the start of the adaptation process are the events generated by the system. These events form the *event model* and are maximally system dependent. An example of such an event could be: “The user skips most of scene 3”.

For personalisation purposes it is more useful to know user properties than to know events. To this extend the adaptation model describes inference rules that can be used to infer the *user model*. An example of a user property is: “The user is not interested in the history of multimedia (the subject of scene 3)”

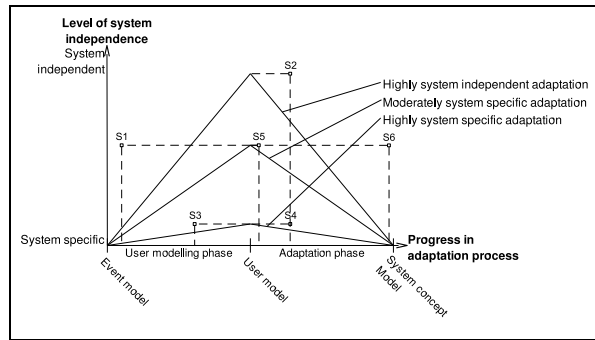


Figure 3. Use of the two-dimensional classification framework

At this point, where the user model is known the system can use it for personalisation. This personalisation involves using the answers to a number of questions about the user. These questions and their answers form the *system concept model*. An example of such a question is: “Should scene 9 which has a subject of multimedia history be displayed” The inference rules that are used for inferring the system concept model from the user model are also described by the adaptation model.

We can use the framework of Figure 2 to determine two properties of systems. Firstly, we can look at the relative height of the triangle to determine how system specific an adaptation system is. For example in Figure 3 we see the systems S2 and S4. S2 is more system independent than S4. This could mean that S2 can be more easily be extended to provide more or different personalisation.

The second property we can distinguish is, where in the inference process a persistent model is stored. This is an important measure as the process is different before and after storage. Before storage push adaptation needs to be used to create the model. Push adaptation here means that the arrival of an event generates a waterfall of subsequent events that lead to updating the persistent model. We call this push adaptation. We will discuss the advantages and disadvantages of push based systems in section 5.

After storage we need to use a pull strategy to perform adaptation. This starts with the system requesting the value of a certain property from the adaptation system. For determining the value of this property the adaptation system might want to use the values of other properties that might also need to be calculated. This goes on until the persistent model is used. We call this pull adaptation.

As an example of the use of the framework we look at Figure 3. In Figure 3 there are six systems with all different properties. System S1

is almost purely a pull-based system, as it's persistent model is created very early on in the inference process, while S5 is can be classified as a hybrid system and S6 is a rule-based system. The other systems are all different kinds of hybrid systems. Note that S5 is almost in the middle, but a system completely in the middle would be rather unrealistic.

Based on the locations of the systems in Figure 3 we can say things about the systems, and especially their relations with each other. As an example looking at systems S3 and S5 we can say that system S3 has a bias on pull modeling compared to S5 and that S3 is more system dependent than S5. This can be used to say things about these systems like: “the persistent model of S3 is probably bigger than the persistent model of S5” and “It is probably more easy to extend the adaptation system of S5 than to extend that of S3”.

4. Properties of a User Modeling System

We have seen that there is push adaptation and pull adaptation. In the coming sections we want to analyse the advantages and disadvantages of these and hybrid adaptation strategies. To make an analysis we have identified a number of key properties of user modeling systems. Although some of these properties are not easily measured, we still believe they are important.

- *Adaptability.* The user should be able to manually adapt his model to a certain extend.
- *Speed.* The users' perception of the system's speed should not decrease.
- *Extensibility.* The system should be extensible while retaining the existing knowledge about its users.
- *Model size.* The model size should not grow too large.
- *Analysis possibilities.* The chosen kind of adaptation model should allow for all kinds of analysis techniques.
- *Privacy.* The system should be designed in such a way as to guarantee the highest possible level of privacy for the users.

Some of these properties are more important than others. It mainly depends on the application. We will not further discuss privacy as it depends mostly upon the application and very little on the adaptation model.

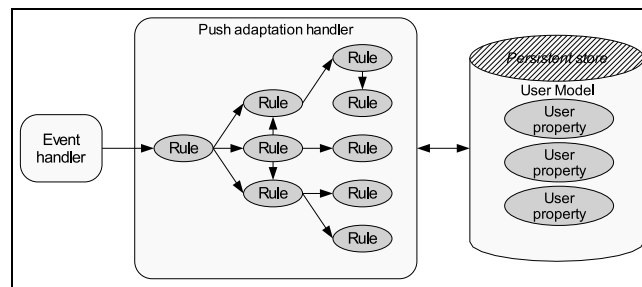


Figure 4. A push adaptation model

5. Push Adaptation Models

Push adaptation models are adaptation models, that let events propagate on to the values of a user model. Many systems that use push adaptation models use a rule-based model as employed in [13]. This paper describes the adaptation system of the AHA! system, a research system for creating adaptive hypermedia. These rule-based models are based on Active Database technology and as such inherit limitations from database systems.

One advantage of push adaptation is the fact that the contents of the user model are well aggregated. This has as advantage that those contents can be easily understood. Another advantage is that the relative size of the user model stays small, and that the size does not change during regular use of the system. This does however impede the possibilities for basing values of newly introduced attributes upon already seen behaviour of the user. In this section we evaluate push based adaptation models based on the points from section 4.

- *Adaptability.* Because the user model stores end values it will be fairly easy for users to adapt the model to their wishes as the results of their changes are obvious and local. There could be too many possibilities for changes though.
- *Speed.* Provided that the amount of rules stays within limits there are no serious speed issues with push adaptation models.
- *Extensibility.* Push adaptation models are similar to database theory, and are often based on it. They have one problem that is similar to the problem of databases. Database systems are not good in data model change. This is the same with rule based adaptation models. At a moment that the adaptation model changes, values for new properties need to be calculated which can be expensive in terms of time.

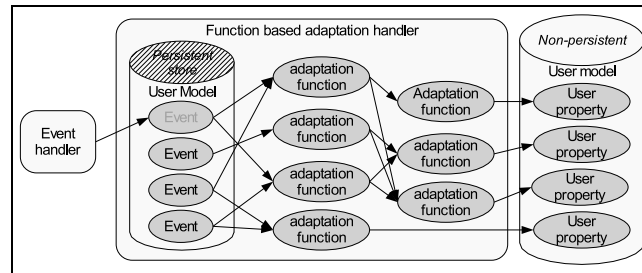


Figure 5. A pull adaptation model

- *Model size.* A push adaptation model has a user model with a stable, limited size. This is because events are aggregated into the user model at the moment they happen.
- *Analysis possibilities.* The fact that event aggregation in rule based adaptation models happens at the moment the events happen makes it hard to impossible to perform time based analysis on user actions. Also aging (as weighing recent events higher than older events) is hard to implement.

From this point by point overview we can see that push adaptation models are especially good in the areas of model size and complexity. The weakest points lay in extensibility of the model.

Push adaptation models are very popular within the domain of educational systems. Those systems can be characterised by the fact that the user properties that need to be modelled are often one-directional. Push adaptation models are used in other system to though. Examples of push adaptation models can be found in: [13],[3],[4] and [10].

6. Pull Adaptation Models

Pull adaptation models perform adaptation from a different direction than push models. In the extremity a pull adaptation model records all events in the user model. High level attributes are then derived based on lower level attributes and querying of the event record.

The pull model is based on calculation at the moment of the request. As such extension of the adaptation model is a lot easier than with push models.

One problem with the functional model though is the fact that the recorded data has very little value on itself. For adaptation purposes one would prefer to know concepts of user behaviour, not individual events. Push adaptation makes sure that concept generation needs to

be done only once. Certain concept generation rules might be quite complex and would take a long time to recalculate on every use. To allow this for the pull model caching could be very helpful.

- *Adaptability.* Pull models have problems with adaptability. This is caused by the fact that user models store huge amounts of abstract facts. One can not expect even experts to be able to make changes with predictable results in such a user model. An exception to this is that exclusion of time periods is easy in pull models. All events have a timestamp, and removal of facts just leads to different results of the functions.
- *Speed.* As user models that store events can get very big there is certainly the need to use extensive caching of intermediate results. The language used to query the user model could provide tools for incremental queries, where old results get enhanced with newer facts. Also the set of matching events can be stored to be used as a base for the query at a later time.
- *Extensibility.* The pull adaptation model scores very well on the point of extensibility. As abstract events are stored there will be many cases where new user attributes can be derived from behaviour before the attribute was introduced.
- *Model size.* Model size is a disadvantage of the pull adaptation model. With a little loss on model quality though old events could be aggregated into smaller parts or even discarded. If the amount of users of the system is not very high we don't believe there is a big problem on model size.
- *Analysis possibilities.* The pull adaptation model allows for more analysis possibilities. As all data in the user model is time stamped, time based analysis and aging are easy performed. There are no analysis possibilities in the push model that are not available in a pull model.

Pull based adaptation models are currently not common. They are especially utilised in cases where combinations of events need to be analysed to retrieve the goals of a user. A pull based adaptation model is for example used in [8]. In this article the interaction of users with a word processor is studied. This interaction is used to make recommendations to the user on doing things more efficient. Another example of pull models are attentive systems. They need to determine whether a user can be disturbed. These systems are highly dynamic and thus do not fit well with the static nature of the push model. Examples of these

systems can be found in [1]. Other pull systems can be found in: [7], [5] and [12].

7. Hybrid Adaptation Models

Both adaptation models have their advantages and disadvantages. The push model for example might need workarounds for ages (as being dynamic properties changing every second). The pull model is not very good at storing static user properties, and can be very space inefficient.

Looking at the two phases of the user modeling process we can see that while the model use phase is especially suited for a pull approach, the modeling phase is more directed towards a push approach. We can use this by using a hybrid adaptation model. Such a hybrid model can combine the advantages of both pure models. Basically the push model has a place in the user modeling phase and the pull model in the adaptation phase.

- *Adaptability.* By storing system independent user properties the hybrid system can offer the user clear high-level properties the user can change (not properties that are either too abstract events with unclear results (pull), or many system specific properties which have too localised results (push). This could mean that the adaptability of a hybrid system is better than both the rule based and functional approaches.

This adaptability advantage could vanish if the rule based and functional models offer adaptability of intermediate concepts that are at the same position as the user properties of the hybrid model.

- *Speed.* Hybrid adaptation models should relieve many of the possible speed problems in the functional model as it can reduce the complexity of the event store in the functional model. It also avoids the rule explosion that comes with a big interrelated push model.
- *Extensibility.* The modeling process goes from very system specific events to less system dependent concepts. Those system independent concepts can be building blocks for extension. System dependent events cannot really do that. So there is no real loss in extensibility when using a hybrid model where concepts are stored that are less system dependent.
- *Model size.* In the hybrid model the model size can be significantly lower than the pull model as not single events are stored, but more high-level concepts.

- *Analysis possibilities.* As hybrid adaptation models allow for different adaptation strategies for different properties, they can retain most of the analysis possibilities that function-based adaptation models have. At the same time hybrid adaptation models can take advantage of properties of rule-based adaptation models where the analysis possibilities offered by a function-based approach is not necessary.

Hybrid adaptation models are more common than one would expect. They can often be found in systems where no special effort was put to the adaptation model. Examples can be found in [10] and [9] where recommender systems are discussed.

8. Multimedia adaptation models

In our sample case we have one document that consists of multiple sub-documents. There are also a number of actions that can be performed. For example: fast-forward (this text is uninteresting), fast-backward (I want to hear this text again) and fragment selection.

An example adaptive multimedia document could be build up from media fragments. Those fragments have subjects (standardized) and subjects have depths. Those depths say how much that subject is explored in the fragment. Some simple rules that might be used are: “If the user fast-forwards the fragment, lower the preferred depth” and “If the user fast-backwards the fragment, increase the preferred depth”.

In the above rules we can see the advantage of push modeling. We record from the user the preferred subject depth and use that for selecting fragments. When we need to select a fragment the reverse happens. The system needs to find out which fragment to select. This means that given the user’s preferred levels, a best-fitting fragment needs to be selected. For a push model this would mean that every time that a preferred depth is changed, all the answers to the “which fragment should be selected?” question that involve fragments with the subject need to be updated. Besides being inefficient this can also limit extension of the multimedia document when users revisit it.

We feel that for the fragment selection question it is more appropriate to do the calculation on demand, i.e. pull adaptation. As the more intensive calculations have been already performed we do not think that this assembling causes a performance hit. Further predictive caching techniques could be used to do just-in-time calculation of the needed answer.

9. Conclusion

In this paper we have introduced a framework for classifying user modeling systems. With this framework we have shown that there are two basic categories of adaptation: rule-based adaptation and function-based adaptation. We have pointed out several examples of such systems including multimedia systems.

Besides the rule-based and function based systems there is also a possibility for hybrid systems. We believe these hybrid systems can be able to solve the problems with both pure approaches, and combine their strong points. We have given an example of a multimedia system using this hybrid adaptation.

We also pointed out that user modeling systems can have differing system dependence. This system dependence measure can be an indication of ease of extensibility of the system.

References

1. ACM: 2003, 'Attentive User Interfaces'. *Special Issue of Comm. ACM* **46**(3), 30–72.
2. Boll, S. and W. Klas: 2001, 'ZYX – A multimedia document model for reuse and adaptation of multimedia content'. *IEEE transactions on knowledge and data engineering* **13**(3), 361–382.
3. Bra, P. D., A. Aerts, G. Houben, and H. Wu: 2000, 'Making General Purpose Adaptive Hypermedia Work'. In: *Proc. of the WebNet Conf.* pp. 117–123.
4. Brusilovsky, P. and D. W. Cooper: 2002, 'Domain, task, and User Models for an Adaptive Hypermedia Performance Support System'.
5. Bull, S. and G. McCalla: 2002, 'Modelling cognitive style in a peer help network'. *Instructional science* **30**(6), 497–528.
6. Fink, J. and A. Kobsa: 2002, 'User Modeling for Personalized City Tours'. *Artificial intelligence review* **18**(1), 33–74.
7. Fleming, M. and R. Cohen: 1999, 'User Modeling in the Design of Interactive Interface Agents'. In: *Proc. o. t. 7th int. conf. on User modeling.* pp. 67–76.
8. Linton, F., D. Joy, and H. Schafer: 1999, 'Building user and expert models by longterm observation of application usage'. In: *Proc. o. t. 7th int. conf. on User modeling.* pp. 129–138.
9. Maglio, P. P. and R. Barrett: 1997, 'How to Build Modeling Agents to Support Web Searchers'. In: *User Modeling: Proc. 6th Int. Conf., UM97.*
10. Montaner, M. and B. Lopez: 2003, 'A Taxonomy of Recommender Agents on the Internet'. *Artificial intelligence review* **19**(19), 285–330.
11. van Bommel, P. and T. P. van der Weide: 1998, 'Multimedia information filtering on the WWW'. In: *Int. forum on multimedia and image processing.*
12. Virvou, M., J. Jones, and M. Millington: 2000, 'Virtues and Problems of an Active Help System for UNIX'. *Artificial intelligence review* **14**(1-2), 23–42.
13. Wu, H.: 2002, 'A reference Architecture for Adaptive Hypermedia Applications'. Ph.D. thesis, Technical University of Eindhoven. isbn: 90-386-0572-2.