A Generic Adaptivity Model in Adaptive Hypermedia

P.t. de Vrieze, P. van Bommel, and Th. van der Weide

University of Nijmegen
{pauldv,pvb,tvdw}@cs.kun.nl

Abstract. For adaptive hypermedia there is a strong model in form of the AHAM model and the AHA! system. This model, based on the Dexter Model, however is limited to application in hypermedia systems. In this paper we propose a new Generic Adaptivity Model. This state-machine based model can be used as the basis for adaptation in all kinds of applications.

1 Introduction

The AHAM model [1],[2] forms an important model in the area of adaptive hypermedia. This model, based on the Dexter Model, however is limited to application in hypermedia systems. In an aim to provide such a model for generic user modelling we have developed the Generic Adaptivity Model (GAM). This model provides a general model for an adaptation engine that can be used in a modular way to provide adaptivity in a variety of applications.

The Generic Adaptivity Model is based on our two-dimensional classification framework as defined in [3] and [4]. It includes push and pull modelling in the shape of rules and questions. This allows designers of adaptive systems to make better decisions on what to store in the User Model. This allows for more flexible adaptation systems.

A prototypical engine has been developed based on the model. Rules and questions are implemented using a simple but full featured imperative scripting language.

2 Generic Adaptivity Model

Aiming at a theory for providing adaptation to applications in general we have developed the Generic Adaptivity Model (GAM). The GAM model is based on a state-machine view on applications. In this state-machine based view we see an application as a state-machine. At each interaction an event gets generated. This event in someway induces an action that results into a state change in the system. Such a state change can be paramaterised by external values. A user model can be used as the source of such values. For this the events also get fed into the adaptation engine. (see Fig. 1)

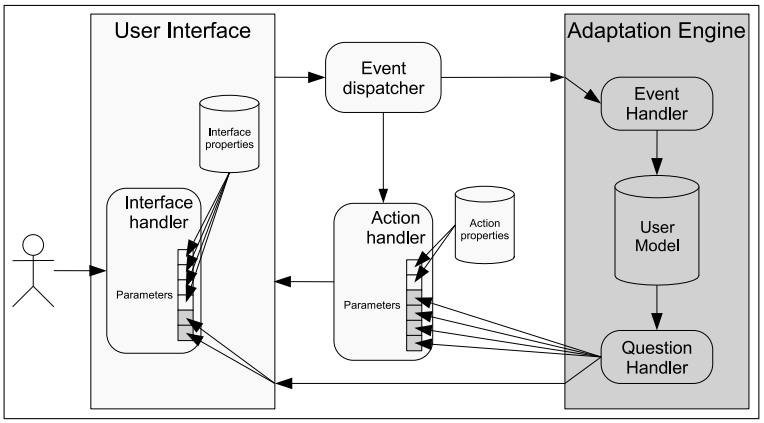

Fig. 1. An interactive system with user modeling

Adaptation Description The two main components of the GAM are formed by the Adaptation Description and the User Model. An *Adaptation Engine* (AE) requires a description of the adaptation it should perform. This description consists of several parts that together form a logical union. This logical union is the *Adaptation Description* (AD).

Interface Model As our AE is general and does not limit itself to hypermedia there are many events that it might need to react on. Similarly there are many questions that a program might want to ask to the AE about the user.

To allow the programs that are clients of the AE to change the actual adaptation logic without needing a change itself, we have the *Interface Model* (IM). The Interface Model describes exactly which events and which questions are available to clients of the AE. It also describes their parameters. The IM does however not contain any logic and as such can be equal for two different adaptation models.

Domain Model Like the AHAM model the GAM also provides a *Domain Model* (DM). In GAM however there is no notion of "concept" as used in the AHAM model. As the GAM model is a general model we cannot use abstractions like the Dexter Model [5]. The GAM model is a model on a higher level of abstraction in which the designer of the DM determines the meaning of the DM elements.

The Domain Model is a source from which the User Model is determined. Every element of the Domain Model has its reflection on the User Model.

The DM consists of definitions of attributes. These attributes have a type: string, integer, boolean, floating point value or an array of such attributes. Further the attributes have a name and a default value. This default value is the value that gets used for this attribute when no value is specified for the attribute in the user model.

Adaptation Model The *Adaptation Model* (AM) specifies the actual dynamic behaviour of the AE with respect to this AD. The AM splits the dynamic behaviour into two phases. These phases correspond to the framework as we have described in [3] and [4]. Basically the two phases represent updating of the user model in response to events and the querying of the user model.

Handling of events happens with a system based on ECA rules [6] similar to AHAM. For querying we have, however, also added a question (or function) based system. This question system allows to store intermediate values at the right level in the User Model. We have evaluated in [4] that there are several advantages and disadvantages of storing user properties instead of calculating them from more basic properties.

Rules The rules in the adaptation model correspond to push adaptation as described in [4]. They are also very similar to the rules as provided by AHAM. The rules in our model are triggered by the events posted to the AE.

Event handling works as follows:

1. An event list is initialised with the posted event.
2. The first event in the list is removed and taken for evaluation.
3. All the rules whose condition is true are evaluated, not only the first one!
4. Every change to an attribute in the user model generates a change event that has the name of the attribute as argument. All those events are added to the event list
5. If there is at least one event in the event list, go back to step 2.

Questions Besides rules the Adaptation Model also provides questions. Questions are basically functions as seen from a programming context. Questions correspond to pull adaptation as described in [4]. As questions are used for querying the user model they are not allowed to update any values in it.

Question evaluation is fairly straightforward. The questions are specified as a program. In these programs it is possible to use other questions as functions for intermediate values. User Model attributes are also accessible as variables.

User Model Besides the Adaptation Description, the GAM defines the existence of a User Model (UM) for each user. Each UM is tied to a specific AD and describes the properties of the user as specified by the DM.

Like the UM in AHAM our User Model is an *overlay User Model* that contains only the changed values. This allows for efficient storage and flexibility towards changes of the adaptation description.

3 Conclusion

Comparing the AHAM model to the GAM model we can find the following differences:

– The GAM aims to provide a generic adaptivity model. As such the GAM is more low-level than the AHAM model and does not provide high-level constructs by itself. The GAM however does allow the specification of more high-level models on top of the GAM model.
– The GAM provides an explicit interface model. In effect the AHAM model can be seen as having an implicit interface model. This is sufficient for AHAM as the set of events available within a web constant is fixed and limited to access to concepts/pages and AHAM does not offer questions.
– The biggest difference between AHAM and the GAM is that in the GAM the concepts of push and pull adaptation have been added. Thus allowing the adaptation to be more flexible in it's storage.

From the above we can conclude that the GAM model is more suited as a description for generic adaptivity than the AHAM model is. In the area of adaptive hypermedia we are currently still missing a model that on top of GAM can provide specific hypermedia concepts. However we believe that, once such a model has been developed, the GAM model is able to provide functionality that extends the AHAM model's functionality.

In further research we plan to create such a hypermedia model on top of GAM. We plan to use this model to create an adaptive hypermedia engine based on the generic engine we have already implemented. To compare the functionality of this engine we plan to write a translator that can translate AHA! documents into a form that can be understood by our engine.

Additionally we also plan to research how artificial intelligence techniques such as agent based learning and Bayesian or neural networks can be integrated into our model.

References

1. de Bra, P., Houben, G.J., Wu, H.: Aham: A dexter-based reference model for adaptive hypermedia. In: Proceedings of the ACM Conference on Hypertext and Hypermedia, Darmstadt, Germany (1999) 147–156
2. Wu, H.: A reference Architecture for Adaptive Hypermedia Applications. PhD thesis, Technical University of Eindhoven (2002) isbn: 90-386-0572-2.
3. de Vrieze, P., van Bommel, P., Klok, J., van der Weide, T.: Towards a two-dimensional framework for user models. In: Proceedings of the MAWIS03 workshop attached to the OOIS03 conference, Geneva (2003)
4. de Vrieze, P., van Bommel, P., Klok, J., van der Weide, T.: Adaptation in multimedia systems. Multimedia Tools and Applications (2004) to appear.
5. Halasz, F., Schwartz, M.: The dexter hypertext reference model. In: Proceedings of the NIST Hypertext Standardization Workshop, Gaithersburg, MD, USA (1990) 95–133
6. Aiken, A., Hellerstein, J., Widom, J.: Static analysis techniques for predicting behavior of active database rules. ACM Transactions on Database systems **20** (1995) 3–41