Self-regulated adaptivity as a design and authoring support tool

Alexandros Paramythis

Johannes Kepler University,
Institute for Information Processing and Microprocessor Technology (FIM)
Altenbergerstraße 69, A-4040 Linz, Austria
alpar@fim.uni-linz.ac.at

Abstract. This paper discusses the changes self-regulated adaptivity will potentially bring about in the culture of designing and authoring adaptive systems. Two dimensions are explored in parallel: the opportunities arising from the employment of self-regulation itself as a tool that can facilitate the design of adaptation; and, the additional requirements self-regulation places upon the traditional authoring process and on the adaptive system itself. The discussion is structured around a specific example tackling an often-discussed problem in the domain of adaptive course delivery systems: adaptive annotation of concept-based hypermedia links. The paper builds upon this example to argue that self-regulation is a viable solution to the "ground up" design of adaptive systems, and may be used even in (or, for that matter, best suit) cases where there is little empirically validated evidence to support design decisions.

1 Introduction

The design and implementation of adaptive systems is a challenging process. Among the major historical factors behind this fact are: (a) the unavailability of open platforms / frameworks that can be used as the basis for implementing adaptivity; (b) a lack of implementation support tools; and, (c) the lack of hard, empirical evidence that can be used as input to the design of adaptivity. In recent years, each of the above areas has been addressed by research to a greater or lesser degree. For instance, on the framework side, one can now employ and build upon open-source frameworks such as AHA! [3]. On the implementation support side, a number of tools have emerged to support the declarative definition / composition of adaptive behavior; there are currently both tools that are delivery framework-specific (e.g., the AHA! authoring support tools [4]) and ones that are largely framework-independent (e.g., MOT [2]). Last but not least, great attention is being paid recently to the principled evaluation of adaptive systems, which progressively gives rise to a body of knowledge that can be directly applied in adaptation design.

Despite these recent achievements, however, there are still at least two aspects of the development process of adaptive systems, that are directly related to the aforementioned three dimensions, and are in need of additional research attention. Firstly, as adaptivity is becoming widespread, we are more often called upon to design adap-

tive system behavior in novel interactive contexts and application domains. There are simply cases where the design has to evolve from a basis comprising educated guesses, design decisions that draw upon intuition, and, sometimes, only remotely related empirically derived evidence. Secondly, the current generation of adaptive systems, although a considerable improvement over their static predecessors, still share a deficiency with them as far as interaction with humans is concerned: they have no means of reasoning about, or modifying in any way, their own adaptive behavior. A direct side effect of this is that any evolutions to the adaptation design must be brought about by the designers themselves. A typical lifecycle for such evolutionary steps involves the setting up of experiments with real end users, the analysis of results, the identification of required modifications in the system's adaptive behavior (e.g., changes in the adaptation logic, or in the adaptation patterns), and the reformulation of those modifications for inclusion into the adaptive system.

This paper argues and provides an example of how self-regulated adaptivity (or, self regulation for short) can address, at least to some extent, both of the problems briefly outlined above. Self-regulation, in this context, refers to an adaptive system's capacity to observe its own adaptive behavior, assess its efficacy in attaining design goals expressed in computable form, and modify its adaptive behavior in an attempt to reach said goals [5]. In other words, self-regulated adaptivity is a form of meta-adaptivity, and lends itself as a pragmatic step towards the second adaptation cycle first put forward by Totterdell and Rautenbauch in the late 1980s [8]. The differentiating traits that self-regulated adaptive systems have, as compared to their ordinary counterparts, are self-evaluation, and the ability to evolve themselves through that. What this paper aspires to demonstrate and discuss is that these traits offer a design tool, or vehicle, that potentially facilitates the design of adaptive systems.

The rest of the paper is structured as follows: The next section presents a worked out example of a system capable of adaptive presentation of hypermedia e-learning materials, and follows through the potential design process, as this would be manifested in the presence of self-regulated adaptivity. The subsequent section then goes on to analyze the presented example, point out the benefits potentially derived from applying the approach, and extract the requirements that such an approach would impose on the system. It further discusses the potential of using self-regulation as a specific form of meta-adaptivity to achieve the desired objectives.

2 Meta-adaptive system design

This section presents a worked out example of a case study of the design of a meta-adaptive system. An important point to note is that, for the sake of simplicity, this section does not discuss one very important aspect of the prerequisite infrastructure, namely the approach and steps required to effect self-evaluation, which is an integral part of self-regulated adaptive system behavior. Instead, a "deus ex machina" view of the respective process is adopted, and further discussion is deferred till section 3.

2.1 The case study

The basis of our exemplary system design is a simple, yet popular, adaptive function in adaptive hypermedia systems: the annotation of links within learning content.

For our needs we will assume a system that exhibits characteristics common to a large range of adaptive systems in the field (e.g., AHA! [3], or NetCoach [9]). The system's most important features can be summarized as follows:

- The system's domain model is a small, course-specific ontology comprising learning concepts and semantic relations between these concepts (e.g., "prerequisite-of"). The domain model also has explicit representations of the modules / pages that make up the actual learning content, annotated with the semantic relation between each module and the respective concept (e.g., "explains", "provides-examples-for", "tests-knowledge-of", etc.)
- The system's user model is a simple overlay model (over the domain model), with a small number of discrete (and possibly mutually exclusive), user-specific "states" with respect to each of the concepts in the domain model (e.g., "has-not-seen", "has-seen", "has-learned", etc.)
- Individual user models are updated on the basis of directly observable user activities (e.g., visiting the module that "explains" a concept, responding to test items)
- Based on the current state of an individual user model, the system can decide on recommendations regarding the future visits of different modules / pages. Once again, these recommendations can be assumed to belong to a small set of discrete, mutually exclusive, module-centric ones (e.g., "ready-to-learn"). It is exactly on the basis of these recommendations that link annotation is being considered in the context of our example.
- Although this is not pertinent to the ongoing discussion, one can assume for completeness that in the adaptive system at hand adaptation logic is expressed through simple adaptation rules, such as in the case of [7].

Further to the above, we will assume that the user model contains other user attributes, some of them explicitly provided by the user (e.g., demographic data), and others inferred from user activity. As an example for the second category, consider "expertise", which refers to the user's familiarity with the system and is updated through a function that takes into account factors such as how often the user enters the system, how long the user's sessions are, "coverage" of system facilities in typical usage patterns, etc. (might be over more than one courses).

The design question at hand is whether to present users with the system's recommendations with respect to links present in a page, and, if yes, what is the best way to annotate links to convey the semantics of the system's recommendations. Although there is a considerable body of research on this question, for the purposes of this example, we will assume that the system's designer has no empirical evidence to support the considered design alternatives. The alternatives themselves are encapsulated in five different strategies as far as link annotation is concerned:

Strategy A: *No annotation*. This can be considered the base-line strategy, and would simply involve not exposing the user to the system's recommendations. In our very simple example, this might be identical to the non-adaptive version of the system.

Strategy B: Annotation using different link colors. In this strategy different colors are used directly on the links to signify system recommendation. From a human-computer interaction perspective, this strategy can be seen as one that might require some learning on the part of the users, but would add as little clutter to the page as possible. One potential problem is that links colors already have well-defined meanings (e.g., whether a link has been visited) that may clash with the strategy at hand.

Strategy C: Annotation using bullets of different colors. This is very similar to strategy B above, with the exception that the colors are applied externally to the links. Annotations (i.e., the bullets) are dynamically added to the document.

Strategy D: Annotation using custom icons. A variation of strategy C above, with the bullets replaced by icons that are intended to carry more semantic information, in effect embodying the system's recommendation in a pictographic manner. The rationale for their use is that they would presumably be more readily recognizable by novice users. On the other hand, they might add a lot more visual noise to a page.

Strategy E: Link hiding. This strategy involves hiding (although not disabling) links (see [1]), for which the system's recommendation is that the user is not yet ready to visit them. This strategy is intended as a more direct attempt to guide the user as compared to the previous ones.

Given the strategies above, the design question at hand is which one(s) to use, and for which users or usage scenarios. Note that the strategies, as formulated herein, are not necessarily mutually exclusive (e.g., C cannot be combined with D, but both C and D can be combined with E). Also note that it would be desirable to identify situations (e.g., increases in the user's "expertise", as recorded in the respective attribute of the user model) that might justify a transition from one strategy to another.

It should finally be explicitly pointed out that the objective of the design process *is not* to arrive at a single alternative that can be used for all users (which, in itself, would defeat the purpose for employing adaptivity in the first place). The primary assumption behind the design is, actually, that no alternative will fit all users under all circumstances, and the desideratum is to identify the cases where a specific alternative would be the best option for a given user and context of use.

2.2 Evolution of adaptive behavior

The designer starts out with no evidence about when and under what conditions to use each strategy, or whether, indeed any one strategy is "better" than all the rest. Each strategy has obvious trade-offs as far as flexibility and user control over the navigation process is concerned. The designer's goal however, is clear: students should encounter concepts that they are not "ready" for as little as possible, and this should be achieved with the least possible restrictions on interaction / navigation.

As already mentioned, this section does not go into a discussion of how self-evaluation might be effected in the example system, but it assumes that self-evaluation does takes place, and that its results drive the subsequent design iterations. Continuing with the example introduced above, we will look at three potential iterations that the design process could have gone through. Please note that these iterations are not "normative", and are meant mainly to facilitate the discussion of how

different types and levels of self-evaluation results can influence the design process. Although the proposed approach is by nature iterative, there is neither any ground, nor any motivation to constraint the number of iterations required; this is a design decision that needs to be made on a per case basis.

Iteration 1: "Tabula rasa"

The first step of the design would involve the encoding of the strategies as sets of adaptation actions – something that could be done in "hard-coded" programmatic logic, or, preferably in a declarative fashion (e.g., as in [6]).

Since the designer has no evidence regarding the applicability of the different strategies, however, these cannot be directly assigned to adaptation logic (e.g., one cannot, yet, create adaptation rules that would link strategies to attributes in the user model). The system would then need to be able to recognize these strategies and apply them (separately or combined) in more or less a trial-and-error fashion.

The design information that already exists, and can be conveyed to the system, is which strategies are mutually exclusive, and which ones can be applied in combination. Using the information in Table 1, the following twelve combinations can be identified: (i) A, (ii) B, (iii) B+C, (iv) B+D, (v) B+E, (vi) B+C+E, (vii) B+D+E, (viii) C, (ix) C+E, (x) D, (xi) D+E, (xii) E.

Given these constraints, and a suitably encoded, computable representation of the design goal stated earlier (i.e., to help guide users to modules / pages that are best suited to their current level of knowledge), the system is then ready to undergo the first round of user testing.

Table 1. Considered adaptation strategies and their compatibility; ● signifies that the strategies are compatible, and X signifies that the strategies are mutually exclusive.

Strategies	A. No annotation	B. Colored links	C. Colored bullets	D. Custom icons	E. Link hiding
A. No annotation		X	X	X	X
B. Colored links			●	●	●
C. Colored bullets				X	●
D. Custom icons					●
E. Link hiding					

Iteration 2: Selecting, categorizing and prioritizing (combinations of) strategies

We will assume that the results of the first round of testing do not yet suffice for building a comprehensive body of adaptation logic to guide the system's adaptive behavior. They do, however, provide enough evidence for the following:

– Eliminating strategies (and combinations thereof) that do not seem to meet the desired design goal under any circumstances. In the context of the ongoing example this might include, for instance, strategy B and all its combinations (presumably because changing links' colors is confusing for users).
– Categorizing and providing a tentative "ranking" of the remaining combinations, based, respectively, on their design / interaction semantics, and on the rate of success they have exhibited during the first round of testing; of course, other factors (for instance, how restrictive or obtrusive alternatives are) can be used as well.

The aforementioned categorization and ranking process, might result in something like the following in the case of our example:

Category I: Includes only strategy A and corresponds to absolute freedom in navigation, with no system assistance / guidance whatsoever.

Category II: Includes the uncombined strategies C and D and corresponds to absolute freedom in navigation, but this time with explicit system assistance / guidance.

Category III: Includes all combinations of strategy E and corresponds to the application of restrictions on navigation, to enforce a path through the learning material.

The above categorization and ranking is, obviously, only one of several possibilities. It does, nevertheless, serve to demonstrate the following points:

– Although it is a ranking, it is not obvious in which "direction" it should be applied. For example, should the system start with the most "liberal" (in terms of navigation freedom) category and move to the more "restrictive" one when attempting to satisfy the overall adaptation design goal? Or should it apply the categories in reverse to accommodate, for instance, increased user familiarity with the system, or to compensate for inferred user frustration with the navigation constraints?
– Applying such a ranking incorporates two concepts that may need to be extricated and made explicit: the concept of the "default" category of strategies that might be applicable for a new user; and the concept of a "fallback" category that gets applied when none of the available categories / strategies has the desired effect.

For our example, we will assume that the designer has opted to use the ranking in the order presented above (i.e., "liberal" to "restrictive"), and to let the default and fallback categories be the first and last ones respectively. With these additional constraints, the system would be ready for a second round of user testing.

Iteration 3: Binding strategies to concrete adaptation logic

The introduction of additional structure in the adaptation design space effected in the previous iteration, along with more results from user testing based on that structure, can be expected to finally provide detailed enough results to start building more concrete and comprehensive adaptation logic around the alternative strategies.

According to results from related research in the literature, this iteration might result in user model-based adaptation logic along the following lines:

– For novice system users, as well as for users unfamiliar with the knowledge domain of the material, the more restrictive category (III) of strategies would be applicable.
– Within Category III, a ranking between strategies would be possible, such as: (i) strategy E – link hiding, no explicit recommendations by the system, (ii) combination D+E – link hiding and icons to "explain" the rationale between the provided

guidance, and (iii) combination C+E – same as previous case, but with less visual clutter on the page (intended for more experienced users).
- Category II (uncombined strategies C or D) would be reserved for users who seem to be sufficiently familiar with the system and the recommendation mechanism. If there is evidence of user confusion or ill effects of the user's increased navigation freedom (e.g., often-occurring negative test results for concepts encountered before their prerequisite concepts have been learned), the system should fall back to using Category III.
- Category I should be reserved for users who already have some familiarity with the knowledge domain, or exhibit behavior indicating intention to circumvent constraints applied on their navigation freedom (e.g., using the course outline to move between modules not interlinked).
- Etc.

The above adaptation logic is only exemplary in nature and might differ significantly from the actual results one might get with a specific system and learning material. It can, however, serve as a basis for the discussion in the forthcoming sections. Also note that, although the example case study is ending here, there is no reason why in real-world settings this would be the last design iteration. In fact, it is quite imaginable that the updated adaptation design might be put to the test again, to achieve even more refined adaptation logic, or more fine-grained adaptation strategies.

3 Adaptive system design revisited

The preceding section has put forward a scenario of how the adaptive behavior of a system could be designed, or "evolved" in a step-wise manner, taking advantage of meta-adaptive facilities. This section will fill in some of the intentional "gaps" in the presented scenario, formulating them as requirements posed by the introduction of meta-adaptivity. Before doing so though, let us first examine these facilities being utilized behind the scenes, and their effects on the design process.

3.1 New possibilities

To start with, the basis of the design iterations has been the derivation of new knowledge regarding the suitability of specific adaptive behaviors for different users (or contexts of use) given the overall design goal / self-regulation metrics. This knowledge, in its simplest form, is derived by applying alternative (combinations of) adaptation strategies, and assessing the extent to which the self-regulation metrics are satisfied, always in connection to the current user's model. Knowledge derivation, then, is achieved by analyzing all recorded cases where a particular strategy has had similar results, and identifying common user model attributes among the respective users. This is the core of the "learning" facilities in the context of self-regulated adaptivity, and their output could be expressed in various forms, including for instance as preliminary adaptation logic, intended to be reviewed, verified and incorporated into the systems by the designers. Although rather straightforward, the above step may

already suffice to provide valuable input to the design process. For example, it should be capable to identify strategies that are not suitable for any (category of) users, in any context of use. This was assumed to be the case in the elimination of strategy B and all its combinations in the previous section.

A second set of capabilities alluded to in the previous section is the categorization, or "clustering" of adaptation strategies, as well as their "ranking". Categorization can take place mainly along two dimensions: (a) The system can try to identify strategies that have similar effects with respect to the self-regulation metrics, given sufficiently similar user models; the output of this process would be a provisional clustering of strategies, based on their "cause and effect" patterns. (b) The system can try to identify the differentiating subsets of user models that render some strategies more effective than others. These dimensions give, respectively, two semantically rich measures of similarity and differentiation of adaptation strategies. When sufficient meta-data about the user model itself is available, the system can combine that with the measures to provide provisional rankings of alternatives within categories.

Before continuing it is important to note that the example in the previous section, as well as the analysis in this section, only assume three types of analytical assessment capabilities on the part of a self-regulating adaptive system. Although these are by far not the only ones possible, they are already adequate for the type of design support put forward in this paper.

3.2 New requirements

The fact that self-regulated adaptivity introduces new requirements becomes already evident in the first iteration of the design described above. Specifically, the first iteration made the following two fundamental assumptions:

- That adaptation strategies may be represented independently from the adaptation logic that drives them.
- That adaptation strategies, potentially expressed as sets of adaptation actions, be applicable in combination.
- That there may exist a representation of one or more adaptation "goals" that drive both the process of self-evaluation and the selection / application of strategies.

Of the above requirements, it might seem that the first two would be relatively easy to achieve in existing adaptive systems. This could be done, for example, respectively through the introduction of adaptation rules that randomly select one of a set of alternatives, and through the exhaustive representation of all possible combinations. Such an approach, however, would be incomplete in the sense that the adaptive system has no "understanding" of the alternative adaptation strategies, a fact which precludes the satisfaction of the third requirement, namely the employment of self-evaluation.

The second design iteration has also introduced additional requirements with respect to the adaptation engine. Specifically, it was implied that the system is capable of maintaining and employing a type of ranking amongst (combinations of) adaptation strategies, in a way that still does not associate the later with concrete, user model-based adaptation logic. Although perhaps inherent in the capacity to support such a ranking, the concepts of a "default" and a "fallback" strategy also merit individual mention. This is especially true in the case of the fallback strategy, which may

often be a mid-ranked alternative, rather than one at either extreme (this depends primarily on the factors taken into account to produce the ranking in the first place).

Finally, implied practically in all iterations, but made more explicit in the third one, is the fundamental requirement of the system being able to perform self-evaluation. As already mentioned, the concept of self-evaluation requires: (a) that the system be "aware" of the existence of alternative adaptive behaviors, and (b) has some way of assessing the extent to which these alternatives, when used, satisfy the design requirements. The first item points to the need for having an explicit representation of meta-data about the alternatives in the system, along with any semantic relations and constraints between them. The second item highlights the requirement that the "intention" behind an adaptive behavior (or set of behaviors) be expressed in a computable / measurable manner, so that the system can undertake the related assessment tasks.

"Armed" with these capabilities, the system can then proceed to analyze the commonalities / differences between the models of users for whom a particular strategy (or category of strategies) has proven successful / unsuccessful. This analysis would most likely be statistical in nature, and could draw upon well-established techniques in data mining. There is, however, a non-conventional requirement in this case: that when analyzing, the system must have access to prior states of a user's model. This is necessary because observations of the system as to the employment of specific (categories of) strategies for a given user, are inextricably linked to that user's individual model *at the time that the strategy was employed*. This is further obviated by the fact that appropriate strategies for users may change as their user model evolves over time. The re-stated requirement, then, is that the system must maintain a "history" of changes in the user models (what the history comprises and how its maintenance can be automated are discussed in the next section).

Finally, the example presented contains a simplification that we need to address. Specifically, in the example, the design iterations are linear and incremental, i.e., each iteration is based on the findings of the previous one, and adds more detail to the system's adaptation meta-data. In the real world, however, this may not always be the case. For instance, it might be necessary to consider entirely new strategies (e.g., adding link annotation through tooltips as an additional possibility) at later stages of the design. What this necessitates is that the system be able to function at two or more levels of granularity simultaneously, where each level represents a different degree of detail in the binding between adaptation logic and alternative adaptive behaviors.

3.3 Deus ex machina – can self-regulation be it?

There are two overarching questions that have still not been addressed: (a) Why introduce self-regulated adaptivity at all? Can't the above design activities be supported through an entirely human-facilitated approach? and, (b) How does one achieve the level of self-regulated adaptivity proposed in the preceding sections?

To start with, let us consider what meta-adaptivity, as implemented through self-regulation, brings to the "design table". Firstly, it allows us to specify and test with end users a potentially large number of alternative adaptive behaviors (or combina-

tions). A human-facilitated approach would require the generation of several instances of the adaptive system (in the worst case, one such instance would be required for each alternative / combination), as well as the overhead of administering tests involving real users for each case. Additionally, it is questionable whether such an approach would allow for the seamless transition between different adaptation strategies within one interaction session. The second benefit we derive from the employment of meta-adaptivity is that the generation of new adaptation knowledge (which, in turn, is subsequently expressed as adaptation logic to drive the system behavior) takes place *within* the system, making use of, and adding to, the adaptation meta-data. Since this process can be automated, it makes little sense to repeat the necessary analysis and encoding steps on a per-case basis. And last but not least, the employment of meta-adaptivity brings forth an entirely novel opportunity: a meta-adaptive system can continue improving itself (e.g., by accumulating adaptation knowledge / evidence, and using it to revisit its own adaptation logic) even without human intervention.

Let us now move over to the second of the questions posed at the beginning of this section, namely, how does one achieve the level of meta-adaptivity described herein. It is argued that self-regulation is sufficient for satisfying the requirements posed thus far. Since a detailed description of self-regulation is beyond the scope of this paper (interested readers may refer to [5]), we will simply focus on the basic premises of this particular type of meta-adaptivity.

The most important facets of self-regulated adaptivity are: (a) it explicitly accounts for alternative adaptive behaviors; (b) self-evaluation in the context of self-regulation is based on metrics that relate adaptive system behavior with changes in the user models (or, potentially, other dynamic models maintained by the system); (c) it entails apparent learning on the part of the system; and (d) it does *not* require that the system modify existing, or devise new behaviors on the fly [5]. Let us address each of the requirements identified in the previous section in more detail to examine the degree to which they are satisfied within the proposed approach.

Explicit representation of alternatives: This is one of the basic premises of self-regulated adaptivity. Specifically, a self-regulating system is capable of maintaining sets of behaviors that are applicable to different users / context of use, and dynamically switch between them on the basis of their "success".

Explicit representation of adaptation goals / objectives: Again, this is one of the fundamental building blocks of self-regulation. Representation in this case is achieved through the formulation of metrics, i.e., computable quantities derived from the current or historical values of attributes in the system's dynamic or static models (the user model being the primary source).

Supporting categorization and ranking of alternatives: Although not an intrinsic requirement for self-regulation, this capability should be relatively easy to implement in a compliant system. Categorization, for instance, can be supported through the application of the primary self-regulating capabilities at different levels of granularity. Ranking on the other hand can be implemented as an algorithm for selecting among alternatives within one level of granularity. Support for "default" and "fallback" strategies can be attained in a similar manner.

Self-evaluation and creation of adaptation knowledge: Again, self-regulation is "at home" with these requirements, since, even in its more basic incarnations, it involves the capability to on-the-fly assess the degree to which the currently employed alternatives satisfy the metrics encapsulating the design objectives, and take corrective actions as necessary. Furthermore, a self-regulating system can infer, over time, the applicability of behaviors to different (states of) user models and contexts of use. What's more, self-regulating systems can even validate the body of adaptation logic present and apply basic improvements over it without any human intervention (e.g., by not using adaptation rules that do not have the desired effects under any circumstances), or provide the evidence required for improvements to be made by humans.

Maintenance of "history" of model changes: As described in detail in [5], access to historical values of dynamic models is essential in enabling the learning part of a self-regulating system. What is interesting to note is that this requirement can be satisfied: (a) automatically, and (b) without maintaining a full record of all changes made to models. This of course implies that the system can "decide" what to place in the history. This information can be extracted, in the case of self-regulation, from the metrics used for self-evaluation. Specifically, provided that the system can "understand" the metrics used, it can identify all model attributes that are involved in any of the known metrics it handles, and record changes to them. This way, the history can be created in both an entirely automated way, and only cover those parts of a model that will be relevant to later learning processes.

4 Conclusions

This paper has presented a case for the use of meta-adaptivity as a facilitator in the design of adaptive systems. It has furthermore argued that the requirements introduced by the employment of meta-adaptivity can be met with only moderate complexity through self-regulation.

The applicability of the proposed approach is of course not universal: it requires, for example, that an adaptive system (or infrastructure) is already operational. It is also mainly intended for cases where there exist several alternative adaptive behaviors, with little or no empirical evidence as to their suitability for different categories of users, or different "states" of a single user.

Another interesting question that has not been addressed in this paper concerns the additional overhead that the proposed approach imposes on the designers / authors of adaptivity. Although the space available does not allow for a full treatment of the topic, it may be of value to cursorily outline the trade-offs: On the one hand, this approach requires that authors spend additional time in authoring adaptation strategies and providing the system with metadata for using and managing them; furthermore, it requires that designers review and, where necessary, validate the findings derived by the system through self-evaluation. On the other hand, some of the preceding activities are arguably ones that would need to be undertaken anyway in the context of iterative design cycles aimed at establishing and improving adaptive system behavior; an exception to this would be the metadata describing the adaptation strategies themselves, however the effort associated with them may be well justified by the

more active role the system can play in facilitating aspects of the evolutionary process.

In closing, it is important to briefly go over a few additional characteristics and constraints of the proposed approach. To start with, self-regulation is by no means a way to forego user studies, but rather an exploratory yet structured tool to employ in conducting them. Secondly, the approach, as put forward in this paper, is targeted to the design of adaptivity: abrupt transitions between potentially substantially different strategies might be unacceptable for a deployed system; as a result, in such settings, self-regulation may need to be further constrained (or limited to the accumulation of knowledge, but not permitted to effect changes in the system's adaptive behavior on the basis of the new knowledge). Thirdly, whether used solely in the design stage, or retained in the final system, self-regulation must be applied with care, as, by nature, it poses an even greater "threat" to traditional usability qualities of interactive systems (e.g., predictability) than traditional forms of adaptivity.

Within the confines discussed above, it is argued that self-regulated adaptivity represents not only the next logical step in the evolution of adaptive systems, but also a potentially irreplaceable tool in their design.

References

[1] Brusilovsky, P. (1996). Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction*, 6(2-3), 87-129.

[2] Cristea, A.I., & de Mooij, A. (2003). Adaptive Course Authoring: My Online Teacher. ICT'03 (International Conference on Telecommunications), Papeete, French Polynesia, IEEE,IEE, ISBN: 0-7803-7662-5, pp. 1762-1769.

[3] De Bra, P., Aerts, A., Smits, D., & Stash, N. (2002). AHA! The Next Generation. *ACM Conference on Hypertext and Hypermedia*, May 2002.

[4] De Bra, P., Stash, N., & Smits, D. (2005). Creating Adaptive Web-Based Applications, Tutorial at the 10th International Conference on User Modeling, Edinburgh, Scotland, July 25, 2005.

[5] Paramythis, A. (2004). Towards Self-Regulating Adaptive Systems. In Weibelzahl, S., & Henze, N. (Eds.), *Proceedings of the Annual Workshop of the SIG Adaptivity and User Modeling in Interactive Systems of the German Informatics Society (ABIS04)*, Berlin, October 4-5 (pp. 57-63).

[6] Paramythis, A., & Stephanidis, C. (2005). A Generic Adaptation Framework for Hypermedia Systems. In Chen, S. Y., & Magoulas, G. D. (Eds.) *Adaptable and Adaptive Hypermedia Systems* (pp. 80-103). Idea Group, Inc.

[7] Stephanidis, C., Paramythis, A., Zarikas, V., & Savidis, A. (2004). The PALIO Framework for Adaptive Information Services. In A. Seffah & H. Javahery (Eds.), Multiple User Interfaces: Cross-Platform Applications and Context-Aware Interfaces (pp. 69-92). Chichester, UK: John Wiley & Sons, Ltd.

[8] Totterdell, P., and Rautenbach. Adaptation as a Problem of Design. In Browne, D., Totterdell, P., and Norman, M. (eds.): Adaptive User Interfaces, pages 61-84. Academic Press, 1990.

[9] Weber, G., Kuhl, H.-C., & Weibelzahl, S. (2001). Developing adaptive internet based courses with the authoring system NetCoach. In S. Reich, M. Tzagarakis, and P. de Bra (Eds.), *Hypermedia: Openness, Structural Awareness, and Adaptivity*, (pp.226-238), Berlin, 2001.