User-Centric Content Negotiation for Effective Adaptation Service in Mobile Computing

Wai Yip Lum and Francis C.M. Lau, *Member, IEEE Computer Society*

Abstract—We address the challenges of building a good content adaptation service for mobile devices and propose a decision engine that is user-centric with QoS awareness, which can automatically negotiate for the appropriate adaptation decision to use in the synthesis of an optimal adapted version. The QoS-sensitive approach complements the lossy nature of the transcoding operations. The decision engine will look for the best trade off among various parameters in order to reduce the loss of quality in various domains. Quantitative methods are suggested to measure the QoS of the content versions in various quality domains. Based on the particular user perception and other contextual information on the client capability, the network connection, and the requested content, the proposed negotiation algorithm will determine a content version with a good aggregate score. We have built a prototype document adaptation system for PDF documents to demonstrate the viability of our approach.

Index Terms—Content adaptation, content negotiation, user preferences, decision engine, context awareness, mobile computing.

1 Introduction

Mobile computing being a new form of computing presents many challenges to hardware and software designers, one of them being that of mobile users using their small devices to view certain Web contents. There exists a huge gap between these small devices which are constrained in many ways and Web contents which were specially authored for normal Web viewing using a desktop computer. Added to this are the constraints of the wireless network through which these mobile devices are connected to the Internet. The limited bandwidth of the wireless network discourages sending of large volume of data. But, the reality is that the demand for using mobile devices to access Web contents is increasing rapidly. Therefore, the challenge is to find an effective way to enable mobile users to view Web contents conveniently and meeting the criterion that the content being viewed maintains a certain level of fidelity.

Content adaptation is an approach to provide some automatic means to convert any existing content to a form (a version) suitable for rendering in a mobile device requesting for that content (Fig. 1). Content adaptation can be based on the target device alone, hence, a *device-centric* approach. Such an adaptation considers the capabilities of a device and produces a content version that is renderable by the device. With no other factors to consider, the system would most likely produce the highest quality version that is renderable. The problem however is that sometimes the user does not really need such a best-quality rendition, but rather a version that is just enough to convey the needed information. Since mobile users tend to have very different requirements than when they are desk-bound in their office, we consider their personal preferences to be of utmost importance when deciding on what version to produce through content adaptation. Hence, we adopt a *user-centric* approach whereby we let user preferences rule. The goal is to generate an adapted version so that the version itself and its rendering process are a best match to the user's preference, and therefore most satisfying to the user.

To practically realize a user-centric design, we need to explore the concept of *quality of service* (QoS). According to [19], QoS is a user-oriented property. Hence, it should be possible for a user to provide or update his presentation preferences as part of the delivery context [6]. As discussed in [9], QoS can be viewed as a "*collective effect* of service performance which determines the *degree of satisfaction of a user* of the service." In our design, we adopted this type of meaning, using a score to quantify and measure *user satisfaction* along different quality axes, such as color, transmission time, etc. The score for a certain content then represents the *collective effect* of the quality value of the content in various quality domains weighted by user perception.

The construction of the system involves the following key design decisions.

- The representation of the user's preference in a structure that enables fast access to preference information.
- The decision logic to select the appropriate trade off among the various quality dimensions corresponding to a content version that best matches the user's preference and is deemed renderable by the device.
- The content realization mechanism to produce the desired content version as proposed by the decision logic.

The system functions as an intermediary proxy server situated between the client and the content provider server.

- *The authors are with the Department of Computer Science and Information Systems, The University of Hong Kong, Pokfulam Road, Hong Kong. E-mail: {wylum, fcmlau}@csis.hku.hk.*

Manuscript received 31 Dec. 2002; revised 6 July 2003; accepted 5 Aug. 2003. Recommended for acceptance by M. Oivo and M. Morisio.
For information on obtaining reprints of this article, please send e-mail to: tse@computer.org, and reference IEEECS Log Number 118784.

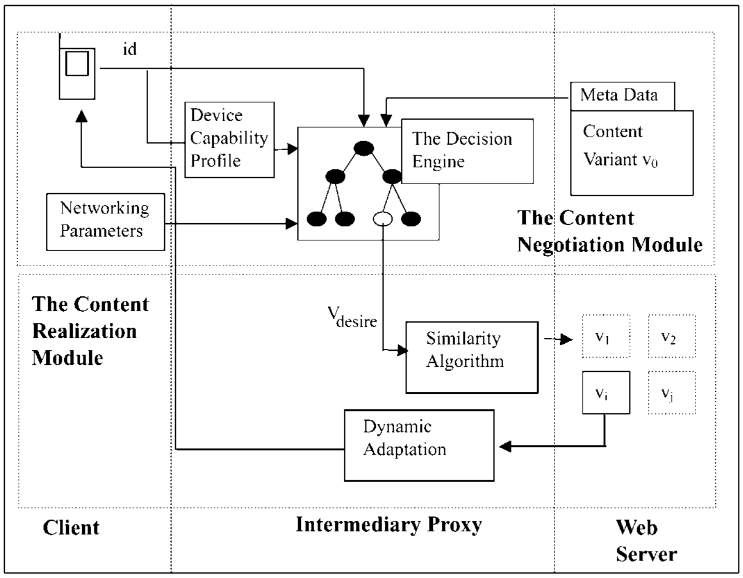


Fig. 1. The content adaptation framework.

There are two main modules in the system: the content negotiation module and the content realization module. The content negotiation module manages the users' preferences and operates a decision logic to derive the desired content version when being presented a request. The content realization module acts on the content version decision by the negotiation module to generate the real desired content version for rendering. This article discusses the design of the content negotiation module. The realization module is covered in [12].

The content negotiation module decides what is best for the user. There are many choices of combination of preference values along the quality dimensions, each of which is represented by a *score node*. The "score" is computed based on the preference values and the user's ranking of the dimensions. The module goes through an iterative process until arriving at the highest-scoring node that represents a renderable version. This process is seen abstractly as a negotiation between the user (represented by his specified preference) and the "context" which consists of the device, the network, and the document being requested—i.e., the user proposes a version and the context will respond with a yes or no. In other words, this is a *context-sensitive* approach in addition to user-centricity. This context-awareness issue has been covered in [11]. The decision engine responsible for the negotiation process and making the final decision can be hosted in an intermediary proxy server.

The content realization module [12] uses a selection logic to select a subset of "significant" content variants to be preadapted in the Web server in order to reduce the amount of on-the-fly transcoding overhead. A "similarity algorithm" in that module will determine the most suitable preadapted content version in the Web server from which the desired content version proposed by the negotiation module can be generated in real-time in the intermediary proxy.

2 Related Work

The principles of device-independence were addressed in [6], where, to achieve the independence, it should be possible for a user to obtain a functional presentation via any access mechanism. The access mechanism denotes the combination of hardware and software that allows a user to perceive a certain content and interact with the Web using one or more interaction modalities. This is easier said than done as there exists a wide variety of hardware and software idiosyncrasies. The work on *device-independent access* [2] offers some good insights on handling the variability of client devices which amounts to staying away as much as possible from creating content versions specifically for individual device types. The study in [7] follows the same vein and suggests that clients can be viewed as varying along three important dimensions: network, hardware, and software. In this article, we look further into the issue of client variability by focusing on the aspect of user perception. The concept of multiusers with multidevices was discussed in [8] which concentrated on an application scenario of content browsing in a lecture. Applying the same to the application scenario of content adaptation, we can create different views of content for different users according to their specific user preferences. Such a user-centric approach and the resulting content should lead to a higher degree of user satisfaction.

Content adaptation systems can make decisions on the response content versions or adaptation strategies based on some situational context. Such a decision can involve judgement based on some temporal-spatial trade off policy. This policy is to find an optimal trade off point representing a good balance between the two conflicting factors of delivery time and spatial size of the content, as can be found in [4], [14]. This judgement can also be derived using some scoring policy whereby the users and the publishers of the content can assign a score to different versions of the content so as to show their preferences on the decision of adaptation [16], [10], [22]. But, in real situations, it is rather

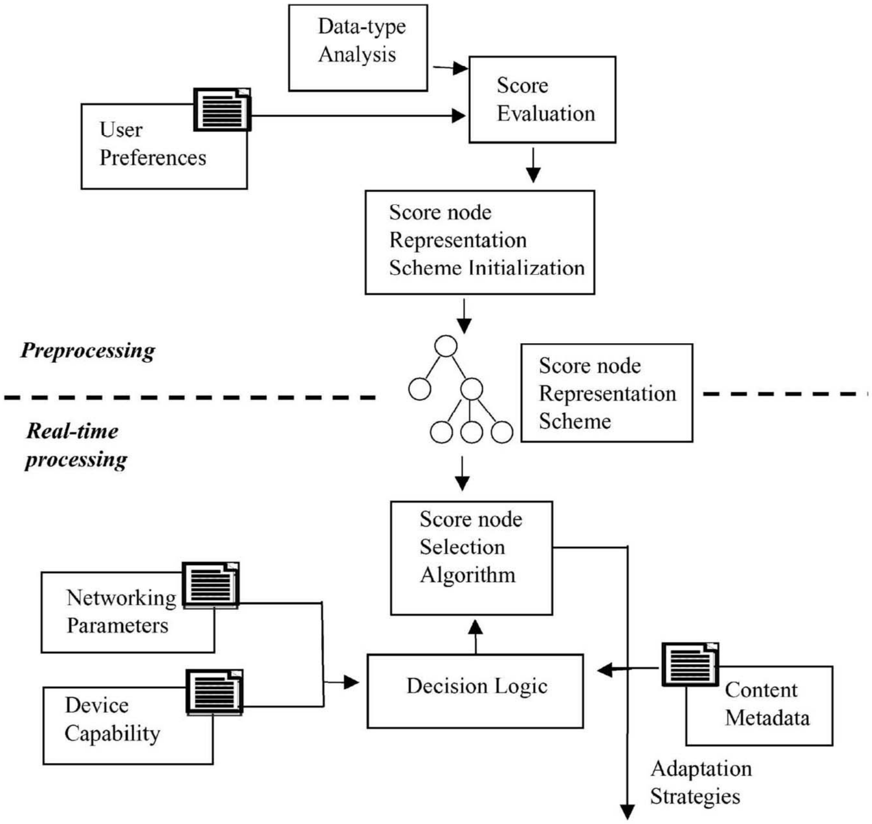


Fig. 2. The content negotiation module.

difficult for or a burden to the users or the publishers to have to assign a score to each content version without any guidance provided to them. To make this assignment automatic, the approach *resource-based content value* was proposed [16], [10], [22]. The content value is dependent upon the resources in the client device that can be used to render the content, not the degree of satisfaction perceived by a user.

In summary, we can see that contextualization can provide a good understanding of the various entities in the mobile computing environment. However, contextualization can provide only the declarative semantics but not the procedural ones such as the means of profile processing or interpretation and how to determine the adaptation strategy based on the detailed specification so as to instruct the technical transcoding procedures to produce the optimized content. Much work has also been done on the techniques employed in different modes of "lossy" [15] transcoding such as compression, color depth reduction or image scaling, etc., but little has been done to address the possibility of providing QoS-sensitive decisions to compensate for the lossy operations. There seems to exist a gap between the declarative specification of the client characteristics (for instance, CC/PP) and what can be achieved via the various techniques used in different transcoding methods. We propose in this article a negotiation module, at the core of which is a decision engine that would try to bridge this gap. The aim is to increase user satisfaction in subscribing to Internet contents in a constrained mobile computing environment. The proposed model can automatically negotiate for the appropriate content adaptation strategies to be used to generate an optimal version of content.

3 The Content Negotiation Module

The content negotiation module features a decision engine whose operation is to generate the necessary adaptation decision in order that the realization modules can synthesize the adapted version of content to the user's satisfaction. Fig. 2 gives an overall picture of the engine's operations. Along the temporal dimension, the module can be divided into the preprocessing stage which takes place before the arrival of the user request and the real-time processing stage when the request is under processing. During the preprocessing stage, the user preferences are captured and represented in the form of "score nodes" as shown in the figure. Then, in the real-time processing stage, the Score-node Selection Algorithm will be applied to search for the best score node taking into account the current real-time context. This article discusses the details of the computation of the score based on the user's perception in the QoS model. This perception provides a value judgement for the different content versions. We also discuss the construction of the special data structure used by the selection algorithm.

A set of quality axes can be identified for different types of multimedia content. This set of qualities forms the working properties that precisely defines the QoS for any multimedia content. The quality of a certain version of an object can be seen as a point in an n-dimensional space, where n is the number of different qualities. Take the QoS of a Portable Document Format (PDF) document as an example:

$$QoS_{pdf\text{-}document} = \\ (\text{color}, \text{scaling}, \text{segment}, \text{transmission-time}, \text{modality}),$$

where *modality* addresses the change in the presentation scheme of the content in order for the content to be rendered in different devices. For example, a PDF document can be presented in the original PDF format, but in view of the constrained bandwidth of the cellular network and the limited resources of the handheld device, it is advisable to convert the document to a representation that the device will be able to comfortably render, such as the format of the Wireless Markup Language (WML) which is supported by many WAP devices.

3.1 User-Centric Score Value

To design a score-based intelligent system, we need to define our own semantics, which is in terms of *score*. This involves numeric judgement of the content versions or adaptation strategies so that one can tell which content version is better than the others. We use numeric values to quantify this judgement. This is in contrast to the resource-based design in [16], [10], [22].

3.1.1 Quantifying Quality

To facilitate the expression and automatic processing of QoS parameters, we need a quantitative approach to characterizing QoS in any axis. We define a metric based on *quality value*(qv). Take the quality axis "color" as an example. An 8-bit color image is assigned a larger qv than that of an image with 1-bit black and white color. It is however not as straightforward to express or measure quantitatively the loss of qv in terms of colors. Also, different quality axes would have different QoS characteristics and such diversity makes capturing all the relevant characteristics quantitatively a nontrivial task. We propose the following model that can produce a satisfactory result in our architecture.

- The *quality value*(qv) is any value between 0 and 1; the higher the qv, the better the QoS.
- The *quantization step*(qs) defines the scale that is applicable to a particular quality axis. Take image as an example, the scale has 2 (colors), 16, 256, and so on, as quantization steps.

Next, we need the modeling functions that capture the behavior of qv against the variation of qs. Modeling function f is a function that captures the variation of qv against the variation of qs in a particular quality domain. It is denoted by $qv = f_i(qs)$ for the ith quality domain.

In the prototype system, we use *first order* and *second order* modeling to monitor the change of qv against qs. Users can input their desired modeling curve to the system if they have a particular perception towards the quality axis.

For the first order modeling, qv increases (or decreases) linearly with increase (or decrease) of qs and can be described as follows:

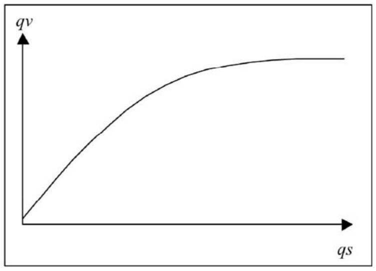


Fig. 3. The second order modeling curve.

$$qv = \frac{qs - qs_{min}}{qs_{max} - qs_{min}} \quad (\text{increasing}),$$

$$qv = \frac{qs - qs_{max}}{qs_{min} - qs_{max}} \quad (\text{decreasing}),$$

where qs_{max} is the maximum step possible in this quality domain, e.g., 100 percent scaling factor, and qs_{min} is the threshold step that can exist and convey a significant content value to the user, e.g., 1-2 percent scaling factor.

For the second order modeling, the modeling curve is characterized by the second order equation

$$qv = a \times qs^2 + b \times qs + c.$$

With this modeling, saturation is applied whereby qv would attain saturation in the far end of qs, as shown in Fig. 3. At or near saturation, qv will have an insignificant increase (or decrease) with a change in qs. Consider again the quality axis of color. When qs is at 16-bit colors (65,536 colors), the addition of more colors will have an insignificant impact on qv when compared to the initial cases where increments of colors near the value of two colors will have more impact on the perceived quality. We expect most user preferences will exhibit this type of saturation pattern. It can give a more concrete perception on QoS for some quality domains.

Other quality domains that do not have such a saturation behavior can be modeled using the first order equation. An example would be the quality axis of modality.

Consider the case where qv increases with qs and, so, we have three sets of equations in the model:

$$1 = a \times qs_{max}^2 + b \times qs_{max} + c,$$

$$0 = a \times qs_{min}^2 + b \times qs_{min} + c,$$

$$0 = 2a \times qs_{max} + b \quad (\text{saturation}).$$

By some mathematical manipulation of the three equations, we can obtain the result for the increasing case:

$$qv = \frac{qs^2 - 2qs_{max}qs - qs_{min}^2 + 2qs_{max}qs_{min}}{2qs_{max}qs_{min} - qs_{min}^2 - qs_{max}^2} \quad (\text{increasing}).$$

Similarly, we can model the QoS quantitatively for the case where the value of qv decreases with the value of qs with the following curve:

$$qv = \frac{qs^2 - 2qs_{max}qs + qs_{max}^2}{qs_{max}^2 - 2qs_{max}qs_{min} + qs_{min}^2} \quad (\text{decreasing}).$$

Together, these two models can capture most if not all of the behavior of the most common quality axes. If users have a particular quality model to a quality axis that these two models do not cover, they can input their desired modeling curves into the system. The behavior of the modeling curve is cleanly separable from the score evaluation process.

3.1.2 Score Evaluation

With the concept of quality axes as just described, the user can easily indicate his preferences. For example, a user may have a weak perception on color but a rather strong sensitivity on the difference in dimensional size. The user will rank qualitatively the quality axis of color to be lower than that of scaling. The user can input his specific perception (via ranking) regarding each quality axis, and the relative weight of each axis can then be calculated. An aggregate score can then be computed based on these weights:

$$w_i = \frac{rank_i}{\sum rank_j},$$

where $rank_i$ is the specific perception given by a user on the quality axis i.

Next, we can define an n-dimensional vector w for the weights to be assigned to each quality axis

$$w = (w_1, w_2, \ldots, w_{n-1}, w_{n)}.$$

Then, the *score* can be considered a dot product between the n-dimensional qv vector and the n-dimensional weight vector w in an n-dimensional vector space:

$$\begin{aligned} score &= qv \bullet w \\ &= \sum qv_i w_i. \end{aligned}$$

Finally, we can obtain

$$score = \sum f_i(qs_i) w_i.$$

The value of score provides the numeric judgement of the content version or the corresponding transcoding strategies. In creating the knowledge base for the decision engine, the user may not be familiar with expressing his preference in terms of quantitative information. But, on the other hand, it is much easier to provide qualitative information for one quality domain over the other. In the latter approach, they merely specify their preference without exact quantification and the qualitative information can be described by a preference relation that defines the relationship between different quality domains. It is the view of many researchers that human frequently reasons in qualitative rather than quantitative terms [3].

3.2 Score Node Representation

Scores corresponding to different versions need to be stored in some organized structure in order to facilitate efficient searching. This structure is either a linked list or a tree where each node represents a content version and stores the corresponding score. Apart from the score, a score node contains also the adaptation settings (the qs's) for possible subsequent generation of this version of content.

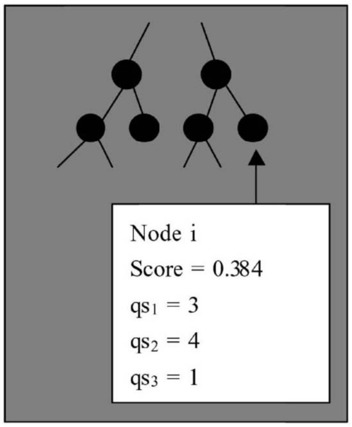


Fig. 4. A score node.

Fig. 4 gives an example of a score node. The user has given weightings of 4, 1, and 5 to three quality domains, respectively. Then, we have $w = (0.4, 0.1, 0.5)$. Consider the node i in the tree, as shown in the figure. The quality values of this node in the three quality domains are $(f_1(3)\ ,\ f_2(4)\ ,\ f_3(1))$ where $f(\ )$ returns the qv value given a qs value as discussed above. Assume the qv values are (0.42 , 0.51 , 0.33); the score of this node is therefore equal to

$$score = 0.42 \times 0.4 + 0.51 \times 0.1 + 0.33 \times 0.5 = 0.384.$$

A search space consisting of all possible score nodes is created at initialization time, which covers all the possible adaptation decisions that the decision engine can make. For example, in our prototype (discussed in Section 4), there are five quality domains for which the user can specify their preference, and, for each, there are four possible adaptation choices (i.e., four qs and qv values). Thus, the resulting search space has 1,024 score nodes. The score to be assigned to each score node is computed at preprocessing time when the user signs in to register and specify his preferences. This can be done at preprocessing time because of the static nature of the score-version association—that is, we expect that the user would rarely change his preference for the various quality domains. The score node space so established is preprocessed into a suitable data structure such that, when a request is received, heuristic search in real-time can be performed effectively and efficiently.

The score nodes can be arranged in a linked list with the parameter score as the key value. To determine the optimal version of content with the highest score value constrained by the rendering requirement, heuristic search ("negotiation algorithm") is applied over the score linked list. Balanced binary tree is another, alternative data structure. Fig. 5 shows a score tree. All the scores in the left subtree are smaller than that of the parent node at the top, and all the scores in the right subtree are larger than or equal to that of the parent.

3.3 Decision Logic

The score nodes for a user capture all the possible combinations of preference values in various quality

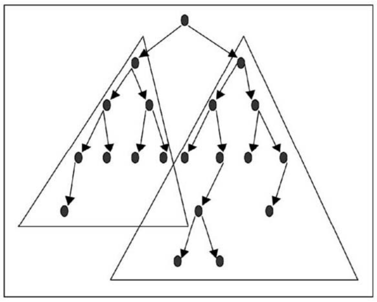


Fig. 5. A typical score tree.

domains. They do not include values of real-time parameters such as the characteristics (metadata) of the Web object being requested, the characteristics of the network connection, and the device capability. The goal of the operation of the decision engine is to find the best scoring node corresponding to a version of the content that is renderable given those real-time parameters. During the process to locate the optimal node, for each score node to be examined, the decision engine will generate a binary decision (True or False) based on the client device capability, the network parameters, the adaptation settings stored in the score node, and the content itself:

$$T \mid\mid F \ \leftarrow decision(\mathrm{score-node}, P_d, P_n, P_c),$$

where *score-node* provides the setting of various quality axes. The binary decision True indicates that the content after transcoding according to the adaptation settings as specified in this score node is renderable in the target device with its particular capability in the current network environment. The binary decision False otherwise. The decision function, $decision()$, interacts with the score node data structure iteratively in a negotiation fashion until a satisfactory score node with a True decision value is found.

The decision logic may operate with different strategies in different application scenarios. In some scenarios, we can identify a metric with the property of *ordered relation*, based on which we can make binary decisions. For example, "Resource" is a metric that is based on an ordered relation, and so the decision logic can operate as follows:

```
if $Resource_i > Resource_{device}$
then
    return False
else
    return True
```

$Resource_i$ is the resource level of the score node i computed using parameters such as the adaptation settings stored in the score node, the network parameters, and the content metadata. We can save some effort during searching for the optimal score node using the observation that whenever the decision corresponding to $Resource_i$ returns False, the score nodes with $Resource$ larger than $Resource_i$ should also return False. Therefore, all the score nodes with $Resource$ larger than $Resource_i$ need not be visited, leading to the saving of a considerable amount of execution overhead when using binary search or a binary search tree. To further illustrate the idea, we can explicitly define the following as part of the decision logic and the logic will return True if and only if

1. content size $\leq$ device memory buffer size,
2. content dimension $\leq$ acceptable device screen dimension,
3. color depth of the content $\leq$ acceptable color depth of the device, and
4. $2D_{RTT} + \frac{Spatial\ size\ of\ the\ content}{Bandwidth\ of\ the\ channel} \leq t_{threshold}$,

where $2D_{RTT}$ is the network round trip time and $t_{threshold}$ is the maximum transmission time the user can tolerate for the current session.

To identify a metric with the ordered relation property, the metric must satisfy a number of criteria. That is, a binary decision function *f* on a score node set is an ordered relation if and only if it is reflexive, asymmetric, and transitive. In some cases, however, it is difficult to identify a metric with the property of ordered relation. Take the following decision logic as an example, where the decision logic should return True if and only if

1. the modality specified in the score node can be supported by the device,
2. the frame rate specified in the score node can be supported by the device,
3. the color depth specified in the score node can be supported by the device, and
4. the resolution specified in the score node can be supported by the device.

Since there exist devices that require exactly some specific values of color depth (instead of a range) in order to operate, and similarly for resolution and frame rate, we cannot devise an ordered relation for the above decision logic in any way for us to apply binary search. For such a case, one might have to visit all the score nodes.

3.4 Score Node Selection Algorithm

The proposed decision engine runs a "negotiation" process between the data structure component containing the user's preference information and a decision function of the decision engine, as shown in Fig. 6. The negotiation process entails a systematic traversal of score nodes by a negotiation algorithm. This is an iterative heuristic search to try to find the best score that meets the rendering constraint. The result is the most optimal score node found by the algorithm. The adaptation strategy as contained in the node will be fed to the realization processes to synthesize the actual version of the adapted content.

We propose four negotiation algorithms to suit different types of scenarios.

- Score Linked List (SLL) Negotiation Algorithm: We apply heuristic search on the linked list until the decision logic can yield True. This is an $O(n)$ algorithm.
- Ordered Relation Score Tree (ORST) Negotiation Algorithm: A balanced binary tree is used whereby an ordered relation can be identified in the decision logic. The comparison logic in ordered relation can facilitate the operation of the binary search tree, resulting in an $O(lg\ n)$ algorithm. We will illustrate

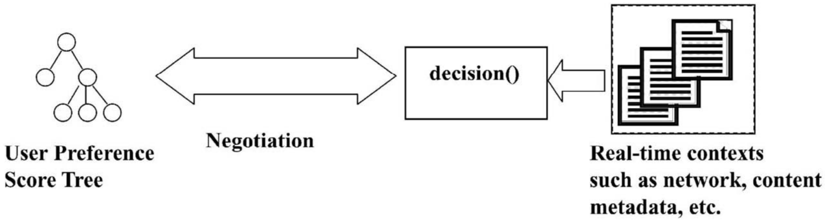


Fig. 6. The negotiation process.

the ordered relation of the decision logic with the PDF document adaptation system later.

- Nonordered Relation Score Tree (NORST) Negotiation Algorithm: When no ordered relation can be identified in the decision logic, we need to visit every node in the listed list in the worst case. We use this probabilistic score tree algorithm that incurs less overhead and can "likely" return a good if not the best score node. Again, this is an $O(lg\, n)$ algorithm. The *optimization accuracy* (of finding the optimal score node) for the NORST Negotiation Algorithm can be proved to be bounded by

$$A_o(p_t) \geq 1-(1-p_t)\left[\frac{(1-p_t)^{\frac{1}{p_t}}+(1-p_t)}{lg(n_{total}+1)} + \frac{2lg(\frac{1}{p_t}+1)(1-(1-p_t)^{\frac{1}{p_t}})}{lg(n_{total}+1)}\right],$$

where n_{total} is the total number of score nodes and p_t is the average probability that the nodes in the subtree will return True at the decision logic.

Fig. 7 shows the plot of the optimization accuracy based on the above equation for different sizes of the score tree, where n is the number of nodes in the tree. As expected, the optimization accuracy increases with the value of p_t. It should be noted that even when p_t is small (e.g., 0.3), the optimization accuracy still lingers around the area of 70 percent.

- Score Linked List-Non-Ordered Relation Score Tree (SLL-NORST) Negotiation Algorithm: We can define a threshold for the optimization accuracy, for example 70 percent, such that when the accuracy level of the NORST algorithm falls below this threshold value, the system will automatically switch to the SLL algorithm. This will guarantee that the accuracy of resulting optimization will be bounded by this threshold value. This is the algorithm used in our prototype system, to be discussed in the next section.

Further details on these negotiation algorithms can be found in [13].

4 PDF Document Content Adaptation System

We have implemented an intermediary proxy that hosts the decision engine as discussed in the previous sections in order to demonstrate the practical viability of our approach.

Our PDF Document Content Adaptation System (PDCAS) is aware of the user context in five quality domains: color, transmission time, scaling, modality, and segment. The modality domain lets the user specify how he feels about preserving the mode versus transcoding of the original content to a different mode. There are four values in the scaling domain, corresponding to the output format of WML, HTML, bitmap, and PDF, respectively. The segment domain corresponds to how the user feels about cropping of the content, with four different cropping ratios for the user to choose from. The user can also specify the

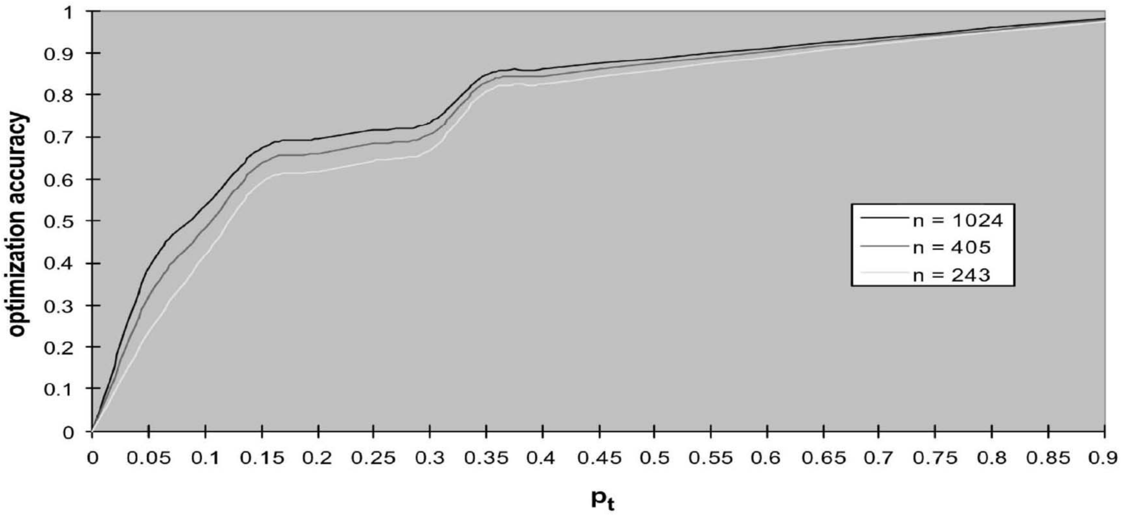


Fig. 7. The plot of optimization accuracy against p_t.

Fig. 8. Some commercial portable device models: (a) Nokia 6210, (b) Ericsson R320, (c) iPAQ H3760, and (d) PALM m515.

TABLE 1
Some Device Profiles

	Nokia 6210	Ericsson R320
Supported Modailties	WML,WBMP	WML,WBMP, GIF
No. of colors	b/w 2 colors	b/w WBMP, 3-color GIF
Max. object size	1.397KB	3KB for deck, 1.5KB for image
Screen size	90×60 Pixels	101×52 Pixels
	iPAQ H3760	PALM m515
Supported Modalities	HTML, BMP, GIF, JPEG	Depends on viewer
No. of colors	4096 colors	65536 colors
Max. object size	100K	2000K
Screen size	240×320 Pixels	160×160 Pixels

maximum transmission time $t_{threshold}$ he can tolerate for the content delivery.

For the context of device capability, we can use information derived from some commercial portable device models (Fig. 8), and the system will take into account the screen size, the supported number of colors, media type(s), markup language(s), and the memory buffer size (e.g., the maximum deck size acceptable by a WAP phone) as shown in Table 1.

In general, the user can set a limit on the file size of the Web objects received in the browser for both the Pocket PC and the Palm. The setting we use here is the default maximum channel size for Avantgo [1]. For Ericsson R320, the dimension (width x height) of the GIF image cannot exceed 12,000 pixels.

For the network context, a set of parameters like bandwidth and round-trip time of some currently popular communication channels, e.g., CDMA, GPRS and CDPD, etc., can be applied to our system so that the system is made aware of the networking context. Here, 300 ms round trip time [23] is used for the calculation.

By supplying different values to the parameters described above, the Web browser or a WAP device simulator (we use the Nokia 6210 simulator [17]) can emulate the operation of any type of client device, even a fictitious one. In practice, techniques for automatically discovery of the client device type (via for example HTTP headers) and networking characteristics as well as some means of client identification (explicit userid or cookies) would be employed in order to generate all the necessary context information. In our system, we use userid embedding in URL [21] to identify the end user.

4.1 The Score Nodes

In our prototype, there are five quality domains for the user to specify a value, and for each there are four possible values for the quantization steps—e.g., for color, we have 2, 4, 16, and 256 colors; for scale, we have 100, 75, 50, and 25 percent scaling factors; and for time, we have 1.0, 0.75, 0.5, and 0.25 of $t_{threshold}$. According to the relationship between the quantization step and the quality value (i.e., the modeling function), the corresponding quality value between 0 and 1 can be found. The summation of these quality values weighted by the specific user-preference for different quality domains will give the aggregate score value. Each score node will hold the specific values for the five quality domains, the corresponding single aggregate score, and the various reference pointers as required by the data structure, be it a linked list or a tree. We have $4^5 = 1,024$ score nodes in the data structure.

4.2 The Decision Logic

According to the parameters discussed above, we can explicitly define the following decision logic in our prototype and the logic will return True if and only if

- content size $\leq$ device memory buffer size (as found in device capability), where the content size is calculated by file size per page (in content metadata) $\times qs_{scaling} \times qs_{segment}$;
- content dimension $\leq$ acceptable device screen dimension (as found in device capability), where the content dimension is calculated by document dimension (in content metadata) $\times qs_{scaling} \times qs_{segment}$;

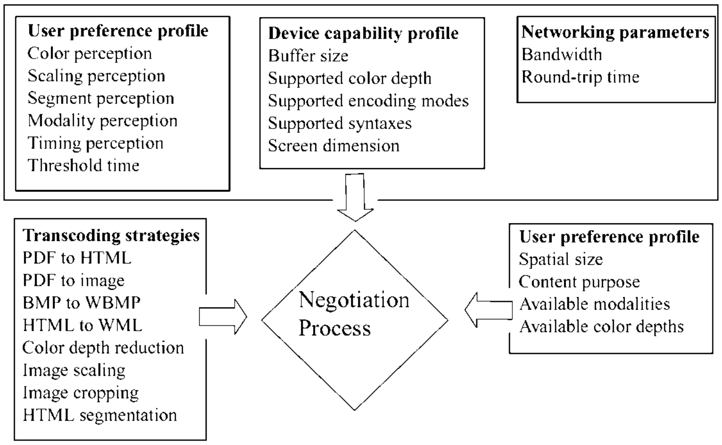


Fig. 9. The PDF document adaptation system.

- $2D_{RTT} + \frac{Spatial\ size\ of\ the\ content}{Bandwidth\ of\ the\ channel} \leq t_{threhold}$, where $2D_{RTT}$ is the network round-trip time and $t_{threshold}$ is the maximum transmission time the user can tolerate for the current session;
- color depth of the content in content metadata $\leq$ supported number of colors for the device (as found in device capability); and
- the modality of the content is supportable by the device.

Note that, the first four are ordered-relation logic and the last one is nonordered and, so, the combined logic is a nonordered one. We therefore employ the SLL-NORST Negotiation Algorithm in our system.

After making the decision on the optimal content version with the negotiation algorithm, various transcoding techniques are employed in order to synthesize the desired content. The technical transcoding techniques employed are summarized in Fig. 9.

4.3 Experimental Results

Several experiments were conducted with an aim to demonstrate the practical usefulness of the prototype in various application scenarios and the viability of the proposed approach. These experiments also show the sensitivity and awareness of the negotiation model towards different situational or user-centric contexts. The graphical manipulation operations such as image modal transcodings involving PDF-to-image conversion, color depth reduction, and image scaling are performed by ImageMagick 5.3.8. The conversion of PDF to HTML is performed by pdftohtml Converter 0.21. We have implemented a simple HTML-to-WML transcoding module that can convert HTML to WML. For all the cases, the actual time spent in executing the algorithm to arrive at a transcoding decision is very small, which is negligible when compared to the transcoding and transmission times.

4.3.1 Delivery Time versus Modality Preservation

Fig. 10 shows some sample results of the same PDF document being delivered to a WAP device and a PDA using our system. The difference in presentation is due to the user's perception on two conflicting factors, the transmission time and the modality preservation. For Fig. 10a, the user has expressed a higher preference on preserving the original modality over that of the transmission time, whereas, in Fig. 10b, the user has indicated the opposite and, so, an image (WBMP) and a text (WML) version would result respectively. For the case of the PDA, we tested the sensitivity of the negotiation algorithm by varying the trade off between the time and the modality quality while keeping the other factors constant, and the original content version in PDF, image (BMP) and text (HTML) would result accordingly, as shown in the figure.

4.3.2 Maximum Tolerable Transmission Time

The user can also specify the maximum transmission time he feels is tolerable and the system will negotiate the content version automatically to suit his particular requirement, as shown in Fig. 11. In order to meet this time requirement, the content may need to undergo a segmenting process—e.g., cropping the large HTML text to meet the

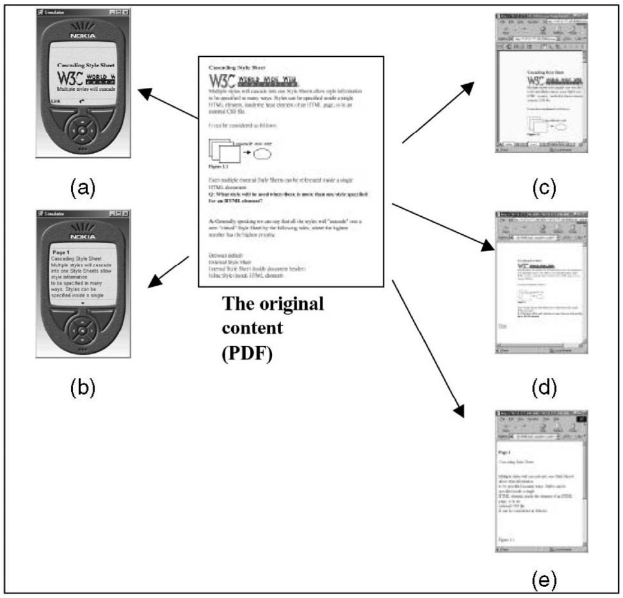
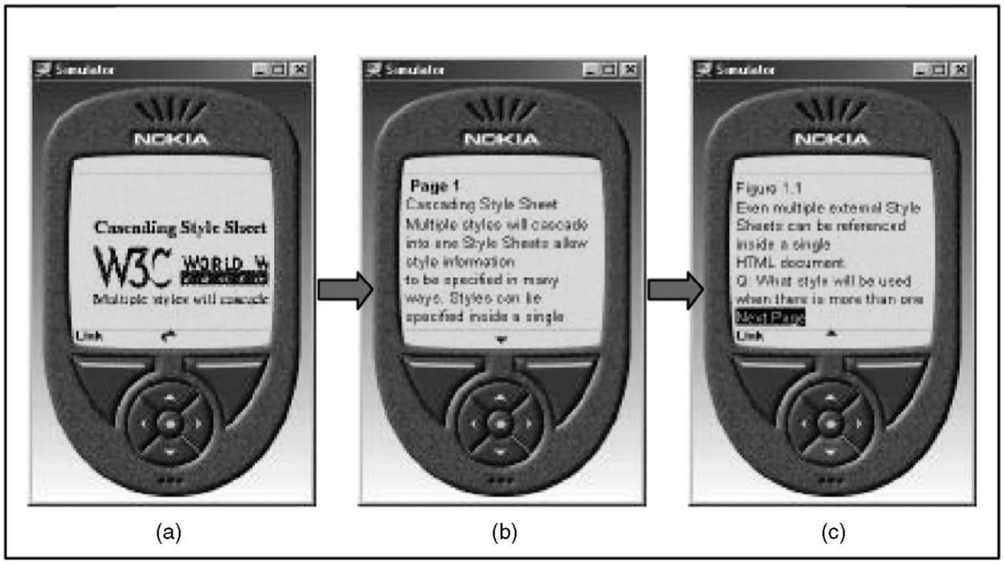


Fig. 10. The change of modality due to device capability and user preference: (a) WBMP, (b) WML, (c) PDF, (d) BMP, (e) HTML.



Cascading Style Sheet
W3C WORLD W
Multiple styles will cascade

Page 1
Cascading Style Sheet
Multiple styles will cascade
into one Style Sheets allow
style information
to be specified in many
ways. Styles can be
specified inside a single

Figure 1.1
Even multiple external Style
Sheets can be referenced
inside a single
HTML document.
Q: What style will be used
when there is more than one
Next Page

Fig. 11. Awareness of the change of maximum tolerable transmission time: (a) WBMP, (b) larger WML, (c) smaller WML.

timing requirement and leaving the residue to the next page and linked by the "next" anchor. There is a decrease in the value of the maximum tolerable transmission time as we go from left to right in the figure for the sample deliveries. For example, Fig. 11c corresponds to the user specifying 0.5 second of tolerable transmission time.

4.3.3 Color versus Time

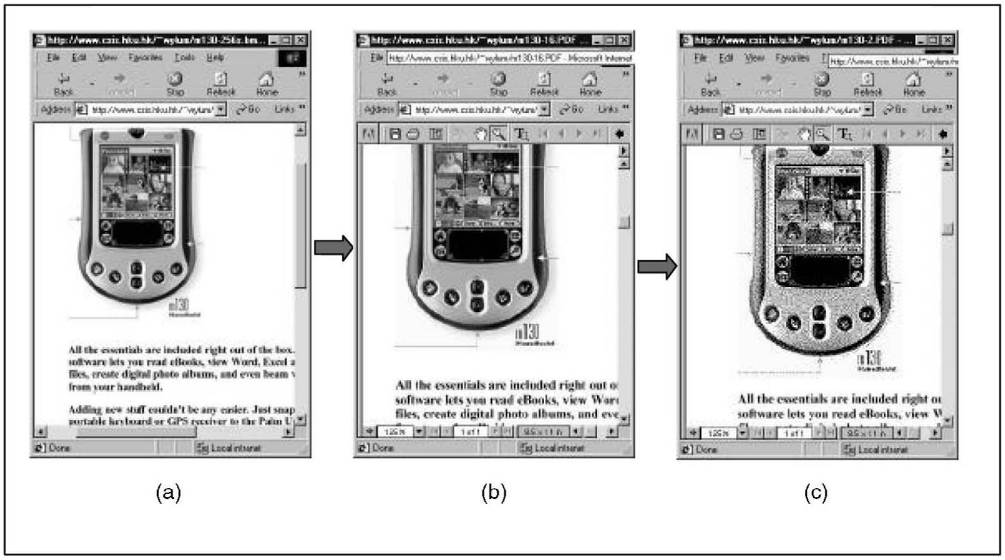
Fig. 12 demonstrates the effect of the negotiation algorithm trying to strike a balance for the trade off between the conflicting factors of color and the transmission time. With decreasing weights on the perception of color, the system would respond with content versions of 256, 16, and 2 colors for the PDF document accordingly in order to favor the transmission time in a slow network.

4.3.4 Networking Characteristics

The adaptability of the system over the networking characteristics is shown in Fig. 13. The other factors in the context are kept constant and we vary the bandwidth parameter to test the outcome from the system. The sample deliveries show that the system would switch modality in order to suit the current bandwidth of the connection so as to keep the transmission time within the allowed tolerance, as shown in Figs. 13a, 13b, and 13c of the figure.

4.3.5 Device Capability

Further experiments were performed on the memory buffer size of the device to see whether the negotiation algorithm can return an optimal content version automatically and accordingly in order to handle the heterogeneous nature of the devices. The results are quite similar to those described


http://www.csis.hku.hk/~wylum/m130-256c.bm...
http://www.csis.hku.hk/~wylum/m130-16.PDF ...
http://www.csis.hku.hk/~wylum/m130-2.PDF - ...
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home
Address http://www.csis.hku.hk/~wylum/ Go Links
m130 Handheld
125%
1 of 1
9.5 x 11 in
Done
Local intranet
(a)
(b)
(c)

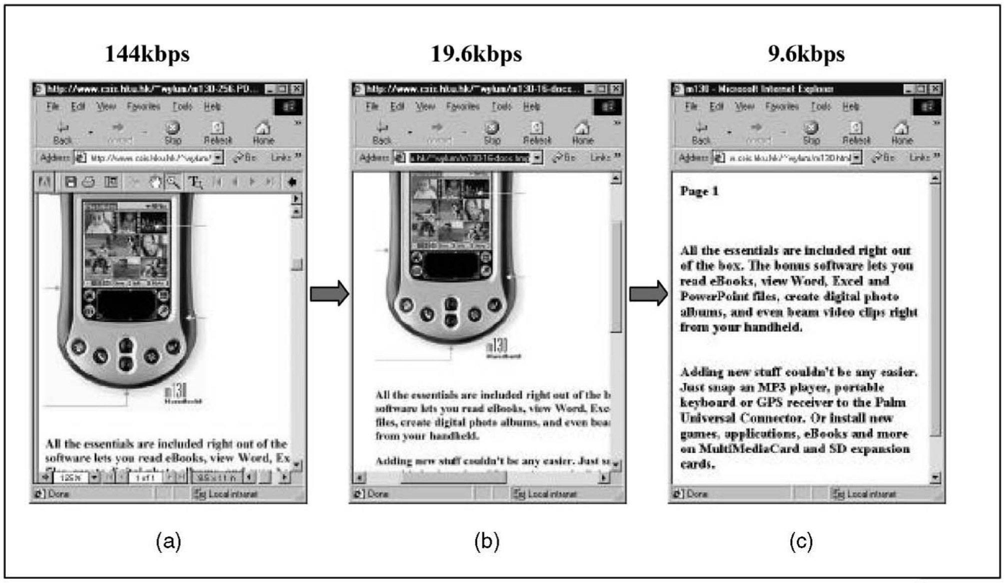
All the essentials are included right out of the box.
software lets you read eBooks, view Word, Excel a
files, create digital photo albums, and even beam v
from your handheld.

Adding new stuff couldn't be any easier. Just snap
portable keyboard or GPS receiver to the Palm U

All the essentials are included right out o
software lets you read eBooks, view Word
files, create digital photo albums, and eve

All the essentials are included right ou
software lets you read eBooks, view W

Fig. 12. Awareness of the color perception of the user: (a) BMP with 256 colors, (b) PDF with 16 colors, (c) PDF with 2 colors.



All the essentials are included right out of the
software lets you read eBooks, view Word, Ex

All the essentials are included right out of the b
software lets you read eBooks, view Word, Exc
files, create digital photo albums, and even bea
from your handheld.

Adding new stuff couldn't be any easier. Just s

Page 1

All the essentials are included right out of the box. The bonus software lets you read eBooks, view Word, Excel and PowerPoint files, create digital photo albums, and even beam video clips right from your handheld.

Adding new stuff couldn't be any easier. Just snap an MP3 player, portable keyboard or GPS receiver to the Palm Universal Connector. Or install new games, applications, eBooks and more on MultiMediaCard and SD expansion cards.

Fig. 13. Adaptability of the system against the networking context: (a) PDF with 256 colors, (b) BMP with 16 colors, (c) HTML.

here for the networking context, with outcomes closely agreeing with our expectation.

5 Conclusion

This article presents a QoS-sensitive decision engine that can determine the optimal adaptation decisions from interpolation of situational context information including user preference, device capability, and network characteristics. In the center of the decision making process is a negotiation algorithm that considers all possible versions of content in order to arrive at an optimal renderable version. The operation of the algorithm makes up for the lossy nature of the transcoding processes and finds a trade off among qualities in various quality domains so that the ensuing transcoding processes can be performed satisfactorily. We have presented a summary of the experimental results that confirm the viability of our approach.

In this work, we assume that user preferences and the adaptation process have no dependence on what is in the content. In another ongoing project, we try to perform content analysis of Web documents to discover meaningful components and their relationships, and the result could lead to more effective adaptations. In such a design, the final content to be generated will be sensitive to changes in the user preference as well as the content.

The decisions that are generated as output from the negotiation algorithm are used to drive a number of transcoding operations to synthesize the desired content. Therefore, the weaker the tie between the adaptation decisions and the content-generation processes the better would be the quality of the returned content. In other words, the optimization accuracy to a certain extent depends on this coupling between the algorithm and the content-generation processes. In our design, we have tried to separate the two as much as possible.

The negotiation module can be further extended such that transcoding could be applied to the "tasks" a user wishes to perform on a device rather than just contents. Also, the document model employed in our prototype with quality domains like colors, dimension, etc. can be extended to cover images so that histograms, segmentations, etc. could be added to the collection of quality domains.

In a related project [12], we study the trade off between dynamic and static adaptation. The production of the content as suggested by the decision engine can take advantage of static preadaptation (thus, sacrificing I/O cost for the CPU cost), or the approach of dynamic real-time adaptation (sacrificing CPU cost for I/O cost). Some mixed approach can be proposed to create a good cooperative environment between these two modes so as to find the best cost-effective methodology to synthesize contents with guidance from the decision engine.

Acknowledgments

This research is supported in part by a Hong Kong RGC grant (HKU 7519/03E).

References

[1] Avantgo.com, http://www.avantgo.com, 2003.

[2] T.W. Bickmore and B.N. Schilit, "Digestor: Device-Independent Access to the World Wide Web," *Proc. Sixth World Wide Web Conf. (WWW6)*, pp. 655-663, Apr. 1997.

[3] *Qualitative Reasoning about Physical Systems.* D.G. Bobrow, ed., MIT Press, 1985.

[4] J. Chen, Y. Yang, and H. Zhang, "An Adaptive Web Content Delivery System," *Proc. Int'l Conf. Adaptive Hypermedia and Adaptive Web-Based Systems (AH2000)*, Aug. 2000.

[5] R. Comerford, "Handhelds Duke It Out for the Internet," *IEEE Spectrum*, pp. 35-41, Aug. 2000.

[6] Device Independence Principle, W3C working draft, Sept. 2001, http://www.w3.org/TR/2001/WD-di-princ-20010918/.

[7] A. Fox and E.A. Brewer, "Reducing WWW Latency and Bandwidth Requirements by Real-Time Distillation," *Proc. Fifth Int'l World Wide Web Conf. (WWW5)*, May 1996.

[8] R. Han, V. Perret, and M. Naghshineh, "WebSplitter: A Unified XML Framework for Multi-Device Collaborative Web Browsing," *Computer Supported Cooperative Work 2000 (CSCW 2000)*, pp. 221-230, 2000.

[9] "ITU-T Recommendation I. 350: General Aspects of Quality of Service and Network Performance in Digital Networks, Including ISDNs," *ITU Telecomm. Standardization Sector (ITU-T),* Mar. 1993.

[10] C.-S. Li, R. Mohan, and J.R. Smith, "Multimedia Content Description in the InfoPyramid," *Proc. IEEE Int'l Conf. Acoustics Speech and Signal Processing (ICASSP 98),* May 1998.

[11] W.Y. Lum and F.C.M. Lau, "A Context-Aware Decision Engine for Content Adaptation," *IEEE Pervasive Computing,* vol. 1, no. 3, pp. 41-49, July-Sept. 2002.

[12] W.Y. Lum and F.C.M. Lau, "On Balancing Between Transcoding Overhead and Spatial Consumption in Content Adaptation," *Proc. Mobicom 2002 Conf.,* pp. 239-250, Sept. 2002.

[13] W.Y. Lum, "Effective Content Adaptation Strategies for Mobile Computing," master's thesis, Dept. Computer Science and Information Systems, The Univ. of Hong Kong, HKSAR, 2002, http://www.csis.hku.hk/char126wylum/thesis2.pdf.

[14] W.Y. Ma, et. al., "A Framework for Adaptive Content Delivery in Heterogeneous Network Environments," *Proc. Conf. Multimedia Computing and Networking,* pp. 86-100, Jan. 2000.

[15] J.C. Mogul, "Server-Directed Transcoding," *Computer Comm.,* vol. 24, no. 2, pp. 155-162, Feb. 2001.

[16] R. Mohan, J.R. Smith, and C.-S. Li, "Adapting Multimedia Internet Content for Universal Access," *IEEE Trans. Multimedia,* vol. 1, no. 1, Mar. 1999.

[17] Nokia Mobile Internet Toolkit, http://www.nokia.com, 2003.

[18] T.L. Pham, G. Schneider, S. Goose, and A. Pizano, "Composite Devices Computing Environment: A Framework for Situated Interaction Using Small Screen Devices," *Personal and Ubiquitous Computing,* vol. 5, no. 1, pp. 25-28, 2001.

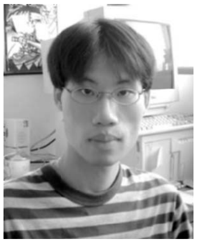
[19] J.-P. Richter and H. de Meer, "Towards Formal Semantics for QoS Support," *Proc. 17th Ann. Joint Conf. IEEE Computer and Comm. Societies (INFOCOM '98),* vol. 2, pp. 472-479, 1998.

[20] S. Saha, M. Jamtgaard, and J. Villasenor, "Bringing the Wireless Internet to Mobile Devices," *Computer,* vol. 34, no. 6, pp. 54-58, June 2001.

[21] T. Shimada, N. Iwami, T. Tomokane, M. Hayashi, and Y. Kuwahara, "Interactive Scaling Control Mechanism for World-Wide Web Systems," *Proc. Sixth World Wide Web Conf. (WWW6),* pp. 1467-1477, 1997.

[22] J.R. Smith, R. Mohan, and C.-S. Li, "Transcoding Internet Content for Heterogeneous Client Devices," *Proc. IEEE Int'l Symp. Circuits and Systems (ISCAS),* May 1998.

[23] D. Stuard, "Voice Quality in PCS and Cellular Networks: Solutions for Echo," *Mobile Radio Technology,* Sept. 1998.



Wai Yip Lum received the Beng degree in computer engineering and the MPhil degree in computer science from The University of Hong Kong. He is a research assistant in The University of Hong Kong's Computer Science and Information Systems Department. His research interest is in the design of content adaptation services for mobile and pervasive computing.

Francis C.M. Lau received the PhD degree in computer science from Waterloo in 1986. He joined the Department of Computer Science and Information Systems at the University of Hong Kong, as a lecturer in 1987, and was appointed and started serving as the head of the department in 2000. Dr. Lau is an active member of the IEEE Computer Society. He pioneered the society's Distinguished Visitors Program for Asia/Pacific in 1993. He served on the society's board of governors in 1998, as a member-at-large of the Publications Board, and then as vice president for chapters activities in 1999. He was a member of the society's Technical Activities Board, in charge of Asian technical activities, and was a core member of the ACM-IEEE/CS Joint Task Force on Curriculum 2001. He received a Golden Core recognition in 1998 and an IEEE Third Millennium Medal in 2000 for outstanding achievements and contributions to the IEEE Computer Society. Dr. Lau's research interests include parallel and distributed computing, mobile and pervasive computing, computer music, Internet, and the Web. He has written many journal and conference papers, and is a coauthor (with C.-Z. Xu) of the book, *Load Balancing in Parallel Computers* (Kluwer Academic Press, 1997).

▷ **For more information on this or any computing topic, please visit our Digital Library at** http://computer.org/publications/dlib.