Realization of distributed content adaptation with service-based approach for pervasive systems

Girma Berhe, Lionel Brunie, Jean-Marc Pierson

Lab. d'InfoRmatique en Images et Systèmes d'information (LIRIS) INSA de LYON

{girma.berhe, Lionel.Brunie, Jean-Marc.Pierson}@insa-lyon.fr

Abstract

Pervasive computing applications allow users to access information from anywhere while traveling and using variety of devices. Heterogeneity and limitation of resources involved in this application demand adaptation of content according to the current context (device, user, network etc.). The trend of publishing different service on the Internet with emerging technologies such as Web Service create an opportunity to distributed content adaptation process. This new approach will enhance performance and increase flexibility of content delivery systems; however it requires an open and flexible infrastructure. In this paper we present content delivery infrastructure enabling the use of third-party adaptation services. This infrastructure provides mechanisms for context provision, adaptation service registry and discovery, and optimal realization of adaptation services.

Keywords: content adaptation, pervasive systems, adaptation services, adaptation graph

1. Introduction

Pervasive computing is the ability to extend applications and services normally conducted on office computers to handheld and wireless devices enabling anywhere, any time, with any device access to different information systems. Pervasive computing environment involves a variety of devices with different capabilities in terms of processing power, screen display, input facilities and network connectivity which can be wired or wireless. For pervasive computing to succeed, fundamental issues in different areas such as network, data management and delivery and security among others have to be well addressed [1].

Heterogeneity and mobility of terminals in pervasive environment is fostering the need to support adaptations of content and services in order to overcome terminal and network limitations. Content adaptation includes format transcoding (e.g. XML to WML, JPEG to WBMP), scaling of images as well as video and audio streams, media conversion (e.g. text-to-speech), omission or substitution of document parts (e.g. substitution of an image by a textual representation), document fragmentation, language translation, etc [7].

A number of approaches are being proposed in the literature to deal with content adaptation (see section 2). Some of the works propose building contents in formats specific for each terminal by the content provider; this is static content adaptation approach. While this approach has several advantages in terms of delivery performance, it lacks consideration of dynamic variations of terminal's hardware or software components. Moreover, authoring and updating of the content is very costly and impractical. To address such issues dynamic content adaptation is proposed. In dynamic content adaptation, the system gathers characteristics of terminals, users and networks, call delivery context or context profile, and adapt the content accordingly.

Based on location of the adaptation process, adaptation approaches are classified into three; server-side, client-side and proxy-based [9]. Recent research works [3, 13] propose service-based solutions to dynamically adapt content. In these works, content adaptations are performed by software elements call adaptation process (in our term adaptation operators or transformation) hosted by service providers and considered as value added services. This new approach will enhance the performance of the adaptation process as well as accommodates a number of adaptation requirements which is not the case in existing approaches; however it demands an open, flexible, and interoperable infrastructure. This infrastructure should provide mechanisms for discovering, selecting and composing adaptation processes required to meet the context constraint.

In this paper, we present service-based content adaptation infrastructure and its current implementation. The remainder of the paper is organized as follows. In section 2, we present briefly some of the related works followed by some usage scenario (section 3). Section 4 describes content adaptation infrastructure. In section 5, we present context and media object profiling. In section 6 we present content negotiation and adaptation module (CNAM). Prototype implementation and experimentation of the infrastructure is presented in section 7. In section 8 we conclude the paper and list future works.

2. Related works

One of the important issues with content adaptation is the location where the adaptation process should be done. With respect to location of the adaptation process, adaptation frameworks can be categorized into three groups [9]: server-side approach, proxy-based approach and client-side approach.

In the case of server-side approach (e.g [11], [12]), the functionality of the traditional server is extended by adding content adaptation process. In this approach both static (off-line) and dynamic (on-the-fly) content adaptations can be applied and better adaptation results could be achieved as it is close to the content; however clients experience performance degradation due to additional computational load and resource consumption on the server [6].

In proxy-based approach (e.g. [6], [10]), a proxy that is between the client and the server, acts as a transcoder for clients with similar network or device constraints. The proxy makes request to the server on behalf of the client, intercepts the reply from the server, decides on and performs the adaptation, and then sends the transformed content back to the client. In this approach there is no need of changing the existing clients and servers. The problem of proxy-based adaptation approaches is most of them are application specific and focus on particular type of adaptation such as image transcoding, HTML to WML conversion, etc. In addition, if all adaptations are done at the proxy it results in computational overload as some adaptations are computational intensive and this degrades the performance of information delivery like the case of the server-side approach.

Client-side approach (e.g [9], [15]) can be done in two ways; performing transformation by the client device or selection of the best representation after receiving the response from the origin server. This approach provides a distributed solution for managing heterogeneity since all clients can locally decide and employ adaptations most appropriate to them. However, adaptations that can benefit a group of clients with similar request can be more efficiently implemented with server-side or proxy-based approaches. Furthermore, all of the clients may not be able to implement content adaptation techniques due to processor, memory and network bandwidth limitation [4].

The above three approaches do not deal the problem of content adaptation from service perspective that can be commercialized and utilized by user, content provider or other service providers (like Internet Service Providers). Introducing content adaptation as a service (service-based adaptation approach) distributes the adaptation processes and results in performance enhancement especially for computational intensive applications such as key-frame extraction as it is done by specialized adaptation services. For example, a server that handles only language translation is inherently more efficient than any standard web server performing many additional tasks [5]. It also opens new opportunities to service providers as additional revenue. However, it is very important to have an infrastructure enabling content delivery system to incorporate such functionalities and mechanisms of discovering and configuring various adaptation services required to support the context constraints.

Works in progress such as the Open Pluggable Edge Services architecture (OPES) [12] and the Internet Content Adaptation Protocol (iCAP) [5] are closely related to our proposed approach. The OPES work deals with standardizing mechanisms to extend intermediaries with application-specific value added services. This work focuses at general level and does not address in depth the issues of content adaptation. The iCAP defines a method for forwarding HTTP messages, that is, it has no support for other protocols and for streaming media (e.g. audio/video) and only covers the transaction semantics (how do I ask for adaptation?) and not the control policy (when am I supposed to ask for which adaptation from where?).

In this paper we introduce a content delivery application infrastructure capable of supporting the utilization of external content adaptation services. This platform is open, flexible and interoperable. Three case scenarios are presented to show the need of such infrastructure and a prototype of one case scenario is developed to realize the conceptual model of the infrastructure.

3. Usage scenario

Scenario 1 Cooking guideline

One of Mrs. Catherine hobbies is cooking. She watches her favorite cooking program on TV. She gets interested in the recipe being explained so she decides to prepare a meal of the recipe. The program uses a multimedia database system to store video clips on cooking procedures, image and textual description of recipes and ingredients list. She logs to the programs web site with her PC to see details of the recipe. She goes to supermarket to buy the ingredients. She connects to the system with her cell-phone and get list of the ingredients. She comes back home and connects to the system with her digital radio and prepares the meal by listening to the radio transmitting the recipe description.

Scenario 2 Medical prescription

Let us consider an orthopedic rehabilitation center which serves European communities. This center is built with the state-of-art of modern health care system. This system is networked with similar centers throughout Europe and allows doctors, nurses, caretakers and other hospital staffs to access patient's medical records from any location, at any time and with any device. For example, a busy physician is making rounds at the center. He is visiting a patient from one of the European states. The patient shows signs of a bronchial infection. Before prescribing any new medication, the doctor wants to checks the patient's medical records on his wireless handheld device. He connects to the system and gets her historical data from previous centers she has visited and notes that she is already taking medications for high blood pressure and heart problems and is also allergic to penicillin. He then uses the handheld device to check an online drug database for the most effective treatment compatible with the patient's other medications. The doctor fills out a short form on the handheld device screen to authorize and send the prescription via a wireless network to a pharmacy.

Scenario 3 Tourism

Mr. Bob is a tourist who likes visiting museums. He is in Paris for a vacation and wants to visit le louver. Before he goes to the museum he wants to know its direction and location. He connects to location map and routing service system with his PDA. He discovers that the system provides route guiding service. He activates the route guiding service with his Smart phone and starts driving to the museum. After the visit he likes to walk around the district. He connects again to location mapping and routing service with his PDA and downloads the districts map. He visits the district appreciating architecture of the ancient building. He does some souvenir shopping and finishes his journey.

The scenarios presented above are from different application areas however, they highlighted the need of content delivery infrastructure that can provide multi-modal access of information such as video, image, sound and text, and supports heterogeneous terminals: PC, PDA, cell-phone, digital radio etc. The infrastructure should also be able to recognize terminal capabilities and contextual situations in order to apply different content adaptation mechanisms such as image resizing (to display the map on PDA), text-to-speech translation (to listen the recipe on the radio assuming the recipe is originally in text), language translation (to deliver to users with different language preference), summarization (patient record consists of consultation and examination data) and so on. The adaptation processes could be part of the application service or software elements provided by third party and accessible through a communication interface.

4. Content adaptation infrastructure

The content adaptation and delivery infrastructure proposed is described in Figure 1. Details of the components and their functions are discussed in [2]. Local proxies are access points to information delivery systems (content provider). They are responsible for retrieving and processing context profile (user, terminal, network), decide the type and number of adaptation operators, discover adaptation services and plan execution of the services.

Client profile is gathered and stored in the client profile repository during user subscription to an information service. Users can update and modify their profiles at any time. Dynamic profiles such as user location and networking condition are determined during user's request execution.

The infrastructure also includes adaptation service registry (ASR) and adaptation service proxies (ASPs). The ASR is like UDDI registry; it stores adaptation service profile (functional and non-functional) and allows for service look up. The UDDI uses a Web Service Description Language (WSDL) [14]. A WSDL file is an interface to use a web service. However, a WSDL file provides no information as to what the services is used for, what its input stands for, what the effect/consequence it will bring about, how much it costs, or its execution time. This is exactly the information required for automated web-service discovery and composition of web-processes (or multiple web-services) which is also the case of adaptation services in our system. In order to add some semantic information and facilitate service look up we have used a very simple and structured services description.

The ASPs host one or more adaptation operators with well define interface to be invoked by external parties. The local proxy plans optimal execution of adaptation process and invokes the services accordingly. In addition to functional properties, quality criteria such as cost, time, etc. are used during service selection.

The content proxies provide access to content servers, formulate user request to source format, manage and provide content profile (meta-data). The infrastructure proposed is flexible and open as new services can be added by just subscribing on the adaptation service registry. It is also interoperable as services can be hosted by different platforms.

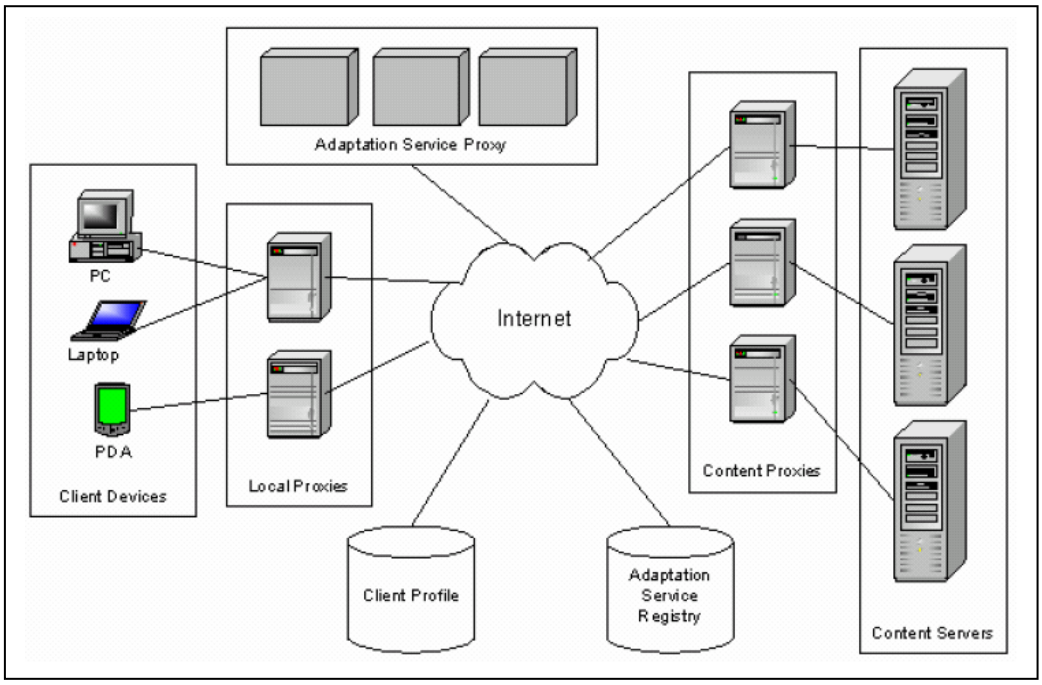

Figure 1: Service-based distributed content adaptation infrastructure

5. Context and media object profiling

Decision of the adaptation process depends on the extent and quality of profile information gathered; hence it is important to have a good mechanism of profile representation and processing. The two important profile informatioin are context and media object profile. While context profile is a collection of context information that is gathered from profiles such as user, device, network, location etc., media object profile is the meta-data of the content (a response to user's request).

Context profile

The context profile must give a complete description of the environment and provides easy way of processing. The context profile data structure (see Figure 2) describes a hierarchical profile representation. This structure is similar to the one developed by Held et al, [7] .To facilitate parsing and evaluation of the profile attributes, resource description framework RDF [8] based profiling called CSCP (Comprehensive Structured Context Profiles) [7] is used. The CSCP is an RDF based meta language, hence it inherits the interchangeability, decomposability and extensibility of RDF. The CSCP permits merging profile fragments that are dynamically retrieved even from different Web sites- for example, from profile repositories of device vendors. The XML representation of the profile is attached as *appendix A*. For instance, from fig 2, the audio capability of one device is described as "ContextProfile.DeviceProfile.Hardware.AudioCapable", that can be either true or false.

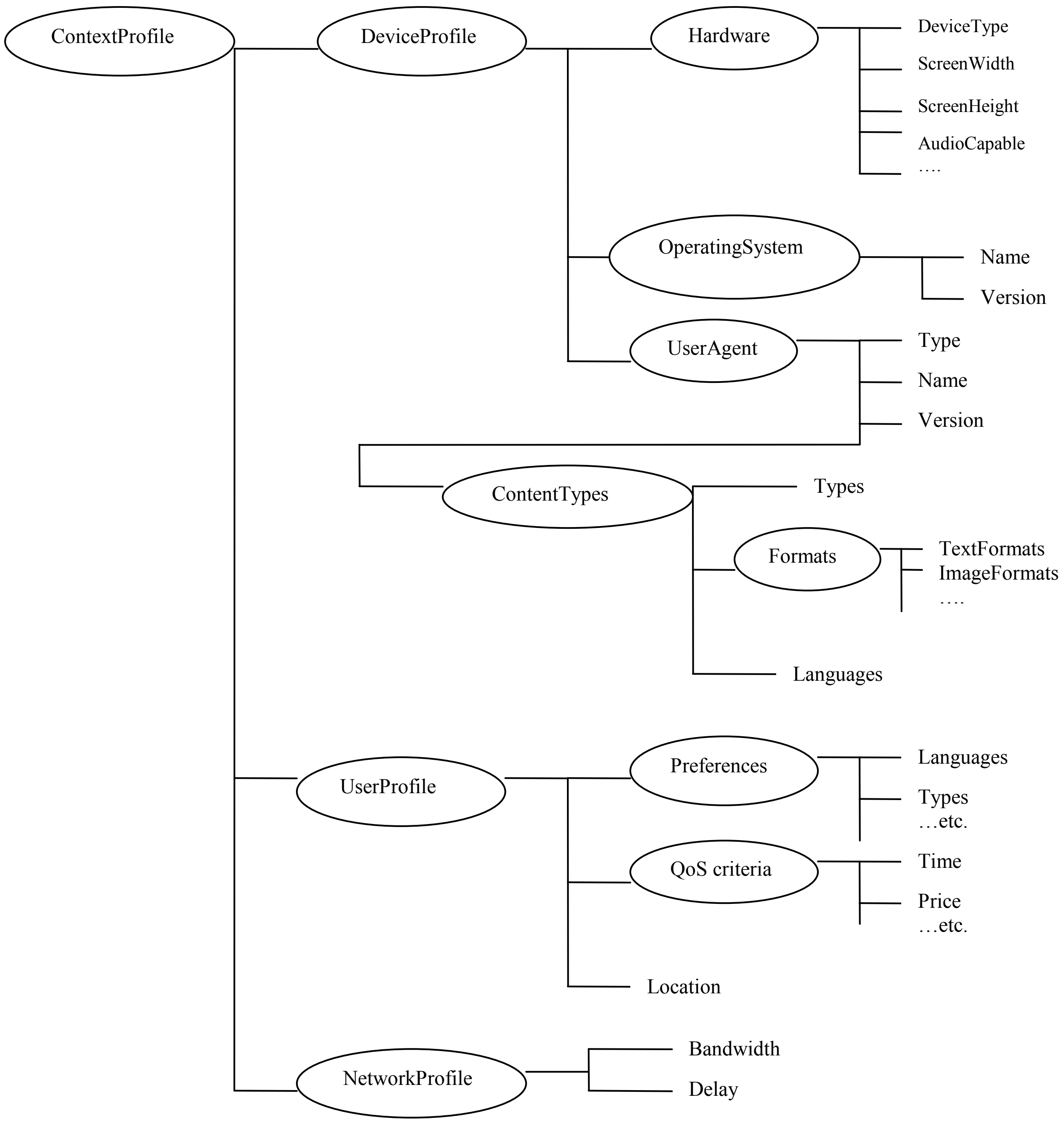

Figure 2: Context profile structure

Media object profile

A media object is a multimedia data stored in the content servers which can be text, image, audio or video accompanied by media object profile/metadata. The metadata consist of content and feature metadata. Content metadata is information about the object(s) captured in the media - the object's title, description, language, etc. and feature metadata is data about the media itself (not about the object(s) in the image). It includes information such as media type, file format, file size, media dimensions/spatial resolution (width and height), and colour information, identifier (location) etc. In addition to facilitating content search, the meta-data allows the adaptation process to preserve semantics of the content during adaptation.

The infrastructure proposed can be applied for the scenarios discussed above or other similar scenarios, however we will focus in medical application scenarios (e.g. scenario 2) to illustrate our purpose. In use case *scenario 2* we consider a physician with handheld device (e.g. BlackBerry 7750) accessing a patient's medical record while making rounds in order to authorize a prescription. In this example scenario, the physician wants to view the patient's medical history to know her/his past medical treatment in order to prescribe her/him right medicine. The medical record consists of text document showing medical treatment and some medical images (image object) of x-rays taken for the disease in question. In order to display the data on the handheld device, the system retrieves context profile and media object profile then it applies profile matching to determine adaptation requirements. Example of context and media object profile is given below:

Device profile

Harware	**Operating System**	**UserAgent**
DeviceType= BlackBerry 7750	Name=RIM OS	Type=Browser
ScreenWidth=240 pixels	Version=1.2	Name=RIM Browser
ScreenHeight=240 pixels		Version=3.2.1
ScreenSizeChar=26x16		…
ColorCapable=Yes		
Memory=14 MB		
Processor=ARM		
TextCapable=Yes		
ImageCapable=Yes		
AudioCapable=Yes		
…..		

User profile **Network profile**

Preferences
Language = {English, French, etc.}
Types= {audio, text, image}
…
	Bandwidth =1000 kbs
	Delay =10 seconds
	…..

Media object profile (e.g. image object)

Title: Examen de la gorge Description: Patient atteint d'un cancer de la gorge. A ce que nous avons pu constater, le patient ne peut s'empêcher de fumer un paquet de cigarette par jour. Language: French Type: image Format: JPEG FileSize: 61.1 kb Width: 660 pixels Height: 445 pixels ColorDepth: 24 bits Identifier: http:/localhost/imagefiles/gorge.jpeg

6. Content negotiation and adaptation module (CNAM)

The local proxy consists of content negotiation and adaptation module (CNAM) and adaptation services agent. The CNAM is responsible for creating an optimal adaptation plan according to the delivery context (client profile and other information). In order to decide appropriate adaptation processes/operators and their composition, decision must be made by considering all the factors. The module consists of profile manager and adaptation operator graph selector. The profile manager analyzes client, network and content profile and generates context profile (*see Appendix A for context profile*). Using the context profile and operator mapping description file (see Figure 3), the context profile processor generates a set of transformation processes/operators required to meet the constraints. Operator mapping description file is implemented as an XML file. The file specifies each transformation operator along with a set of constraints for that operator. The file contains Boolean expressions for aggregating constraints: and, or, and not. It has also a number of conditionals: *lessthan, lessthanequal, greaterthan, greaterthaneqaul, equal*, and *contains*.

```
<?xml version= "1.0" encoding= "UTF-8"?>
<operators>
<operator name= "text-to-audio">
<and>
```

```
 <equal value= “text”> MediaObjectProfile.Type</equal>
 <equal value= “Yes”> ContextProfile.DeviceProfile.Hardware.AudioCapable</equal>
 <contains value= “audio”> ContextProfile.UserProfile.Preferences.Types</contains>
 <contains value= “audio”> ContextProfile.DeviceProfile.UserAgent.ContentTypes.Types</contains>
</and>
</operator>
<operator name= “text-language-translation”>
<and>
 <equal value= “text”> MediaObjectProfile.Type</equal>
<or>
<and>
<equal value= “Yes”>ContextProfile.DeviceProfile.Hardware.AudioCapable</equal>
<contains value= “audio”>ContextProfile.UserProfile.Preferences.Types</contains>
<contains value= “audio”>ContextProfile.DeviceProfile.UserAgent.ContentTypes.Types</contains>
</and>
<and>
<equal value= “Yes”>ContextProfile.DeviceProfile.Hardware.TextCapable</equal>
<contains value= “text”>ContexProfile.UserProfile.Preferences.Types</contains>
<contains value= “text”>ContextProfile.DeviceProfile.UserAgent.ContentTypes.Types</contains>
</and>
</or>
<not>
 <contains reference= “MediaObjectProfile.Language”>ContextProfile.UserProfile.Preferences.Languages
</contains>
</not>
</and>
</operator>
<operator name= “text-summary”>
<and>
 <equal value= “text”> MediaObjectProfile.Type</equal>
<or>
<and>
<equal value= “Yes”>ContextProfile.DeviceProfile.Hardware.AudioCapable</equal>
<contains value= “audio”>ContextProfile.UserProfile.Preferences.Types</contains>
<contains value= “audio”>ContextProfile.DeviceProfile.UserAgent.ContentTypes.Types</contains>
</and>
<and>
<equal value= “Yes”>ContextProfile.DeviceProfile.Hardware.TextCapable</equal>
<contains value= “text”>ContextProfile.UserProfile.Preferences.Types</contains>
<contains value= “text”>ContextProfile.DeviceProfile.UserAgent.ContentTypes.Types</contains>
```

```
</and>
</or>
<equal value= “cell-phone”>ContextProfile.DeviceProfile.Hardware.DeviceType</equal>
</and>
</operator>
</operators>
```

Figure 3: XML representation of operator mapping rules

The **Context profile processor** uses the operator mapping description file as follows: it parses the file and constructs a postfix description of each set of constraints (see Figure 4). It stores this postfix description in a vector for evaluation later. For example the text-to-audio transformation is represented as:

expression	Profile variable	value	operand
equal	MediaObjectProfile.Type	“text”	
equal	ContextProfile.DeviceProfile.Hardware.AudioCapable	“Yes”	
contains	ContextProfile.UserProfile.Preferences.Types	“audio”	
contains	ContextProfile.DeviceProfile.UserAgent.ContentTypes.Types	“audio”	
and			4

Figure 4: profile processing

The processor evaluates the postfix description of a set of constraints by retrieving each expression, evaluating it and then writing the result back to a result stack. In the case of text-to-audio transformation it examines audio capability of the device, user’s content type preference, accepted content types and writes true or false accordingly. Then it applies the ‘and’ operator on the results to determines if text-to-audio transformation is required. The processor repeats this process for each transformation operators and returns a vector containing the names of transformation operators necessary to meet the context constraint. Result of the processor is a file consists of transformation operators. We call this file *transformation prescript* which is stored in XML file. Considering *scenario 2* and a text object response containing the patient’s medication history, we will get a transformation prescript represented in XML as follows:

```
<?xml version= "1.0" encoding= "UTF-8"?>
<transformationPrescription>
<source>
<MediaProfile type="text"
identifier="http://lolalilou.webzzanine.net/text-test.txt" format="text/plain"
language="EN"/>
</source>
<transformation name="text-to-audio">
<input type="text"/>
```

```
<output type="audio"/>
</transformation>
<transformation name="text-language-translation">
<input type="text"/>
<output type="text"/>
</transformation>
<transformation name="text-summary">
<input type="text"/>
<output type="text"/>
</transformation>
</transformationPrescription>
```

Figure 5: a transformation prescript

6.1 Adaptation operator graph

The transformation prescript represented in Figure 5 is a graph of operators. This graph is called logical adaptation operators graph (LOG) or transformation prescript graph (see Figure 6).

A logical adaptation operators graph (LOG) is a DAG where

- The start nodes of the graph are media sources, the intermediate nodes are operators and the end nodes are media sink or adapted media.
- The edges are media/data flow from one operator (or media source) to another operator (or media sink).

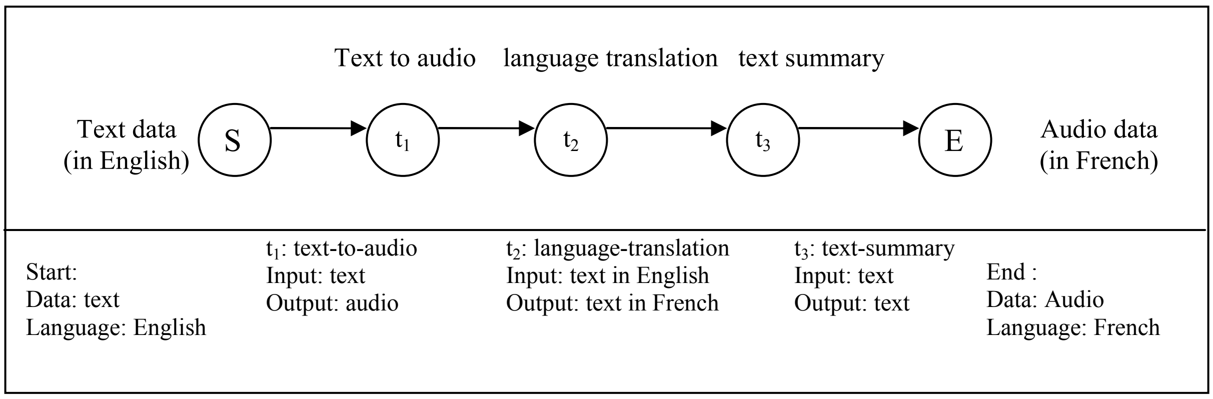

Figure 6: transformation prescript graph

The transformation prescript is just a mapping of the constraints. It does not include operator implementation details such as operator signature (specific media format, quality values etc.) and execution dependency. Compatibility between operators is at media type level (e.g mediaTranslation (text, audio), languageTranslation (text, text, English, French)). Hence, we introduced two operator graph conversion rules a) substitution of an operator by implementable operator (the operator with its signatures) b) tuning or reordering. These rules enable to produce a complete graph. Before we see how the conversion rules are applied, let us see two basic concepts of operators: equivalence relationship and execution dependency.

Equivalence relationship

The graph conversion rules are derived from a number of equivalence relations. We have defined the following three important equivalence relationships:

Neutrality: operators that can be inserted anywhere in an operator graph or removed without changing the semantics of the graph e.g. format conversion operators.

Reversibility: an operator is reversible if it has an inverse operator such that their pairing produced semantic neutrality e.g. compression and decompression.

Commutative: two operators are commutative if their order of application does not affect the semantics of the graph.

The substitution rule replaces an operator by its corresponding implementable operator and there may be one or more additional operators inserted before and after the implementation operator for the purpose of type compatibility or else. These additional operators are semantically neutral or pair of reversible operators.

The basic equivalence relationships defined above lead to different tuning rules. By applying tuning rules (adding or removing one or more operators), we produce semantically equivalent operator graphs. We say an operator implementation graph is complete when (1) all the operators of the graph are instantiatable and (2) no signature mismatch between operators.

Execution dependency

Sequential dependency: with this type of execution dependency, the operators are invoked successively one after the other. The execution of the next operator depends on its preceding operator, i.e. the next operator can not begin unless its preceding operator has terminated. For example color depth reduction operator and image compression operator. The execution order could be color depth reduction and compression. The compression process will be activated once the color depth reduction is done.

Pipeline dependency: in a pipelined dependency, the operators are invoked successively but without waiting for termination of the first operator. The next operator can start with partial outputs of the preceding operator. Pipeline execution produces operator parallelism. For example, the *summarizing* operator and *language translating* operator can be executed in pipeline since the *language translator* can start executing without waiting for termination of the *text summarizer*.

Parallel dependency with synchronization (intra-operator): in this type of dependency the operators are invoked concurrently. However, the results of their execution need to be combined. For example, resolution reduction and language translation of a video can be executed in parallel by separating the video frames and the audio part: however, they need to execute to completion and their results need to be combined in order to obtain the video with reduce resolution and translated language.

Parallel dependency without synchronization (inter-operator): in this type of dependency the operators are invoked concurrently and there is no need of synchronization. For example, an image object has description metadata. It may be necessary to apply operators on the description at the same time on the image, e.g. language translator on the description and format converter on the image. In this case the two operators can be executed in parallel and no need of synchronization; of course the result will be merged to produce the final output.

6.2 Operator implementation graph

An operator implementation graph (OIG) or transformation implementation graph is a labeled DAG, G= (V, E, L) where

- V is the set of vertices representing implementable operators
- E is the set of edges over V represents data/message flow between operators
- L is a label representing execution dependency between operators.

Taking the transformation prescript of Figure 6 and applying the conversion rules, we will have a transformation implementation graph (Figure 7) and its XML representation (Figure 8).

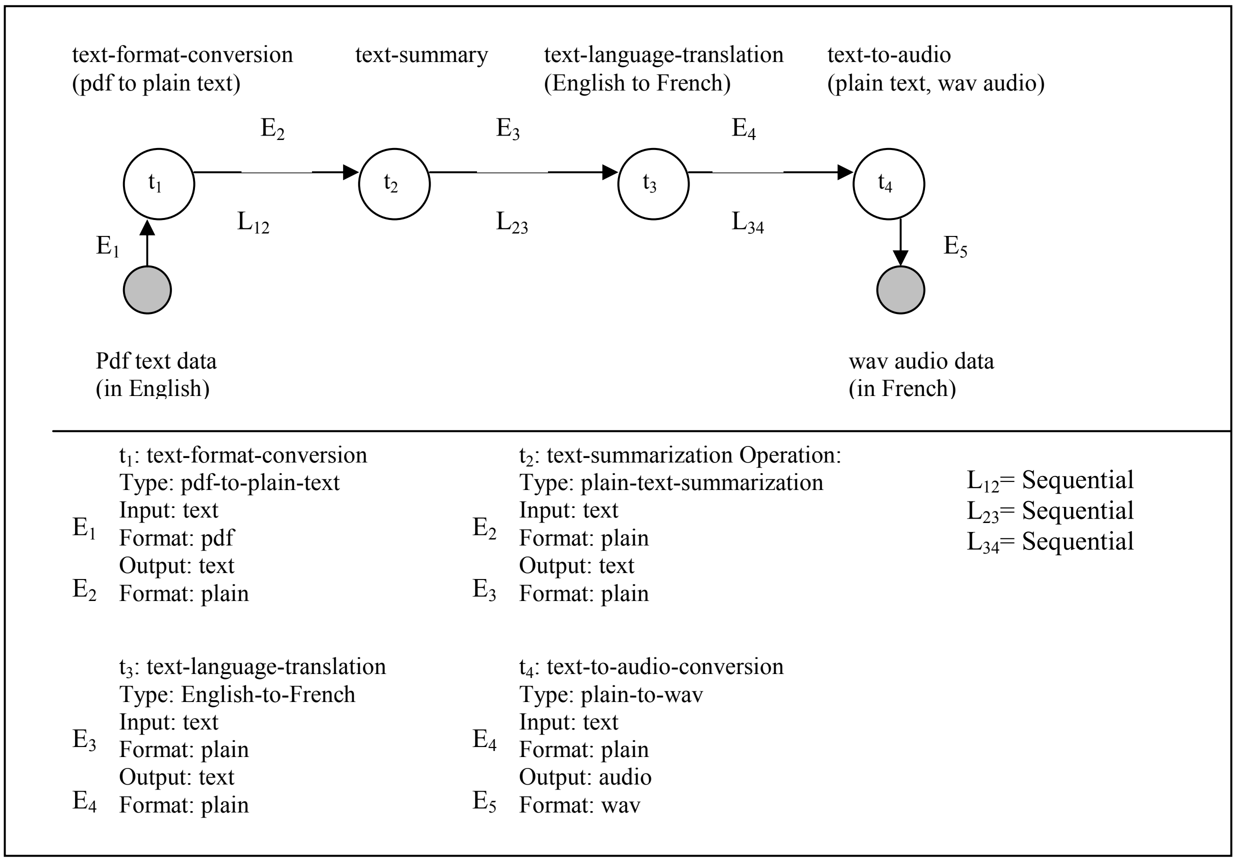

Figure 7: transformation implementation graph

```
<?xml version= “1.0” encoding= “UTF-8”?>
<transformationPrescription>
    <source>
        <MediaObject ALIAS= “object_text” REF=identifier/>
    </source>
    <transformation name= “text-format-conversion” >
    <input ALIAS= “text_in” REF= “object_text”/>
    <parameter name= “Language” Value= “EN”/>
    <signature>
        <property Name= “Type” >TEXT</property>
        <property Name= “Format”>pdf</property>
    </signature>
    <output ALIAS= “text_out” REF= “object_text”/>
    <parameter name= “Language” Value= “EN”/>
    <signature>
        <property Name= “Type” >TEXT</property>
        <property Name= “Format” >plain</property>
```

```
</signature>
</transformation>
<transformation name= “text-summary”>
<input ALIAS= “text_in” REF= “object_text”/>
<parameter name= “Language” Value= “EN”/>
<signature>
      <property Name= “Type” >TEXT</property>
      <property Name= “Format” >Plain</property>
</signature>
<output ALIAS= “text_out” REF= “object_text”/>
<parameter name= “Language” Value= “EN”/>
<signature>
      <property Name= “Type” >TEXT</property>
      <property Name= “Format” >plain</property>
</signature>
</transformation>
<transformation name= “text-language-translation”>
<input ALIAS= “text_in” REF= “object_text”/>
<parameter name= “Language” Value= “EN”/>
<signature>
      <property Name= “Type” >TEXT</property>
      <property Name= “Format” >Plain</property>
</signature>
<output ALIAS= “text_out” REF= “object_text”/>
<parameter name= “Language” Value= “FR”/>
<signature>
      <property Name= “Type” >TEXT</property>
      <property Name= “Format” >plain</property>
</signature>
</transformation>
<transformation name= “text-to-audio” >
<input ALIAS= “text_in” REF= “object_text”/>
<parameter name= “Language” Value= “EN”/>
<signature>
      <property Name= “Type” >TEXT</property>
      <property Name= “Format”>Plain</property>
</signature>
<output ALIAS= “audio_out” REF= “object_text”/>
<parameter name= “Language” Value= “EN”/>
<signature>
```

```
        <property Name= “Type” >AUDIO</property>
        <property Name= “Format” >WAV</property>
    </signature>
    </transformation>
</transformationPrescription>
```

Figure 8: XML representation of transformation implementation graph

6.3 Executing a transformation graph

Once the operator implementation graph is generated, an adaptation service agent module sends a request to ASR for list of available services for each operator and returns the result to the CNAM. The CNAM uses a selection strategy model developed in [2] to determine which services to use. This model considers quality criteria such as cost, time etc. to determine an optimal service selection and execution. The XML description of operator graph together with the services selected to each operator is passed to the adaptation service agent then the services are invoked.

7. Prototype Implementation and experimentation

We have developed a prototype implementation of the content adaptation infrastructure in the context of medical application with some sets of adaptation services that allow us to test the viability of the ideas presented such as optimal execution of operator graph and service composition. The prototype has been developed in Java using MySQL.

Usage example of the prototype

Consider scenario 2, a physician connects to the system through the local proxy and request for patient’s medical history with his handheld device. The local proxy gathers client profile (user and terminal) from the profile repository and forwards user request to content proxy. The content proxy sends meta-data response to the request. This meta-data is used for filtering the response by comparing it with the client profile. The user’s profile shows that the user prefers in French and audio format. On the other hand the meta-data indicates that the text is very long and written in English.

The above simple example shows the need of applying a series of adaptation process on the content to deliver at least in the current context. In this example we need a minimum of three processes that are language translation, text summarization and text to audio translation. These adaptation processes can be provided by local or remote services. In our prototype we have used remote adaptation services developed with Web Service technologies. We have not yet made the prototype accessible from external; however, we have put all the source codes and executable files with full documentation manual on the URL (http://jmp.dyndns.org/AdaptationProject). The documentation manual is well prepared and simplifies the installation process so any one can install and test the prototype.

Preliminary experiment

Detailed experimentation has not been done to measure performance of the CNAM, this is the next work of the research, however, our early experience with the service-based infrastructure and its prototype implementation gives us encouraging results in terms of openness, flexibility, and interoperability. The infrastructure is able to use external services hosted in different platform and new services can be incorporated easily through a mechanism of publishing on the service registry.

As a preliminary experiment, we have measure execution time of different adaptation services for different image sizes. As appears in Figure 9, the execution time is not always the same for example in the case of image resizing it varies from 100 ms (minimum) to 400 ms (maximum). The variation is expected since the adaptation services are at distance and the load of the network varies with time. From this result we can observe that network connection of a service is an important parameter that must be taken into account in the process of services selection.

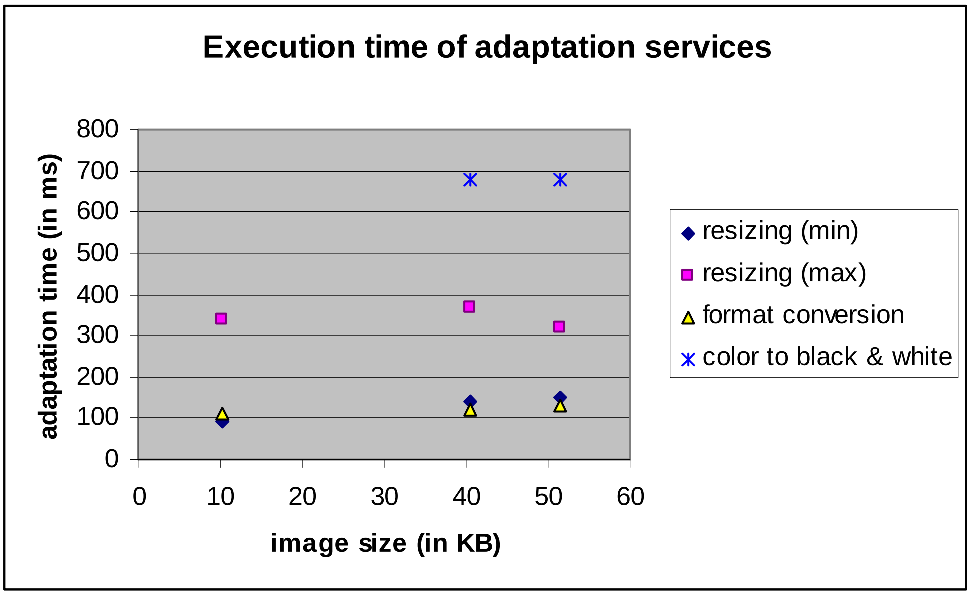

Figure 9 Execution time of image adaptation services.

8. Conclusion and future work

In this paper, we have presented a service-based infrastructure for distributed content adaptation approach. One of the key features of the infrastructure is the use of external or remote adaptation services developed in Web Service technology. This allows development of new adaptation techniques and incorporated easily in the system. The service-approach results in better adaptation process at the same time it is open, flexible and interoperable. The prototype we have developed justifies these properties. Moreover, performance measurement of different adaptation services done on the prototype indicates feasibility of utilizing remote services. Future work focuses on full implementation of the optimal adaptation strategy and experimentation of its performance with respect to operator and service complexity.

Appendix A: Context profile in XML representation

```
<? xml version=”1.0” encoding “UTF-8”?>
<rdf:rdf
  xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
  xmlns:cp ="http://example.org/CPSyntax#">
  xmlns ="http://example.org/ContextProfileSyntax#">
  xmlns:dev="http://example.org/DeviceProfileSyntax#">
  xmlns:use="http://example.org/UserProfileSyntax#">
  xmlns:net="http://example.org/NetworkProfileSyntax#">
<ContextProfile rdf:ID= ‘Context’>
      <device>
            <dev:DeviceProfile>
                  <dev:Hardware>
                        <dev:deviceType> </dev:deviceType>
                        <dev:displayHight> </dev:displayHeight>
                        <dev:displayWidth </dev:displayWidth>
                        <dev:ImageCapable> </dev:ImageCapable>
                        ……
                  </dev:Hardware>
                  <dev:Software>
                        <dev:Name></dev:Name>
                        <dev:Version></dev:Version>
                  </dev:Software>
                  <dev:UserAgent>
                        <dev:Type></dev:Type>
                        <dev:Name></dev:Name>
                        <dev:Version></dev:Version>
                        <dev:contentTypes>
                        <rdf:Bag>
                        <rdf:li>
                        <dev:contentType> </dev:contentType>
                        </rdf:li>
                        <rdf:li>
                        <dev:contentType> </dev:contentType>
                        </rdf:li>
                        </rdf:Bag>
                        </dev:contentTypes>
                  </dev:UserAgent>
            </dev:DeviceProfile>
```

```
    </device>
    <user>
        <use:UserProfile>
            <use:Preferences >
                <use:Languages>
                <rdf:Bag>
                <rdf:li> </rdf:li>
                </rdf:Bag>
                </use:Languages>
                ……
    <network>
        <net:NetworkProfile>
            <net:Bandwidth > </net:Bandwidth>
            <net:Delay > </net:Delay>
        </net:NetworkProfile>
    </network>
</ContextProfile>
</rdf:RDF>
```

References

[1] Acharya, S., “Application and Infrastructure Challenges in Pervasive Computing”, NSF Workshop on Context-Aware Mobile Database Management (CAMM), Providence, Rhode Island, January 24-25, 2002.

[2] Berhe, G., Brunie, L., Pierson, J-M., “Modeling Service-Based Multimedia Content Adaptation in Pervasive Computing”, ACM Proceedings of the first conference on computing frontiers (CF’04). Ischia, Italy, April 14 - 16, 2004 pp. 60 – 69.

[3] Buchholz, S., Schill, A., “Web Caching in a Pervasive Computing World”, Proc. of the 6th World Multiconference on Systemics, Cybernetics and Informatics (SC12002), Orlando, FL, USA, Jul 14-18, 2002.

[4] Cardellini, V., Yu, P. S., Huang, Y.W., “Collaborative proxy system for distributed Web content transcoding" , Proc. of 9th Int'l ACM Conf. on Information and Knowledge Management, Washington, DC, pp. 520-527, Nov. 2000.

[5] Elson, J., Cerpa, A., “ICAP-the Internet Content Adaptation Protocol”, Internet Draft draft, the ICAP Protocol Group, June 2001. (URL: http://www.i-cap.org/spec/icap specification.txt).

[6] Han, R., Bhagwat, P., LaMaire, R., Mummert, T., Perret, V., and Rubas, J. “Dynamic Adaptation in an Image Transcoding Proxy for Mobile WWW Browsing”. IEEE Personal Communication, 5(6):8--17, 1998.

[7] Held, A., Buchholz, S., Schill, A., “Modeling of Context Information for Pervasive Computing Applications”, Proc. of the 6th World Multiconference on Systemics, Cybernetics and Informatics (SC12002), Orlando, FL, USA, Jul 14-18, 2002

[8] Lassila, O. and R. Swick (1999). Resource Description Framework (RDF): Model and Syntax Specification. W3C Recommendation, [available online from www.w3.org/TR/REC-rdf-syntax/,

[9] Lei, Z., Georganas, N.D., “Context-based media adaptation in pervasive computing”, Proceedings of IEEE Canadian Conference on Electrical and Computer (CCECE'01), Toronto, May 2001

[10] Lum, W.Y. and Lau, F.C.M., “A Context-Aware Decision Engine for Content Adaptation”, IEEE Pervasive Computing, Vol. 1, No. 3, July-September 2002, 41-49.

[11] Mohan, R., Smith, J., Li, C.S., “Adapting Multimedia Internet Content For Universal Access”, IEEE Transactions on Multimedia, March 1999, pp. 104-114.

[12] Noble, B.D., Price, M., and Satyanarayanan, M., “A Programming Interface for Application-aware Adaptation in Mobile Computing”, Proc. 2nd USENIX Symposium on Mobile and Location-Independent Computing, Ann Arbor, MI, USA, Apr. 1995.

[13] Rahman, R., Menon, R.R. and Rafalow, L., “Enabling OPES to Use Web Service for Callout Service”, draft-rrahman-web-service-00.txt (work in progress), May 2002 (http://standards.nortelnetworks.com/opes/non-wg-doc/ draft-rrahman-web-service-00.txt).

[14] Web Service Definition Language. http://www.w3.org/TR/wsdl.

[15] Yoshikawa, C., Chun, B., Eastham, P., Vahdat, A., Anderson, T., and Culler, D., “Using smart clients to build scalable services”, Proc. Winter 1997 USENIX Technical Conf., Jan. 1997.