Distributed Multimedia Content Adaptation for Pervasive Systems

PhD Thesis : Girma Berhe
Advisors : Jean-Marc Pierson, Lionel Brunie

Lab. LIRIS
INSA de Lyon

Overview

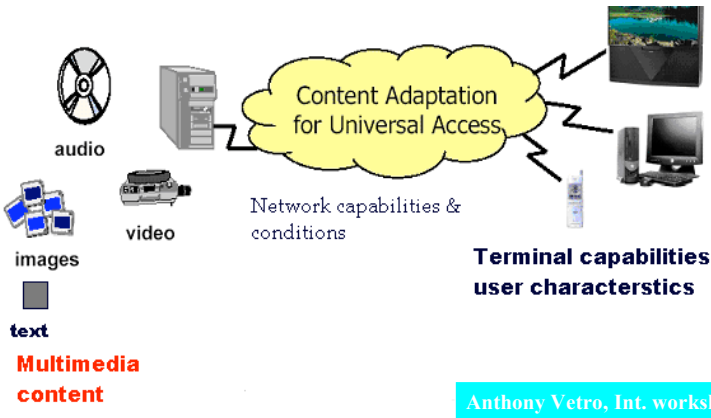
- Pervasive Computing
- Content Adaptation
- Distributed approach
- Negotiation and adaptation model
- Conclusion
- Future Work

What is Pervasive Computing?

Definition: extending applications and services to handheld and wireless devices enabling anywhere, any time, any device access to information systems.

- Pervasive computing devices (Beth Archibald Tang)
  - tiny even invisible devices
  - mobile or embedded in any type of object including cars, tools, appliances, clothing and various consumer goods
  - communicate through interconnected networks

Content Adaptation (Figure 1)

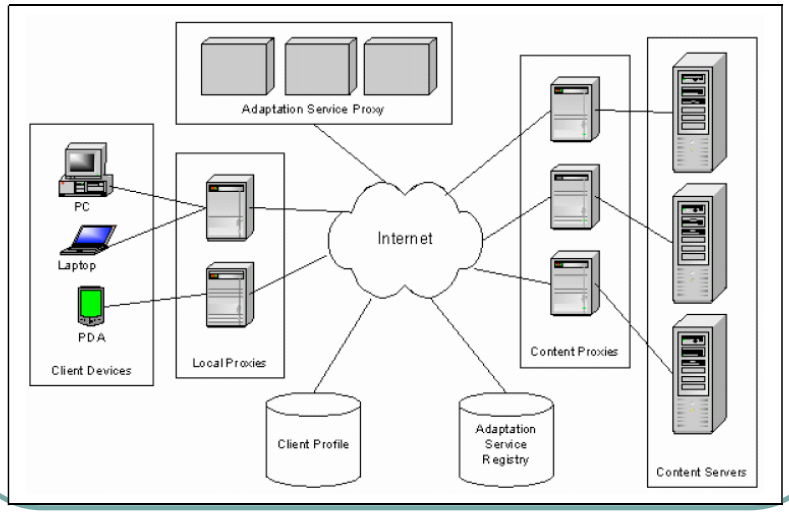


Anthony Vetro, Int. workshop VLB03 Panel on Content Adaptation September 18-19, 2003 Madrid, Spain.

Adaptation Approaches

- Server-side: content server does the adaptation (static and/or dynamic)
- Client-side: client device doing transformation or selection of the best representation
- Proxy-based: a proxy between the server and the client does the adaptation.
- Service-based (distributed) : third party provides the adaptation service

Distributed Content Adaptation Framework Architecture (Figure 2)



Components of the Architecture

- Local Proxies (LP)
- Content Proxies (CP)
- Adaptation Service Proxies (ASP)
- Adaptation Service Registry (ASR)
- Client profile repositories

Content Negotation and Adaptation Module (CNAM)

- Consists of profile manager and adaptation graph generator
- Profile manager analysis client, network and content profile and generates context profile
- Using context profile and operator mapping description file (both are in XML form), the Context profile processor generates a set of transformation processes called **transformation prescript** graph required to meet the constraints.

Transformation Graph

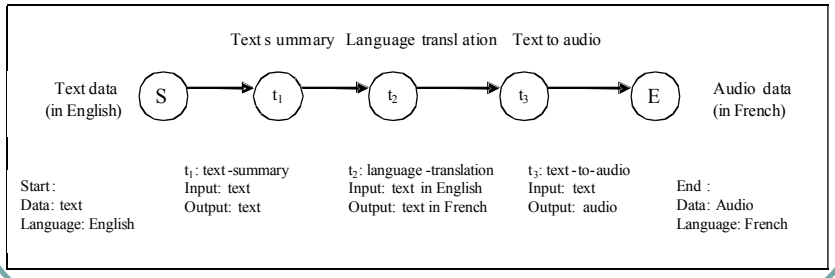
Consider a user request with mobile phone (and language preference French); let us assume we have one result of the request which is text (in pdf format and English language).

To match the context constraint, we need a transformation of text-to-audio which consists of {textSum, English2French, text2audio}.

Transformation Graph (Figure 3)

For the given example, several transformation graphs can be conceived that can accomplish the desired transformation.

One possible graph is:



Content Adaptation Graph

- Using a transformation prescript graph, T, we construct the adaptation graph as follows:
- Let T= $\{t_1, t_2, \ldots t_n\}$ where n is number of operators
- Step 1:
  - For each operator t_i, we find candidate adaptation services that can execute t_i: this will generate a set of adaptation services for each operator.

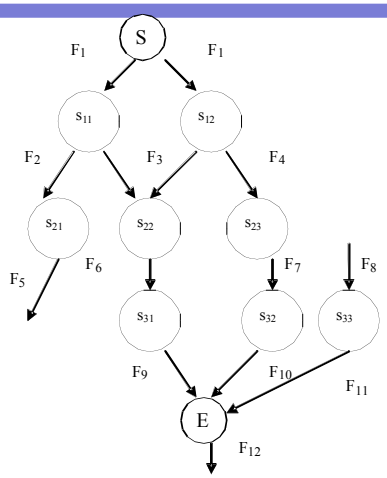
G= $\{<t_1, s_{11}, s_{12}, ..>, <t_2, s_{21}, s_{22}, \ldots>, \ldots, <t_n, s_{n1}, s_{n2}, \ldots>\}$

Content Adaptation Graph…

- Step 2:
  - To connect services, input-output compatibility of application-level QoS parameters (eg. Data format) are used.

-> The result is a directed graph (Figure 4).

Content Adaptation Graph…(Figure 4)

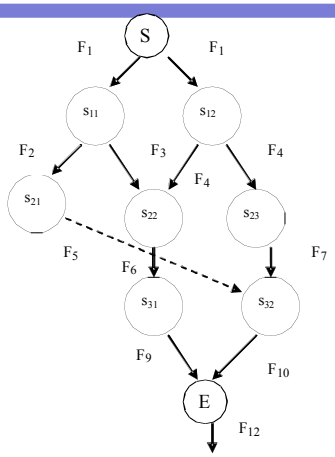


Content Adaptation Graph…

- Step 3:
  - There are unconnected edges like F_5 and F_8.
  - To connect these edges we look for one or more services if not found we remove them from the graph.

-> The result is a complete adaptation graph (Figure 5)

Content Adaptation Graph…(Figure 5)



Optimal path selection

- Step 4:
  From Figure 5 we can see that we have four possible adaptation paths.
- *The problem of path selection can be stated as follows:*
  - *given a set of transformation processes and their logical links and given a set of adaptation services; the best mapping of the processes, required to support the request's constraints, onto the adaptation services that supports them must be computed.*

Optimal path selection…

- *How?*
  - *Based on quality criteria e.g. cost and time*
  - Given time and cost as quality criteria, the quality vector if a service s for an operator t is defined as follows:
  - $Q(s)=(S_{cost}(s,t), S_{time}(s,t))$

Optimal path selection…

- The quality vector of a path is calculated as the sum of the quality vector of its component services and defined as follows;
- $Q(P) = (Q_{cost}(P), Q_{time}(P)$, where

$$Q_{cost}(p) = \sum_{i=1}^{n} S_{cost}(s_i, t_i)$$

$$Q_{time}(p) = \sum_{i=1}^{n} S_{time}(s_i, t_i)$$

Optimal path selection…

- In order to aggregate values of the different criteria we scaled them using the following equation:

$$Qs_{ij} = \begin{cases} \dfrac{Q_j^{max} - Q_{ij}}{Q_j^{max} - Q_j^{min}} \; if \; Q_j^{max} - Q_j^{min} \neq 0 \\ 1 \quad if \quad Q_j^{max} - Q_j^{min} = 0 \end{cases}$$

$$Qs_{ij} = \begin{cases} \dfrac{Q_{ij} - Q_j^{min}}{Q_j^{max} - Q_j^{min}} \; if \; Q_j^{max} - Q_j^{min} \neq 0 \\ 1 \quad if \quad Q_j^{max} - Q_j^{min} = 0 \end{cases}$$

Optimal path selection…

- User's preferences of quality criteria such as fastest or cheapest can be incorporated using weighting values in calculating overall quality score value of each path as follows:

$$Score(P_p) = \sum_{i=1}^{m} \sum_{j=1}^{n} Qs_{ij} * W_j$$

Conclusion

- We have developed an open, flexible and interoperable distributed content adaptation architectural framework
- The preliminary experiment (performance measurement) done on the prototype indicates feasibility of using the adaptation services.

Future Work

- Doing more experiment on the prototype to study the performance of graph construction and path selection algorithm.
- Enhancing the adaptation service description for example using OWL.