# Object Placement and Caching Strategies on AN.P2P*

Su Mu[1], Chi-Hung Chi[2], Lin Liu[2], and HongGuang Wang[2]

[1] School of Computing, National University of Singapore, Singapore
[2] School of Software, Tsinghua University, Beijing, China
chichihung@mail.tsinghua.edu.cn

**Abstract.** This paper discusses the object placement and caching strategies on the AN.P2P, which is an Application Networking infrastructure to implement pervasive content delivery on the peer-to-peer networks. In general, the AN.P2P allows the peer to send the content's original object with the associated content adaptation workflow to other peers. The recipient peer can reuse the original object to generate different content presentations. In order to achieve reasonable performance, this paper proposes several object placement schemes and caching strategies. Our simulation results show that these methods could effectively improve the system performance in terms of query hops, computation cost and retrieval latency.

## 1 Introduction

In recent years, the peer-to-peer (P2P) systems have witnessed more heterogeneous presentation requirements than before because of the emergence of diverse devices. However, legacy P2P file sharing facilities [5,6,11,16,17] cannot deal with these requirements effectively without necessary system support to content adaptation and service customization. For instance, a PC based peer shared a piece of high quality media content. However, a smart phone based peer cannot render this content unless it is adapted to a lower quality version. This necessitates the architectural change in tandem.

Our previous work [20] on Application Networking (App.Net) was to implement pervasive content delivery for heterogeneous requirements. Its key idea was rooted from the observation that despite the heterogeneous requirements for object presentations, the processes to generate the presentations are homogeneous. In general, the App.Net enables the delivery of content's original or intermediate object with an associated workflow. The recipient node can thus reuse these objects to generate different presentations by performing the tasks in the workflow. In particular, our work [21] on AN.P2P attempted to apply the Application Networking mechanism onto the P2P file sharing systems. We have shown that the AN.P2P could help to reduce the average search size because of the reuse of original object and the associated workflow.

However, integrating the Application Networking mechanism into a practical P2P system is not a trivial exercise. It involves multiple aspects as performance

* This research is supported by the funding 2004CB719400 of China.

J.X. Yu, M. Kitsuregawa, and H.V. Leong (Eds.): WAIM 2006, LNCS 4016, pp. 182 – 192, 2006.

optimization, service security, application reuse, and so on. The work of this paper focused on improving overall performance on AN.P2P through dedicated object placement methods and the revised caching strategies. In particular, we proposed two placement schemes to populate objects and their associated workflow, in order to improve the query efficiency and to reduce the mean retrieval delay. Our simulation showed that these methods could help to improve the overall system performance.

## 2 Summary of Previous Work

In general, the Application Networking mechanism [20] allows the content provider associate a piece of content adaptation logic to the published content. Instead of encapsulating the logic into a single application module, we specified the logic as a ***workflow*** composed by multiple tasks, each of which can be instantiated by an application module. We have defined a standard interface, called ANlet, for mobile applications. Given application modules that implement the ANlet interface, the Application Networking nodes can dynamically load them locally and remotely. In particular, we defined a callback function in ANlet, as given by equation (1).

*AppNetContent modResponse (queryMsg msg, AppNetContent content);* (1)

The function has two input parameters: the query message and available content object. The query message contains the presentation profile of the client and AppNetContent contains both the content object and the attributes of the object. Therefore, the ANlet can generate the appropriated output ANP2PContent according to the input parameters.

Hence, if we input the original object to a workflow, it can output the final presentation of the input object by performing the adaptation tasks. In particular, we believe the workflow structure can facilitate the progressive application deployment in that the fundamental applications are loaded first, while the remaining ones are loaded later on demand. Moreover, the workflow can get the intermediate objects of the original object. Our simulation results will show that caching these intermediate objects can help to reduce the average computation time by avoiding repeated content adaptation.

Upon receiving a request, the content host can send the original or intermediate object and the associated workflow to the requestor. The recipient can thus reuse the retrieved object to generate appropriate content presentations by performing the workflow associated with the response object. In addition, a workflow normally specifies a list of URLs, which can provide the relevant application modules. In case of no available local application module to instantiate a task, the node can download a mobile application from remote site according the URL specified by the workflow. Due to dynamic application loading, we can achieve reasonable system scalability in the open Internet environment.

In particular, our recent work on AN.P2P [21] implemented an Application Networking platform on P2P networks. The AN.P2P can reside on any P2P substrate and is not mandatory to install on each peer within the network. When an AN.P2P peer serves an ordinary peer, it will send a particular presentation of the content as if the AN.P2P is transparent to the ordinary peers. In contrast, when an AN.P2P peer serves another AN.P2P peer, it can send either the original object and the workflow or