멀티미디어 스트림을 위한 적응적 트랜스코딩과 캐슁 프록시 시스템 아키텍처

설성운° 서대화
경북대학교 전자공학과 컴퓨터시스템 연구실
{sulsa21°}@palgong.knu.ac.kr, {dwseo}@ee.knu.ac.kr

Adaptive Transcoding and Caching Proxy System Architecture for Multimedia Stream

Seong-Woon Seol°, Dae-Wha Seo
Dept. of Electronic Engineering, Kyungpook National University, Korea

요 약

이동 컴퓨팅 환경에서의 프록시는 트랜스코딩과 캐슁 등의 방법으로 미디어 스트림을 재가공하여 멀티미디어 스트리밍의 연속성을 보장해 준다. 프록시에 대한 기존의 접근들은 대체로 트랜스코딩과 캐슁을 별개의 주제로 다루었었다. 따라서 본 논문에서는 트랜스코딩과 캐슁을 통합하여 연동시킨 프록시 시스템 아키텍처를 구현하였다. 또한 서버와 프록시 사이의 전송 지연, 프록시에서의 미디어 스트림 재가공에 걸리는 지연, 프록시와 클라이언트 사이의 bandwidth 자원 등을 참조하여 adaptive한 트랜스코딩 및 캐슁을 할 수 있도록 해주는 transco-prefix caching에 대해 제안하였다. 그 결과 transco-prefix caching을 통하여 트랜스코딩과 캐슁의 장점들을 동시에 살릴 수 있었다.

1. 서 론

스트리밍 기술은 대용량의 멀티미디어 데이터를 즉시 재생해주는 기술이다. 여기서 전달되는 주체인 멀티미디어 데이터는 일반적으로 대용량이라는 속성과, 재생 시간 안에 반드시 재생되어야 한다는 실시간적인 속성을 갖는다. 하지만 인터넷은 최선의 서비스(Best Effort Services)를 토대로 한 네트워크이기 때문에 대용량의 데이터를 실시간으로 전송하기 힘든 구조이다. 따라서 스트리밍에 사용되는 기술들로는 대용량의 데이터의 부피를 줄이기 위한 멀티미디어 압축기술과 실시간 전송을 지원하는 네트워크 기술로 나눌 수 있다. 그 중에서 멀티미디어의 압축등을 포함한 미디어 재가공 기술들은 멀티미디어 스트리밍의 연속성을 보장해주어야 하는 이동 컴퓨팅 환경에서 절실히 요구되고 있다.

이동 컴퓨팅 환경에서 멀티미디어 스트리밍의 연속성을 방해하는 요인들로는 세 가지를 들 수 있다. 무선망의 빈곤한 bandwidth 자원, 모바일 호스트의 낮은 resource capacity, 서버와 모바일 호스트 사이의 전송 지연 등이다. 그 외 모바일 호스트의 handoff로 인한 미디어 스트림의 불연속성 문제도 있으나 이는 Mobile IP 관련 주제로 수렴시킬 수 있다.

보통 모바일 호스트가 접해 있는 무선망은 유선망보다 상대적으로 빈곤한 bandwidth 자원을 가지고 있다. 따라서 서버로부터의 높은 품질과 고용량의 멀티미디어 스트림이 전송된다 하더라도, 무선망의 빈곤한 bandwidth로 인해 모바일 호스트에서의 연속적인 미디어 스트림 수신이 불가능한 일이 발생한다. 그러므로 이를 해결하기 위한 미디어 스트림의 재가공 기술이 필요하다.

그리고 보통 모바일 호스트는 유선 호스트보다 상대적으로 낮은 resource capacity와 computing power를 지니고 있다. 따라서 다양한 모바일 호스트 기기들[3]에서 수용가능한 resource에 따라 멀티미디어 스트림의 품질과 크기를 적절하게 바꾸어, 기기의 제약과는 무관하게 연속적인 스트리밍 서비스를 제공할 수 있어야 한다.

또한 서버로부터 모바일 호스트로의 전송과정에서 발생하는 지연 또한 미디어 스트리밍의 연속성을 방해하기 때문에 이에 적절한 대처가 필요하다.

이러한 이유들로 인해 결국 이동 컴퓨팅 환경에서 멀티미디어 스트리밍 서비스의 연속성을 제공하기 위한 해결책으로 프록시가 등장하게 되었다. 그러나 기존의 프록시에 대한 연구들은 트랜스코딩과 캐슁 자체의 방법론에 집중하고 있었으며, 트랜스코딩과 캐슁을 통합시켜 연동시키려는 노력들은 드물었다.

따라서 본 논문에서는 트랜스코딩과 캐슁을 통합시켜 이 둘 사이의 연동작업이 가능하도록 하는 프록시 시스템 아키텍처를 제안하였다. 또한 프록시와 모바일 호스트 사이의 bandwidth 자원에 따라 adaptive한 트랜스코딩과 캐슁을 동시에 제공하는 transco-prefix caching에 대해 제안하였다. 제시된 transco-prefix caching을 통하여 프록시와 모바일 호스트 사이의 bandwidth 트래픽의 크기를 줄이는 트랜스코딩의 장점과, 서버와 프록시 사이의 지연은 물론 트랜스코딩 과정에서의 지연까지 없애줄 수 있는 캐슁의 장점을 동시에 살릴 수 있었다.

2. Background

프록시란 대역폭이 넓은 유선망과 대역폭이 좁은 무선망을 연결해주는 하나의 중계 시스템이다. 연속적인 멀티미디어 스트리밍 서비스를 제공하기 위해 미디어 스트림을 가공하는 프록시와 관련된 주제는 보통 트랜스코딩과 캐슁, 그리고 버퍼링에 대한 연구들로 나누어진다. 트랜스코딩과 캐슁은 멀티미디어 스트리밍의 연속성을 제공해 주기 위해 각각 나름대로의 방법들을 제공한다.

트랜스코딩은 미디어 스트림을 재가공하여 화질열화(Generation Loss)를 허용함으로써 미디어 스트림의 정보량을 낮춘다. 이는 한정된 프록시와 모바일 호스트와의 bandwidth 자원을 효율적으로 사용하면서 동시에 연속적인 미디어 스트리밍을 제공해 준다. 대표적인 트랜스코딩으로는 low-path filtering[2,4] color-monochrome filtering[2], color-DC filtering[2], requantizing[2], frame dropping[2,4,10]들이 제안되었다.

caching은 서버와 모바일 호스트와의 전송 지연에 따른 멀티미디어 스트리밍의 비연속성을 해결하는 데 도움이 된다. 서버로부터의 미디어 스트림이 첫 번째로 모바일 호스트로 전송될 때, 프록시의 캐쉬에도 미디어 스트림을 같이 복제시켜 저장한다. 이 후 모바일 호스트로부터 동일한 미디어 스트림이 다시 요구될 때, 서버로부터 미디어 스트림을 재 전송받을 필요 없이 프록시에서 바로 전송받는다. 그러나, 프록시 저장용량의 한계로 적절한 replacement method 등이 요구된다. 멀티미디어 스트림을 위한 캐슁으로는 *segment-based* caching[6], pre-fetching[8], video staging[5], prefix caching[1], hot spot[7] 등이 제안되었다.

segment-based caching은 연속되는 미디어 파일을 같은 크기의 블록들로 나누어 선택의 선호도에 따라 다양한 갯수의 블록 집합들로 만들어 캐슁하도록 하였다. pre-fetching에서는 미디어 스트림을 계층적인 레이어별로 나누어 캐슁하게 한다. 그 후 스트림을 재전송 하게 될 때, 모바일 호스트의 평균 bandwidth가 캐쉬에 저장된 미디어 스트림의 평균 bandwidth를 웃돌아 그 이상을 감당할 수 있을 경우에는 서버로부터 상위 확장 레이어를 프록시에서 prefetching 하여 미디어의 품질을 단계적으로 높일 수 있도록 하였다. video staging은 미디어 스트림의 개별 프레임 size에서 모바일 호스트가 감당할 수 있는 bandwidth 분량 만큼을 뺀 나머지를, 프록시에 캐슁시키는 방법이다.

prefix caching은 서버와 프록시사이의 전송 지연분량 만큼 프록시에 캐쉥시키는 방법을 제안하였다. hot spot은 연속적인 미디어 스트림으로부터 비연속적인 몇 장의 픽쳐들을 주기적으로 뽑아내어 캐쉥하도록 하였다.

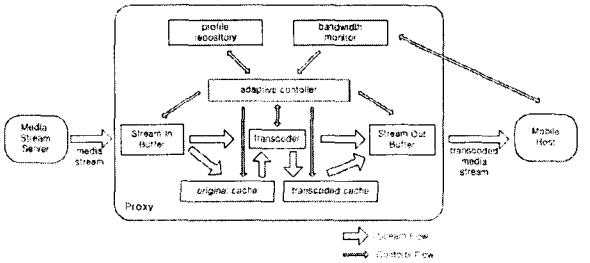
마지막으로 멀티미디어 스트림을 위한 버퍼링 메소드들로 playout control[9]과 workahead smoothing[1] 등이 제안되었다.

위의 연구들을 토대로 분석해 볼 때, 지금까지 프록시에 관한 연구들의 대부분은 트랜스코딩과 캐쉥 자체의 방법론에 집중하고 있었다. 곧 트랜스코딩과 캐쉥 자체가 각각 개별적으로 논의되었었다. 따라서 멀티미디어 스트림을 위하여 트랜스코딩과 캐쉥을 통합하여 연동시키려는 접근은 매우 드물었다. 그나마 웹 서비스를 위한 트랜스코딩과 캐쉥의 통합에 대한 제안으로 Squid[11]와 TransSquid[3]가 있으나, 역시 멀티미디어 스트림에 적용시키기에는 적합하지 않다. 왜냐하면 Squid와 TransSquid 에서의 트랜스코딩 및 캐쉥 기본 작업 단위는 작은 크기의 웹 오프젝트이다. 결국 대부분의 멀티미디어 스트림 데이터가 고용량이기 때문에 한 오브젝트 단위로 처리하기에는 문제가 있다.

따라서 본 논문에서는 멀티미디어 스트림에 적합하며, 또한 트랜스코딩과 캐쉥을 통합시켜 트랜스코딩과 캐쉥의 장점을 동시에 살릴 수 있도록 하는 프록시 시스템 아키텍처를 제안할 것이다. 또한 프록시와 모바일 호스트 사이의 bandwidth 자원에 따라 adaptive한 트랜스코딩과 캐쉥을 동시에 제공하는 transco-prefix caching에 대해 제안하려고 한다.

본 논문에서 사용되는 미디어 스트림은 mpeg-2로 인코딩 하였으며, 프록시에서 미디어 스트림을 인코딩하고 디코딩하는 등의 과정에서 적용될 트랜스코딩도 mpeg-2 표준을 이용하여 구현하는 것으로 했다.

3. 제안 프록시 시스템 구조



<그림 1> 멀티미디어 스트림을 위한 트랜스코딩과 캐쉥 통합 시스템

그림1에서 멀티미디어 스트림을 위한 트랜스코딩과 캐쉥 통합 시스템 아키텍처를 나타내었다. "*original cache*"는 미디어 스트림의 원본만을 저장하는 곳이고, "transcoded cache"는 트랜스코딩 된 스트림 데이터만을 저장하는 곳이다. Stream In Buffer와 Stream Out Buffer는 미디어 스트림이 들어오고 나가는 입출구 역할을 한다. 이 두개의 버퍼로부터 입출력되는 미디어 스트림을 고르게 만드는 역할을 하는 smoothing 버퍼링 메소드를 사용할 수도 있지만, 일단 본 논문에서는 제외시켰다. transcoder는 미디어 스트림을 트랜스코딩 하는 부분이다. profile repository는 "transcoded cache"에 저장되어 있는 미디어 스트림의 트랜스코딩 level을 저장하는 곳이다. bandwidth monitor는 프록시와 모바일 호스트 사이의 bandwidth를 주기적으로 체크하는 모듈이다. adaptive controller는 transco-prefix caching을 수행하는 핵심 알고리즘이 구현되어 있는 모듈이다.

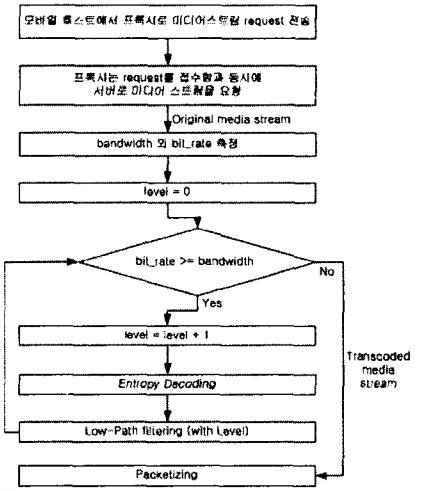
4. Adaptive Transcoding

그림2에서 프록시와 모바일 호스트 사이의 bandwidth에 따라 미디어 스트림을 adaptive하게 재가공하는 과정을 나타내었다. 미디어 스트림의 시간당 bit throughput인 bit rate와 프록시와 모바일 호스트 사이의 bandwidth를 비교하여 트랜스코딩의 level을 결정한다. 프록시와 모바일 호스트 사이의 bandwidth가 미디어 스트림의 bit rate가 더 작을 경우 transcoding level 값을 한 단계씩 상승시킨다. 이러한 일련의 반복작업을 통해 모바일 호스트와 프록시 사이의 bandwidth가 감당할 수 있을 정도가 될 때까지 미디어 스트림의 정보량을 계속 줄이도록 한다.

표1은 low-path filtering 과정에서 참조될 transcoding level table의 예를 나타내었으며, 각각의 level 값들은 특정 시스템 환경에서의 실험치로 결정하였다. 이 외에도 requantizing과 frame dropping에서도 cut-off point를 단계화 시켰다. 또한 mpeg-2 색차신호-휘도신호의 다양한 샘플링비율들을 이용하여 color-monochrome filtering과 color-DC filtering에서도 cut-off point를 단계화 시켰다.

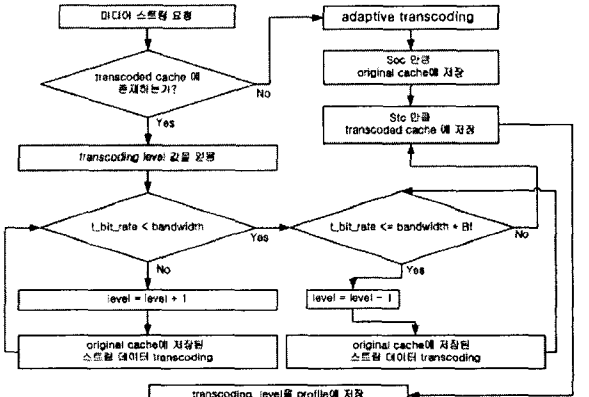
Level	Cut-off points of coefficient
0	Full quality
1	-16 ~ +16
2	-8 ~ +8
3	-4 ~ +4
4	-2 ~ +2
5	-1 ~ +1

<표 1> level of low-path filtering



<그림 2> adaptive transcoding flow : low-path filtering

5. Adaptive Caching : transco-prefix caching



<그림 3> transco-prefix caching flow

그림3은 앞의 adaptive transcoding 과정에 adaptive caching 과정을 연동시켜 통합한 것이다. Soc는 하나의 미디어 스트림이 "original cache"에 캐쉥되는 양이고, Stc는 하나의 미디어 스트림이 "transcoded cache"에 캐쉥되는 양을 나타낸다. bandwidth는 프록시와 모바일 호스트 사이의 bandwidth다. bit_rate는 서버로부터 전송되

는 스트림의 시간당 bit throughput이고, t_bit_rate는 "transcoded cache"에 저장된 스트림의 시간당 bit throughput이다. 그리고 Bf는 bandwidth boundary factor 0.1에서 0.9사이의 값을 가지며 retranscoding 여부를 결정하는 변수다. sp_delay는 서버와 프록시 사이의 전송 시간이며, t_delay는 프록시에서 트랜스코딩 하는 데 걸리는 시간이다. r_delay는 retranscoding 하는 데 걸리는 시간을 나타내며, Δ level은 retranscoding 되는 횟수를 나타낸다.

Case ① : Caching Miss in "transcoded cache"

서버로부터 프록시로 첫 번째로 전송되는 미디어 스트림의 경우, 원본 미디어 스트림을 transcoder의 입력부분으로 연결시키는 동시에, 시작 프레임부터 시작하여 서버와 프록시와의 전송 delay 분량만큼을 "original cache"로 캐쉥한다. 또한 transcoder에서 트랜스코딩이 완료된 미디어 스트림을, 서버-프록시 사이의 전송 지연과 첫 번째 미디어 스트림이 transcoding이 되어 나올 때까지 걸리는 지연을 더한 분량만큼, "transcoded cache"로 캐쉥한다.

```
Soc = bit_rate * sp_delay
Stc = Soc + bit_rate * t_delay
```

Case ② : Caching Hit in "transcoded cache" and "not retranscoding"

트랜스코딩 되어 있는 미디어 스트림의 bit rate가 프록시와 모바일 호스트 사이의 bandwidth에서의 허용 boundary안에 있다면, 원본 미디어 스트림을 다시 트랜스코딩 하지 않고, "transcoded cache"에 저장되어 있던 미디어 스트림을 모바일 호스트로 곧바로 전송한다. 동시에 서버로부터 나머지 원본 미디어 스트림을 전송받으면서 트랜스코딩 작업을 한다. 또한 "transcoded cache"에 저장되어 있는 미디어 스트림 데이터와 트랜스코딩이 되고 있는 미디어 스트림을 연결시켜서, 모바일 호스트로 연속적으로 전송한다.

```
"not retranscoding" boundary condition :
( t_bit_rate < bandwidth ) and ( t_bit_rate > bandwidth * Bf )
```

Case ③ : Caching Hit in "transcoded cache" and "retranscoding"

반면에 "transcoded cache"에 캐쉥되어 있는 미디어 스트림의 bit rate가 프록시와 모바일 호스트 사이의 bandwidth 허용 boundary 밖에 있을 경우에는, "original cache"에 이미 저장되어 있던 원본 미디어 스트림을 다시 트랜스코딩 하면서, 서버로 나머지 미디어 스트림을 요청한다. 그 다음엔, 새롭게 트랜스코딩된 미디어 스트림을 모바일 호스트로 전송하면서 "transcoded cache"로 다시 새롭게 캐쉥한다. 서버로부터 나머지 미디어 스트림이 도착하면 역시 곧바로 트랜스코딩하여 이전 스트림과 연결시켜 모바일 호스트로 전송한다.

```
③-① "retranscoding" if bandwidth <= t_bit_rate

③-② "retranscoding" if bandwidth > t_bit_rate then
                        if bandwidth * Bf >= t_bit_rate

r_delay = Σ_{j=0}^{Δlevel} j * t_delay
```

③-②는 "transcoded cache"에 저장되어 있는 미디어 스트림의 품질을 올려도 될 정도로 bandwidth 자원이 여유로울 경우를 나타낸다.

트랜스코딩 후 캐쉥 되어 있는 각 미디어 스트림에 대한 transcoding level을 별도의 profile에 저장시켜서, 미디어 품질을 결정할 때 참조하도록 한다.

6. bandwidth boundary factor(Bf)

"transcoded cache"에 저장되어 있는 미디어 스트림의 품질을 올려도 bandwidth 자원이 충분히 감당할 수 있을 때, 사용자는 두 가지의 선택을 할 수 있다. 하나는 신속한 전송을 원할 때이다. 이런 경우에는 프록시의 "transcoded cache"에 저장되어 있는 미디어 스트림의 품질을 올리기 위한 별도의 작업 없이 그대로 모바일 호스트로 전송한다. 다른 하나는 bandwidth 상황에 걸맞은 보다 높은 품질의 스트림 미디어를 원할 때이다. 이 때는 프록시의 "transcoded cache"에 저장되어 있는 미디어 스트림을 삭제하고, transcoding level을 한 단계 하향시킨 값으로 "original cache"에 저장되어 있는 미디어 스트림을 다시 트랜스코딩 하여 모바일 호스트로 전송한다. 물론 부가적으로 별도의 품질을 올리기 위한 트랜스코딩 과정의 반복으로 인한 지연이 더 발생한다는 단점이 있다.

따라서 사용자의 기호에 따라 Bf(bandwidth boundary factor)을 바꾸어 적절하게 미디어 스트림의 품질을 결정할 수 있다. 곧 미디어 스트림의 품질 향상을 위한 별도의 트랜스코딩이 더 요구되지 않는다면 Bf 값을 아주 낮게 잡으면 된다. 반대로 미디어 스트림의 품질을 향상시키고 싶다면 Bf값을 올리면 된다.

8. 결 론

지금 까지 이동 컴퓨팅 환경에서의 연속적인 멀티미디어 스트리밍을 위해 트랜스코딩과 캐쉥과정을 통합하는 프록시 시스템 아키텍처를 제안하였다. 그리고 프록시와 모바일 호스트 사이의 bandwidth 자원에 따라 adaptive하게 트랜스코딩과 캐쉥을 동시에 제공하는 transco-prefix caching에 대해 제안하였다.

실험 결과, 개별적인 트랜스코딩과 개별적인 캐쉥보다 transco-prefix caching을 이용하였을 때, 보다 나은 멀티미디어 스트리밍의 연속성을 보장할 수 있었다. 곧 transco-prefix caching을 통하여, 프록시와 모바일 호스트 사이의 bandwidth 트래픽의 크기를 줄이는 트랜스코딩의 장점과, 서버와 프록시 사이의 지연은 물론 트랜스코딩 과정에서의 지연까지 없애줄 수 있는 캐쉥의 장점을 동시에 살릴 수 있었다.

차후 좀 더 다양한 상황설정을 부여한 실험들을 통해 도출된 결과를 자세하게 분석할 예정이다. 그리고 제안한 프록시 시스템 아키텍처와 transco-prefix caching method에 스트림 트래픽을 고르게 해주는 버퍼링의 기능까지 통합시킬 예정이다. 또한 mpeg-2 스케일러빌리티(scalability) 기능과 데이터 분할(data partitioning)에 대한 확장까지 고려할 것이다.

참고 문헌

[1] S. Sen, J. Rexford, D. Towsley, "Proxy Prefix Caching for Multimedia Stream", INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , Volume: 3, 21-25 Mar 1999, Page(s): 1310-1319 vol.3

[2] N. Yeadon, F. Garcia, D. Hutchison, D. Shepherd, "Filters : QoS Support Mechanisms for Multipeer Communications", Communications(JSAC) forthcoming issue on Distributed Multimedia Systems and Technology, 3rd Quarter 1996, IEEE Journal

[3] A. Maheshwari, A. Sharma, K. Ramamritham, P. Shenoy, "TranSquid:Transcoding and Caching Proxy for Heterogenous E-Commnerce Envirionments",Proc. 12th Int'l Wrkshp on Research Issues in Data Engineering:Engineering e-Commerce/e-Business Systems (RIDE'02) 2002 IEEE

[4] C.M. Huang, P.C. Liu, R.L. Chang, "QoS Streaming Based on a Media Filtering System", Parallel and Distributed Systems, 2001. ICPADS 2001. Proceedings. Eighth International Conference on, 2001, Page(s): 661-666

[5] Z.L. Zhang, Y. Wang, David H.C.Du, D. Su, "Video Staging : A Proxy-Server-Based Approach to End-to-End Video Delievery over Wide-Area Networks", IEEE/ACM TRANSACTIONS ON NETWORKING. VOL.8.NO.4,August 2000

[6] K.L.Wu, P.S.Yu, and J.L.Wolf, "Segment-Based Proxy Caching of Multimedia Streams, "Proc.10th Int'l World Wide Web Conf.,Elsevier Science,Amsterdam,2001,pp.36-44

[7] H. Fahmi, M. Latif, S. Sedigh-Ali, A. Ghafoor, P. Liu, L. H.Hsu, "Proxy Servers for Scalable Interactive Video Support",Computer , Volume: 34Issue: 9, Sep 2001,Page(s): 54-60

[8] Reza Rejaie, H. Yu, M. Handley, D. Estrin, "Multimedia Proxy Caching Mechanism for Quality Adaptive Streaming Applications in the Internet",INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , Volume: 2, 2000, Page(s): 980-989 vol.2

[9] D.H. Nam, S.K. Park,"Adaptive Multimedia Stream Service with Intelligent Proxy", Information Networking, 2001. Proceedings. 15th International Conference on, 2001, Page(s): 291-296

[10] C.M. Huang, T.H. Hsu, C.K. Chang, "A Proxy-based Adaptive Flow Control Scheme for Media Streaming", SAC 2002, Madrid, Spain, 2002 ACM 1-58113-445-2/02/03

[11] Squid Proxy, http://www.squid-cache.org