УДК 004.773.7:37

МЕТОДИ РЕАЛІЗАЦІЇ ПОТОКОВГО ВІДЕО НА ОС ANDROID ДЛЯ ОНЛАЙН-ТРАНСЛЮВАННЯ ОСВІТНІХ ЗАХОДІВ

Саприкін А.В., Найдиш А.В.
Мелітопольський державний педагогічний університет імені Б.Хмельницького,
м. Мелітополь

Сучасний ритм життя диктує зовсім новий підхід до розваг і засобам отримання інформації: все більше людей читають інтернет–газети замість того, щоб чекати вечірнього випуску новин; погоду і афіші кінотеатрів переглядають на WAP–сайтах, а час в дорожніх пробках коротають, слухаючи музику по спеціальним мобільним розважальним каналам.

І це лише невелика частина того, що може запропонувати нам засоби потокового відео.

Потокове відео (Streaming Video) – це технологія буферизації і стиснення даних, що дозволяє вести трансляцію мультимедійного контенту (відео) через Інтернет в режимі реального часу [4].

Потокове відео - це процес перетворення відео і аудіо контенту в стислий цифровий формат з його подальшим розповсюдженням через комп'ютерні мережі. Стислі дані легко можуть бути доставлені з використанням комп'ютерних мереж в силу їх невеликих розмірів. Відео та аудіо може бути постійно потоковим, або доставлятися за запитом [1].

А також, на сьогоднішній день, потокове відео є досить популярним напрямком в споживчих технологіях.

Для того, щоб реалізувати цю технологію на Android, необхідно вирішити два завдання: конвертації відео до підтримуваного на Android форматі, відтворення відео з віддаленого джерела.

І так, перш ніж відтворити якесь відео на Android пристрої, треба це відео перекодувати в підтримуваний формат. У документації до Android чітко позначений список цих самих форматів.

Для того, що б перекодувати файли, що заливаються користувачами на сервер, або ж записати потік з TV–тюнера, буде потрібна допомога спеціальної утиліти, яка є стандартом у галузі.

Найбільш поширеним зараз способом зберігання відео є контейнер MP4 з використанням кодека H.264 AVC.

У першу чергу необхідно звернути увагу на той факт що, що Android підтримує не всі можливості кодека H.264, а лише певний набір – профіль, іменований Baseline Profile. Так, наприклад, в Baseline Profile не належать такі корисні фічі H.264 як CABAC або B–Frames [2].

А це означає, що якщо використовувати ці фічі при кодуванні відео, то програвати це відео Android не зобов'язаний, хоча і може, якщо телефон досить потужний і вендор подбав про встановлення та підтримку додаткових кодеків. Так, наприклад, відео у MainProfile без проблем програється на Samsung Galaxy SII. На телефонах ж звичайного класу (наприклад, Samsung Galaxy Ace) з`явится повідомлення про неможливість відтворення відео і помилку з кодом невірного кодека в logcat'e.

Найпростіший спосіб відтворити відео з віддаленого сервера – це завантажити його в тимчасове сховище і відтворити локально. Однак, якщо враховувати розміри сучасних відеозаписів – це не варіант.

Але ця проблема має рішення. Платформа Android пропонує нативну підтримку наступних технологій/протоколів: HTTP / HTTPS progressive streaming; HTTP / HTTPS live streaming; RTSP (RTP, SDP) [2].

Progressive streaming – це найбільш простий спосіб роздачі відео за допомогою звичайного web–серверу, що зводиться по суті до скачування заздалегідь підготовленого файлу по HTTP(S) протоколу. Особливість даного способу полягає в тому, що відтворення файлу починається не по закінченню завантаження, а як тільки буде завантажено достатньо даних.

Тут варто зауважити, що при використанні контейнера MP4, необхідно сформувати файл так, що б метадані про відео поток (moov atoms) розташовувалися на початку файлу (після атома ftyp), перед відеоданими (mdat atoms) .

Основною проблемою progressive streaming'a є відсутність можливості перемотування відео до недовантаженого фрагменту, наявність достатньої кількості вільного місця на пристрої і необхідність підтримки великого числа «товстих» клієнтів, що викачують відео, на web–сервері [1].

Відтворення за допомогою даної технології підтримується платформою Android нативної. Файл можна програти без проблем (якщо не враховувати канал зв'язку, потужність девайса і наявність вільного місця) за допомогою стандартного класу MediaPlayer.

Pseudo streaming – це технологія є логічним розширенням progressive streaming'a і дозволяє вирішити одну з його головних проблем – перегляд ще недовантаженого фрагменту. Застосовується для контейнерів MP4/FLV з кодеком H.264/AAC.

Єдиною відмінністю від progressive streaming'a в даному випадку є, той факт, що для цього необхідний спеціальний web–сервер, який з урахуванням тимчасової мітки в GET–запиту буде віддавати потрібний фрагмент відео файлу. Прикладом такого web–сервера природно може служити NGINX з його ngx_http_mp4_module.

Live streaming є досить нестандартним протоколом передачі даних від компанії Apple. Суть його зводиться до того, що файл ділиться на безліч невеликих частин, що об'єднуються спеціальним файлом – playlist'ом формату M3U8. Передача даних відбувається по протоколу HTTP(S) [4].

Ні яких проблем з перемоткою і вільним місцем на пристрої в даному випадку не виникає. Більш того, за деяких умов з'являється можливість вибирати якість відео.

Однак, з'являються і проблеми. Для поділу файлу і створення playlist'a потрібно ресурси процесора, час і місце на сервері. Для передачі файлу в мережу, також потрібен HTTP сервер (без будь–яких додаткових модулів).

Підтримка HTTP Live Streaming на нативному рівні в Android присутній починаючи з версії 3.0. За допомогою сторонніх плеєрів (DicePlayer, MX Player), можна домогтися підтримки з версії 2.2.

Real Time Streaming Protocol (RTSP) – це протокол прикладного рівня з підтримкою стану, розроблений спеціально

для передачі відео. Формат команд дуже нагадує HTTP. Самі команди нагадують кнопки на звичайному касетному магнітофоні: PLAY, PAUSE, RECORD і т.д.

На відміну від HTTP Live Streaming RTSP не вимагає розбиття фалів на дрібні частини і складання playlist'ов. Потрібні частини файлу будуть генеруватися і віддаватися клієнту нальоту.

Варто зауважити, що сам протокол RTSP не визначає спосіб передачі даних, а делегує це іншим протоколам. Наприклад, RTP.

З недоліків можна відзначити те, що при використанні RTSP–сервера для додавання/видалення файлів на сервері доведеться оновлювати його конфігурацію (список vod'ов). Для цього є telnet, але це складніше, ніж просто заливати або видаляти файли з каталогів web–сервера [4].

Відтворення за допомогою даної технології підтримується платформою Android нативно. Наприклад, за допомогою все того ж стандартного класу MediaPlayer.

Отже, для реалізації потокового відео на Android, потрібно конвертувати відео в потрібний формат, і вибрати спосіб відтворення. І, якщо, з вибором формату проблем не виникає, то різні способи відтворення відео мають, як свої переваги так і недоліки, а саме: Progressive streaming працює завжди і в будь-якому місці, але відсутність перемотування та завантаження всього файлу повністю – великий недолік; головним недоліком live streaming є нативна підтримка в Android починаючи з версії 3.0. Тобто ігнорується більше 80% користувачів с версією 2.x; також не варто використовувати RTSP, хоч він розроблений спеціально для відео. Але тут існує два моменти. По перше - необхідно постійно оновлювати конфігурацію сервера, по друге, HTTP відкритий завжди і в будь-якому місці, чого не можна сказати про RTSP.

З усіх способів відтворення відео варто виділити pseudo streaming. Він дозволяє здійснювати перемотування і при цьому не скачувати весь файл повністю.

Література

1. Википедия — свободная энциклопедия [Електронний ресурс] — Режим доступу: http://ru.wikipedia.org/wiki/ Потоковое_мультимедиа/
2. Крестьянинов М. Потоковое видео в Android [Електронний ресурс] — Режим доступу: http://habrahabr.ru/post/148754/
3. Майер Р. Android. Программирование приложений для планшетных компьютеров и смартфонов. / Р.Майер, К: Эксмо, 2011 – 672 с.
4. Почти всё о связи, учебно-информационный портал, технологии и протоколы связи [Електронний ресурс] — http://www.connect-portal.info/video_sviaz_potok.html
5. Хашими С. Разработка приложений для Android. / С. Хашими, М.: Бук-Пресс и К, 2011 – 736 с.

Аннотация. В данной статье рассмотрены проблемы обработки потокового видео. Проанализированы проблемы конвертации видео и способы их решения. Рассмотрены существующие методы восстановления видео файла. Выявлено оптимальный способ воспроизведения видео.

Ключевые слова: сервер, файл, команда, фрагмент, видео.

Анотація. У даній статті розглянуті проблеми обробки потокового відео. Проаналізовано проблеми конвертації відео та способи їх вирішення. Розглянуто існуючі методи відновлення відео файлу. Виявлено оптимальний спосіб відтворення відео.

Ключові слова: сервер, файл, команда, фрагмент, відео.

Summary. This article discusses the problem of processing streaming. The problems of converting video and how to solve them. The existing recovery techniques video file. We found the best way to play video.

Keywords: server, the file command, fragment, video.