

# BIM MP3 Instant Messenger System

An Honors Project by:  
Ben Rubinger



# Go Back a Few Months

**BIM**

<http://www.freewebs.com/bimsystem>



# In Depth

- Instant Messenger functionality  
(communicate via internet w/ text chat)
- In addition: either user can play an MP3 file. File is streamed to other user and both users listen to the song in sync while continuing with text chat.



# Comparison w/ other IM Clients

	<b>AOL Instant Messenger (AIM)</b>	<b>Microsoft MSN Messenger</b>	<b>BIM Client (Name subject to change)</b>
Text Chat	YES	YES	YES
File Transfer	YES	YES	Intentionally left out to avoid copyright infringement issues (w/ streaming mp3s)
Synchronized Streaming MP3 Playback	NO	NO	<b>YES</b>
Central Server w/ Screen names, Buddy List	YES	YES	NO
News Portal (welcome) / News Alerts / Additional Content	YES	YES	NO

**BIM**

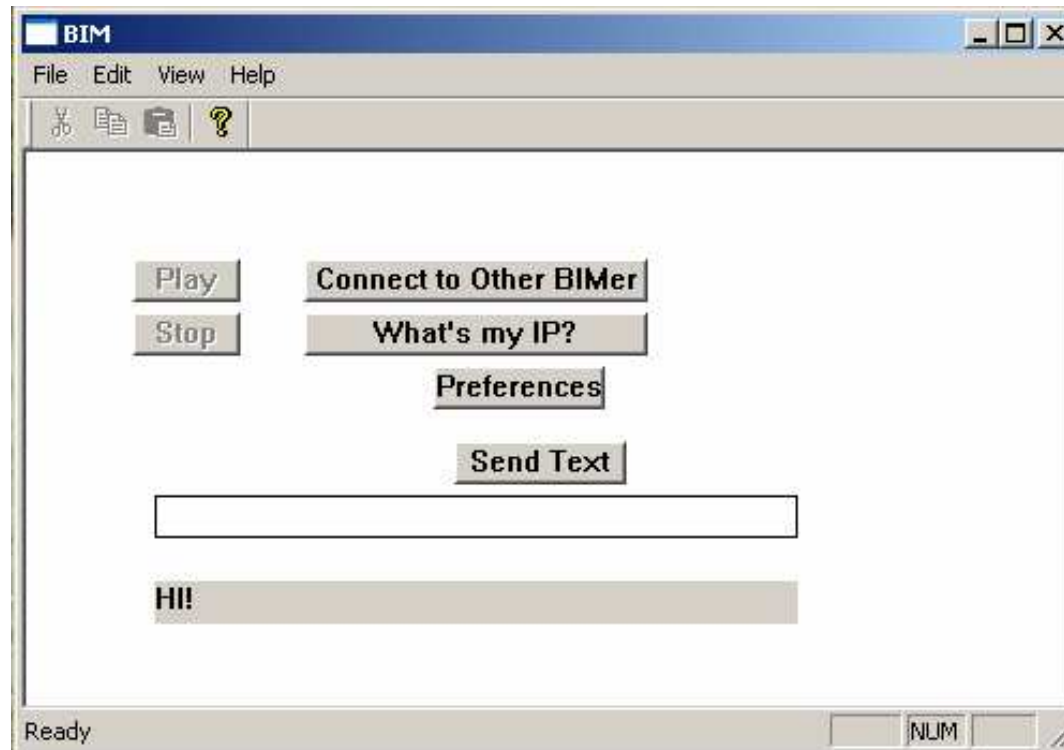
<http://www.freewebs.com/bimsystem>



# Challenges

- Learn MFC (Microsoft Foundation Classes) thoroughly
- Learn message passing w/ MFC (to control MP3 system)
- Learn XAudio Messages/Commands (interface)
- Learn windows sockets in C++/MFC

# To the Present



**BIM**

<http://www.freewebs.com/bimsystem>



# Challenge:

## Using XAudio MP3 Decoder

- Several different audio kits available on web, XAudio best, most robust choice.
- Very little documentation available (Google returning virtually no examples to follow, only same documentation as was included).
- Had to work exclusively from somewhat limited official documentation.
- In order to read MP3 from memory directly, had to use the more difficult, low-level interface of the two available interfaces in the system.



## Challenge (continued): Properly using MP3 Decoder

- Using the more difficult interface, was forced to manually handle process threading. Otherwise, once an MP3 plays, user can't control program, it exclusively plays MP3, won't respond to user input.



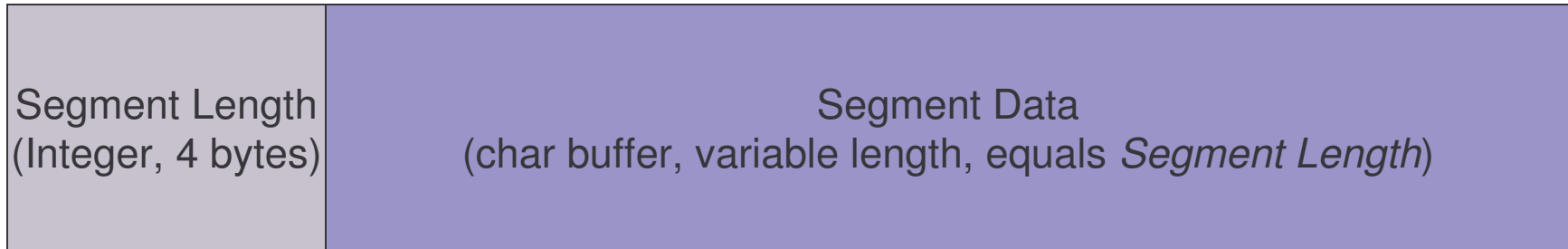
## Challenge:

# Ensuring TCP segments arrive properly

- At first, thought using *send* function would ensure that data arrived properly. In reality, when other end calls *receive*, there's no way of guaranteeing that the entire segment of data arrived. Trying to process only part of the data leads to problems.
- Solution:
  - On sending side: Implemented system that stamped every packet such that it was preceded by its length.
  - On receiving side: wait until entire length reached before processing. If only part reached, hold onto that data and then resume when you receive more data. (Continue until packet complete)



# Lower Level Protocol



*Processing*  
State Machine

Process the complete  
received packet.  
(Only segment data  
sent for processing)

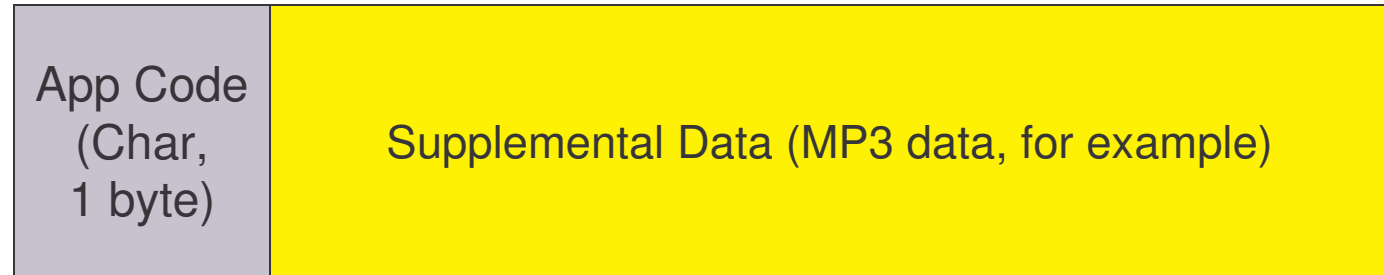
Read up to 4 bytes.  
If 4 arrive, move on.

Read up to  
*Segment Length*  
Bytes. If that many  
arrive, move on.

**BIM**

<http://www.freewebs.com/bimsystem>

# Upper Level Protocol

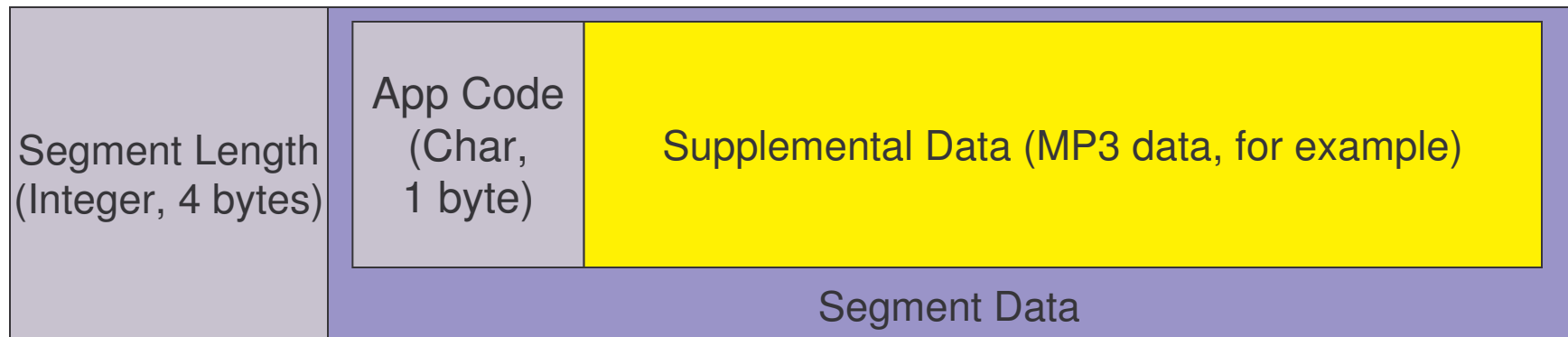


- App code informs BIM of what kind of information it is being sent so it can appropriately deal with the information. Some examples of BIM app codes:
  - D: this is a song proposal containing the singer and song names. BIM will place this information in a yes/no dialog box for the user to decide whether or not to listen.
  - T: this is a segment of text that the user has sent from the textbox.
  - M: this is a segment containing MP3 data. The data will be added to the buffer (keeping track of where to add the next chunk of data when it arrives).
  - S: this is a request to stop the playback of the audio. (Both sides stop playback)

**BIM**

<http://www.freewebs.com/bimsystem>

# Combined Layers Picture

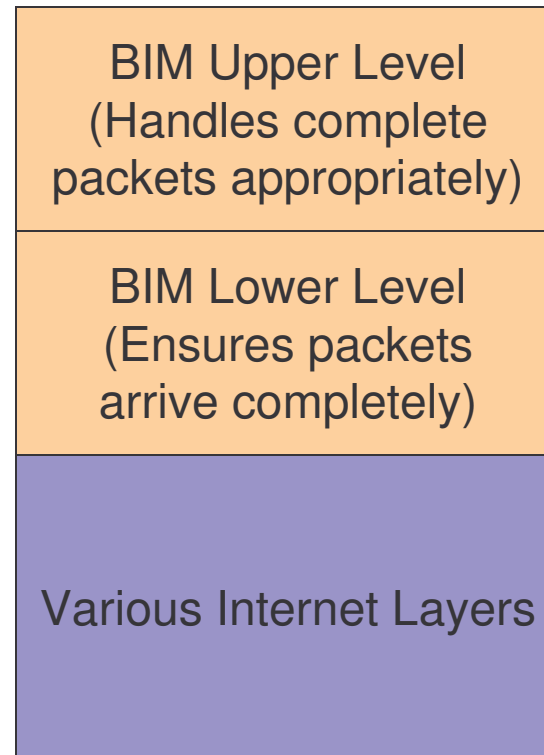


**BIM**

<http://www.freewebs.com/bimsystem>



# Internet Layers



**BIM**

<http://www.freewebs.com/bimsystem>



# Challenge:

## Circular Buffer Causing Skipping

- Limited amount of data that can be in memory at any one time.
- 2 computers both on Clark campus: Internet connection very fast.
- Result:
  - Data arrives too quickly
  - When data arrives and buffer full, some mp3 data dropped.
  - User hears periodic skips in audio



## Challenge:

# How to Keep MP3 in Memory

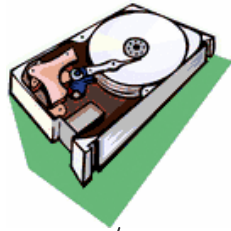
- On side receiving MP3 data, data is placed in memory buffer and decoded directly from memory (rather than a file on hard drive).
- 2 methods for this:
  - Using a circular buffer, passing data into the buffer periodically
  - Passing in the address of a linear buffer in memory (all the space for the song is pre-allocated)



# Solution: Linear Buffer

- With linear buffer, speed at which data arrives makes no difference. (No limit to speed)

# When We Hit Play...



MP3 Data sent from hosting computer's hard drive to be stored in client computer's RAM



2<sup>nd</sup> copy of MP3 data read on host computer for local playback

Once some data is in client's memory, playback will simultaneously start on both ends



BIM

<http://www.freewebs.com/bimsystem>



# If Time Weren't an Issue

- Address book system
- Multiple user (3+) connectivity
- More graphical of an interface
- Central server design & screen name system rather than IP system

**BIM**

<http://www.freewebs.com/bimsystem>