6.033 Spring 2016 Design Project

See also: DP FAQ, DP Errata. Last updates: 3/31/2016 3:10pm.

1 Due Dates and Deliverables

There are three deliverables for this design project.

1. A preliminary report of approximately 2000 words, due on March 18, 2016 at 5:00pm (see also: preliminary report writing guidelines, including a description of the cover memo).
2. A 10-to-15-minute oral presentation given to your recitation instructor, to be scheduled with your recitation instructor, for some time between April 11, 2016 and April 22, 2016. The oral presentation will assess your progress and provide some feedback prior to your final report submission.
3. A design report of approximately 5000 words, due on May 6, 2016 at 5:00pm.

Each deliverable will have specific guidelines, which will be linked above.

The preliminary and final report should be submitted via the online submission site. As with real-life system designs, the 6.033 design project is under-specified, and it is your job to complete the specification in a sensible way given the stated requirements of the project. As with designs in practice, the specifications often need some adjustment as the design is fleshed out. Moreover, requirements will likely be added or modified as time goes on. We recommend that you start early so that you can evolve your design over time. A good design is likely to take more than just a few days to develop. A good design will avoid unnecessary complexity and be as modular as possible, to enable it to evolve to changing requirements.

Late submission grading policy: If you submit any deliverable late, we will penalize you one letter grade per 48 hours, starting from 5 pm on the submission day. For example, if you submit the report anywhere from 1 minute to 48 hours late, and your report would have otherwise received a grade of "A", you will receive a "B"; if you submitted 49 hours late, you will receive a "C".

Large systems are never built by a single person. To that end, you will be working in teams of three for this project. Part of the project is learning how to work productively on a long-term team project. All three people on a team **must** have the same TA (your team may include people from either of your TA's sections).

Note that although this is a team project, some of the deliverables have individual components. See the individual assignment links for more information.

2 Introduction

Wireless networks such as MIT's are made up of access points (APs), which are able to communicate with wireless devices—such as laptops—in range.

At a high level, user devices—"clients"—connect to the MIT network via wireless access points, or APs. These APs also have wired connections to the rest of the Internet. When a client transmits data to `google.com`, say, that data travels from the client, through one of the APs, to the rest of the Internet.

When your device tries to connect to a wireless network, it's not uncommon for it to be in range of multiple APs on the same logical network; in that case, your device will connect to the one with the strongest signal (often that is the AP that is nearest to you).

However, the strongest AP may not be the best AP in terms of performance. Signal strength doesn't capture, for instance, congestion on the network. If your application requires more throughput, it may be better to connect to a less-congested AP with a weaker signal.

Your job is to design a system that gives users of MIT's wireless network the ability to connect to an AP that offers acceptable performance, while also enabling some additional network- and Institute-wide goals.

- Assuming that a wireless device has defined its performance requirements, your system will allow the device to connect to an in-range AP with acceptable performance. If no acceptable APs are in range, your system will recommend an acceptable AP nearby.
- Your system will allow for high network utilization even under periods of heavy load. Utilization should be high across the entire network, not just at, say, a single AP.
- Your system will allow IS&T to collect data from the APs, to aid in their network management. For instance, they will be able to locate APs that are consistently congested and add additional APs to help lighten the load.

3 Existing Infrastructure

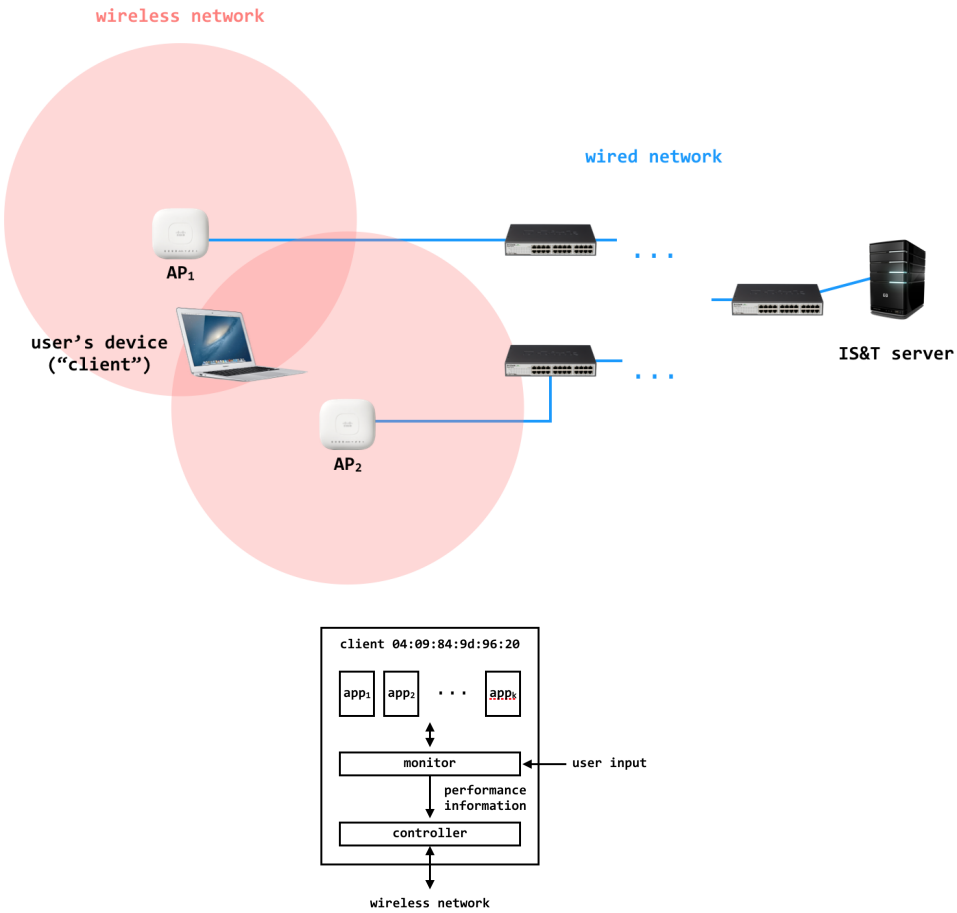
Your system will deal with clients at MIT trying to connect to the MIT network (that is, the network with SSID `MIT`). You don't need to worry about other networks (`MIT SECURE`, `MIT GUEST`, etc.).

3.1 Devices

3.1.1 Clients

Each client has a unique address known as its MAC address. This address is a 48-bit value, often written as 12 hexadecimal digits: `11:22:33:AA:BB:CC`. Client MAC addresses do not change over time.

You can assume that the client—whether it's a laptop, a phone, or some other wireless device—is already equipped with two software modules:



- The **monitor**, which monitors applications running on the client and their performance. "Performance", here, means throughput (a unit of data over a unit of time). The monitor determines the *required throughput* of the applications and some additional statistics about the current *achieved throughput*. The throughput that applications require does not change; the throughput they actually achieve may. (We're assuming that the required throughput doesn't change to make your lives easier; in reality, application demands change over time.)

  The monitor will also receive input if the user is unhappy with the level of performance they're experiencing. More on that in the next section.

- The **controller**, which communicates with the APs and determines which one to connect to. Every second, the controller polls the monitor to determine the current level of performance.

The amount of throughput that a client requires represents its maximum requirement. At any given point, though, the client may not send that much data (e.g., just because a client's maximum requirement is 10Kbytes/sec does not mean it will send ten kilobytes every second).

In your system, clients fall into one of two categories: large clients that require a large number of bytes per second and use close to that amount every second, and small clients that require a smaller number of bytes per second but whose usage varies more. An example of a large client is one where the user is watching Netflix. An example of a small client is one where the user is

checking email.[1]

A large part of your job is to design the rest of the controller: how it determines which AP to connect to, how/when it communicates with APs, etc. You can assume that the client has a reasonable amount of storage available for you to use; you should specify how much storage you require.

3.1.2 Access Points

Every AP in the system belongs to the same network: the `MIT` network. Like the clients, APs also have unique 48-bit MAC addresses. Up to 128 clients may be connected to a particular AP, but more than that may be in range of an AP at a particular time. The average range of an AP is 125ft; individual ranges will vary depending on the environment.

Each access point has 64 MB of flash storage and 32 MB of RAM, a simple UNIX-like operating system, and a 1.2 GHz processor.

3.1.3 Servers

IS&T has dedicated one machine to your system. This machine has 10 TB of storage and is connected to the Internet; APs and clients can both communicate with it (client communications would, of course, go through an AP).

This server stores some meta-information about the network, and your system will also need to store historical performance data about each AP on the server. Depending on your design, you might find it useful to store other information there, too.

3.2 Networks

3.2.1 Wireless Networks in General

The Appendix contains a detailed primer on wireless networks in general. Briefly:

- Direct communication between the clients and APs happens at the link-level, which means that this communication deals with frames between the two nodes, connected by a physical layer (radio signals). Each frame includes a link-layer header, which includes a few things, in particular the MAC addresses of the source and destination.
- Wireless is a broadcast medium. In general, devices only process frames that are addressed to them, but it is possible to broadcast information out to all in-range devices.
- Wireless networks composed of multiple APs—for instance, the `MIT` network—will almost always have APs that operate on different channels. This means that when a client is communicating with a particular AP, it must do so on that AP's channel. If a client on channel 6 is communicating at the same time as a client on channel 11, their communications will not interfere, but they will also not be able to communicate with each other.

[1] This breakdown into two categories is another simplifying assumption for this design project, but it is not entirely unrealistic (see http://networkheresy.com/2013/11/01/of-mice-and-elephants/).

Recall from 6.02 that two entities in range of each other cannot transmit at the same time on the same channel; the two frames will collide and both will be lost. You can assume that there is already a MAC protocol in place to mitigate these types of collisions.

3.2.2 Your Wireless Network

In this system, you're using a network technology that is similar to, but not exactly like, 802.11; let's call it 802.033.

In 802.033, when a client is in range of more than one AP, you can assume that, by default, all of the APs are operating on different channels. This means that a client communicating with an AP on channel 1 will not interfere with transmissions to the other APs that are in range.[2]

There are 11 channels available for clients to send application data on, plus a 12th reserved channel, with limited bandwidth, described in the next section. A client determines what APs are in range on a particular channel by listening on that channel; you can assume that after listening on a channel for 30ms, the client will discover whether there is an in-range AP operating on that channel (with the exception of the 12th channel). APs ensure this by broadcasting a heartbeat message every 30ms on their channel. Assume that clients have the ability to measure signal strength from these heartbeats (if you should need to), though your system cannot change these heartbeats.

The frames that are transferred between the client and AP take the format:

```
|src addr | dst addr | meta | data |
```

Where:

- `src addr` is the 48-bit source address (either the client's, the AP's, or a broadcast address).
- `dst addr` is the 48-bit destination address
- `meta` is an 8-bit value that specifies that the frame is either application data (`meta = 00000000`) or control data (value set defined by you).
- `data` contains the actual data of the communication (which, as described above, is typically part of a network-layer packet, which is part of a transport-layer packet, which contains the application data).

When the client is sending application data, you can assume that `meta` is set to all 0's. When the AP receives a frame with a `meta` value of all 0's, it will process that frame and send it on to the rest of the Internet. You do not need to worry about how that is done; it is part of the AP's normal operation.

In your system, you will need to define some extra communication between the clients and APs; we'll refer to that as **control** data (vs. application data). Control data will have a `meta` value not equal to zero.

You can assume that a link-layer reliable transport protocol exists, and that delivery of a frame between the client and AP is guaranteed. Most APs can support a maximum data rate of 54 Mbits/sec, but some are **high-capacity**: they can support a maximum data rate of 96 Mbits/sec.

[2] This lack of interference is not strictly true in practice, but you can assume it for this project.

You can assume that every AP knows whether it's a high-capacity AP, and that this information is also stored on IS&T's server.

3.2.3 Wired Network

As part of their connection to the "rest" of the Internet, APs can communicate with IS&T's server via a wired network. The wired network, thankfully, is much less involved than the wireless network. You can assume that it operates as a "normal" network, with a reliable-transport protocol already in place. The IS&T machine may fail for up to two minutes at a time, after which it will recover (upon recovery, no data stored on the server will have been lost; you are not responsible for describing this recovery process).

The average round-trip network latency between an AP and the server is 5-10 milliseconds. The maximum throughput between an AP and the server is 1Gbit/second, but the average throughput may be less if other users are using the network.

4 Interfaces

4.1 Client Controller

You can assume that the controller knows a 32-bit integer, R (**r**equired throughput), representing the maximum number of bits that the client will send or receive over a one-second period; i.e., the number of bits sent to or from a client in any one-second period will not be greater than R. You can also assume that the client knows whether it is large or small.

When the controller polls the monitor, it receives two pieces of information:

- A 32-bit integer G, representing the number of bits the client's applications generated in the past second.
- A 32-bit integer A, representing the number of (data) bits that the client actually sent in the past second.

It is possible for A to be strictly less than G; for instance, if the conditions at the client's AP have changed, so that bandwidth that was available earlier is no longer available. This may or may not happen frequently, depending on how you design your system. For example, if you allow only one client to connect to each AP, conditions at the client's AP will not change. You do *not* need to consider how the network beyond the AP affects performance. Assume that the link between the AP and the client is the bottleneck.[3]

If A is less than G, it is possible that the user will be unhappy with their perceived level of performance. Users can indicate unhappiness via the client software (e.g., via an "I'm unhappy!" button), and that information will be communicated to the controller via the monitor. Your system should strive to keep users happy; we give specific user-perceived performance requirements in the next section.

[3] This is *not* always the case in the real world!

4.2 APs

The AP can quickly (near-instantaneously) calculate the following statistics:

- Instantaneous outgoing-queue size, in bytes (i.e., how many bytes are currently in the queue that stores outgoing frames).
- Instantaneous incoming-queue size, in bytes (i.e., how many bytes are currently in the queue that stores incoming frames).
- Total number of bytes transferred since the last reset, which rolls over once the total number of bytes exceeds $2^{32} - 1$.
- The number of users currently connected.

Since the AP can calculate these numbers, it can also store them (though you would need to define how and when that occurs).

The AP also has the ability to install filters, which will cause it to automatically store all frames that match the filter. The filters must be boolean expressions that use information in the frame header; for instance, "all frames with source address `11:22:33:AA:BB:CC`" or "all frames with meta value `00000001`". The AP can install up to 8 such filters and can process up to 150 frames per second in total from these filters.

4.3 Communications between Clients and APs

When a client enters the network, it does two things: discovers which APs are in range and chooses that AP that it wants to connect to (and, subsequently, send its application data through). Once a client is connected, it may optionally choose to send (or receive) messages to (from) the AP; these messages might include information about the network conditions, for instance. A client may also choose to send a particular message when it decides to disconnect. Remember that only 128 clients may be connected to a particular AP at once.

These steps require **control** data to be sent between the client and the AP. There are two ways you can do that.

1. Directly through the normal data channel that the AP operates on. Given the frame format discussed in the previous section, an AP will consider any frame with `meta = 00000000` to be an application frame that it needs to process and forward on to the rest of the Internet. You can define how the AP reacts to `meta` values other than 0.
2. Using channel 12, which has a bit rate of 1 Kb/sec. The only communications that happen on this channel are the ones your system defines, i.e., you do not need to worry about application data interfering here. For an AP to communicate or listen on channel 12, it must stop transmitting on its data channel.

   Note that, if you choose to use channel 12, you will need to specify how a client discovers APs operating on that channel. On their data channel, APs broadcast heartbeats every 30ms to allow for discovery; that does not happen by default on channel 12. You can assume that the heartbeats have a `meta` value other than `00000000` so that they can be differentiated from

other packets.[4]

Note that direct communication between the clients and the APs will consume part of the channel, so be wise about how frequently such communication happens. You are not allowed to change the AP's default data channel; i.e., if an AP transmits data on channel 6, your system cannot change it to channel 7. It takes 5msec for an AP to switch from its data channel to the broadcast channel (and another 5msec to switch back).

If none of the in-range APs offer satisfactory performance, the system must recommend a nearby AP for the user to move to, if one exists.

4.4 Servers

IS&T's machine comes pre-populated with a table that stores the MAC addresses and location (building number, room, GPS latitude in decimal degrees, GPS longitude in decimal degrees) of each AP in the system, as well as whether it's a high-capacity AP. Given a MAC address, one can query the server for its location. This table takes less than one MB to store, so you don't need to worry about how it affects the available space on IS&T's server.

You can assume that the AP knows the address of the IS&T server and can communicate with it via the wired network. You will need to specify the details of such communication: what is sent and when.

Your system may store additional data on IS&T's server if you wish, so long as it meets the minimum requirements in the next section.

5 Objectives

5.1 Requirements

User performance

- Keep user unhappiness to a relative minimum. In particular, no client should find that its user is unhappy with performance more than ten times per hour. It's safe to assume that users will wait at least fifteen seconds between clicks (i.e., you don't have to worry about a case where a user makes multiple clicks in quick succession, giving your system no time to react).
- When no acceptable APs in range, but there is an acceptable AP within roughly 500ft of the user, the system must recommend it to the user. (Note: you do *not* need to geolocate the user down to their precise latitude and longitude. A rough estimate is fine.)
- The system should minimize disconnections/switches between APs. Switching a client between APs or repeatedly disconnecting it from the network can cause bad things to happen to the transport-layer traffic, and can result in up to one second of poor performance. Thus, switching a client is likely to result in an unhappy user, at least initially. Depending on the situation, this initial unhappiness may be acceptable if overall network utilization improves.

[4] We added this clarification late in the project. So as not to impede anyone's designs, we'll allow you to specify exactly what the `meta` value of heartbeats is, if you need to.

- Clients should connect to the AP relatively quickly upon entering the system. It is not acceptable for a client to take more than a couple of seconds to connect to an AP.

Network performance

- The system must maximize utilization: if the in-range APs have room for a new client, the client should connect even if it means shifting other clients around.
- The system should scale up to the total number of clients/devices in range.

IS&T

- IS&T should receive a record of the number of bytes transferred by each AP every second, as well as an estimate of the number of unique users connected in each second. "Estimate" is intentionally under-specified. You could take a single sample each second, take multiple samples and calculate the average, etc. Whatever you want, as long as you justify your design choice. If the server fails for any period of time, it should still receive the data collected by APs during that time (presumably it will receive such data after it recovers).

Scale

- Your system must work at the scale of MIT's campus: up to 25,000 total clients using the system at once (i.e., actively trying to connect to MIT's network), and roughly 4000 total APs in the system.

Security

- Consider your design in light of security. Since client MAC addresses do not change, the system youre building may provide an attacker with a way to track clients geographically (i.e., to figure out where they often are during what times of the day, and to potentially even figure out who they are when cross-referencing that information with either a class schedule or housing information).

  If your system stores data on IS&Ts server—or in other places in the network—that would allow an attacker to track a client in this matter, specify what data that is, and why you need to store it. Do you believe that the security trade-off is worth it?

  You do **not** need to re-design your system to prevent such an attack.

5.2 Use-cases

When designing your system, you should consider how it works under each of the following scenarios.

- A single client—large or small—connecting to a largely under-utilized part of the network.
- Hundreds of clients in one room (e.g., 26-100) all connecting to the network at once. In this scenario in particular, consider how the APs' performance capabilities change as users (rapidly) connect.
- An area of the network with high churn, where new users are constantly connecting to an AP, but disconnecting after 5–10 seconds. This scenario can occur with an AP in a space that users move through quickly, e.g., a hallway. The majority of these clients will be small, but there will be many of them at once (up to 100 new connections every 10 seconds).

Also consider how your system will scale beyond MIT. Suppose that a larger university wants to adopt your system, or imagine that MIT grows—perhaps taking over a liberal arts institution nearby. Will your system be able to adapt to such growth?

5.3 Summary of Design Trade-offs

As you design your system, you will be faced with the following trade-offs.

- How risky is your system in allowing new connections? You can design a system where each AP knows the maximum amount of data that its clients will send at a given time, but the amount that they *actually* send may be less. How do you choose whether to exploit that available, but not guaranteed, bandwidth?
- How risky is your system in terms of client performance? Will you allow users to be occasionally unhappy in service of a more balanced, well-utilized network? How quickly will your system react if a user reports that they're unhappy?
- What module makes what decisions? For instance, does the client alone get to decide which AP it connects to? Does the AP have any say in that decision? Do *other* APs have say in that decision? Does IS&T's server? What are the performance trade-offs with your approach? How does your approach handle a part of the network that is under stress? All of these issues relate to making global decisions vs. local ones.
- A related global vs. local issue: What data might be useful for your system to store (beyond IS&T's requirements), and what module(s) should store it?

5.4 Summary of Modules

- **Client:** Has access to performance requirements and a measure of achieved performance. Receives user input when performance levels drop too low.
- **APs:** Each AP has can process and store a small amount of data.

  Clients connect to APs via the wireless network. Much of the traffic between them is driven by client applications, but your system will almost certainly need to send some control traffic between the client and the AP.
- **IS&T Server:** Knows the MAC address and location of every AP. Your system will store historical data here to meet IS&T's requirements. You may choose to use this server to store additional data.

  APs can communicate with the IS&T server via the wired network.

6 Appendix - Wireless Networks

The communication between clients and APs is a bit different than communication between normal Internet endpoints, because APs are "Layer 2" devices.

Most of the traffic from the clients is driven by the applications themselves; we'll refer to this as application data. This would be the data that, say, comprises the Netflix video that a user is watching or the email that they're sending.

That application data is sent via a particular transport-layer protocol, which divides the data into packets and adds a transport-layer header. We talked a lot about this in 6.02; this is the header that would specify, for instance, the sequence numbers in the reliable transport protocol.

However, that transport-layer packet is sent over a particular type of network, which requires its own header to deal with addressing and routing. Thus, the transport-layer packets are used as data in *network-layer* packets, which have their own header. The most common network-layer protocol is IP (the Internet Protocol). We refer to the network layer as Layer 3.[5]

The Layer-3 traffic, similarly, is used as the data in Layer 2: the link layer. At Layer 2, the network-layer data is coalesced into a stream and sent as frames, which include a link-layer header. This header includes a few things, in particular the MAC addresses of the source and destination.

As mentioned, APs are Layer 2 devices. They operate at the link layer, and don't parse any of the information given by higher layers (e.g., they don't inspect frames looking for a TCP header to interpret). Direct communications between a client and an AP, then, are done at the level of frames.

***

A second thing that makes communication on wireless networks a bit different than usual is the fact that wireless is a broadcast medium.

Because wireless is a broadcast medium, any devices—clients or APs—on a particular channel that are in range of each other will be able to hear each others transmissions. A device will only process frames that are addressed to it. If a device transmits on a particular channel to the address `FF:FF:FF:FF:FF:FF`, however, all devices in range on that channel will process that frame (this address is know as the "broadcast address").

Recall from 6.02 that two entities in range of each other cannot transmit at the same time on the same channel; the two frames will collide and both will be lost. A MAC protocol does the job of mitigating these types of collisions.

Wireless networks composed of multiple APs—for instance, the `MIT` network—will almost always have APs that operate on different channels. This means that when a client is communicating with a particular AP, it must do so on that AP's channel. If a client on channel 6 is communicating at the same time as a client on channel 11, their communications will not interfere, but they will also not be able to communicate with each other.[6]

[5] https://en.wikipedia.org/wiki/OSI_model

[6] If you want to see the traffic that exists on channels around you, there are a lot of apps that do this. "Wifi Analyzer" for Android (free), "WiFi Explorer" for OSX (not free, in fact disappointingly expensive), and others.