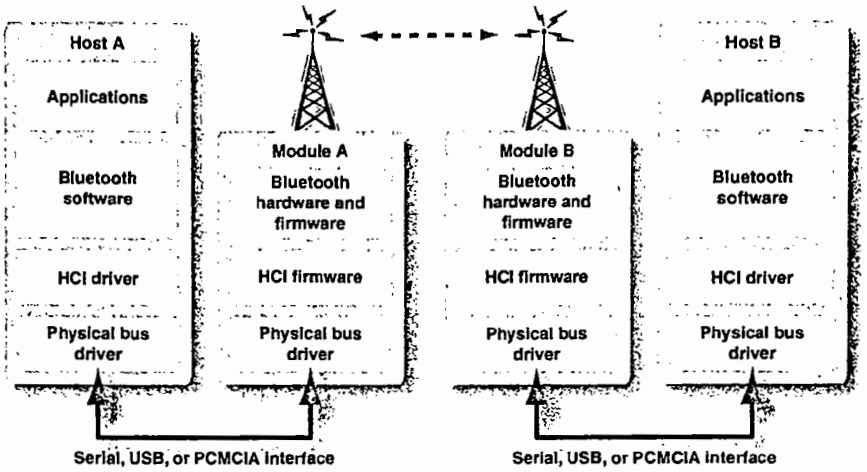


Figure 1-9
An add-on module is one method that Bluetooth can be connected to a computer.

The OSI Model

Take a look again at the Bluetooth-to-host interface in Figure 1-9. A very complex process must occur just to get data from the host application to the radio in the module, across the wireless link, and to the destination host. How should the data be structured? How fast should the data be sent? And (*gasp*) how should the link be established and configured? After thinking about all of these factors together, it's tempting to give up in despair and return to a simpler life of designing vacuum tube oscillators.

In 1977, the *International Standards Organization* (ISO) established a subcommittee to research the need to develop a standardized, layered approach to general computer communications. This work culminated in 1982 with the *Open Systems Interconnection* (OSI) reference model, shown in Figure 1-10. By working through the layers of the model, a designer can gradually create an entire computer communication system without becoming overwhelmed, and (theoretically, at least) a particular layer can be changed without affecting the other layers.[3] The downside to using this method is that redundancy, along with its resulting inefficiencies, can creep into a design. We shall discover this characteristic when we examine the details of Bluetooth packet structure in Chapter 4, "Baseband Packets and Their Exchange."

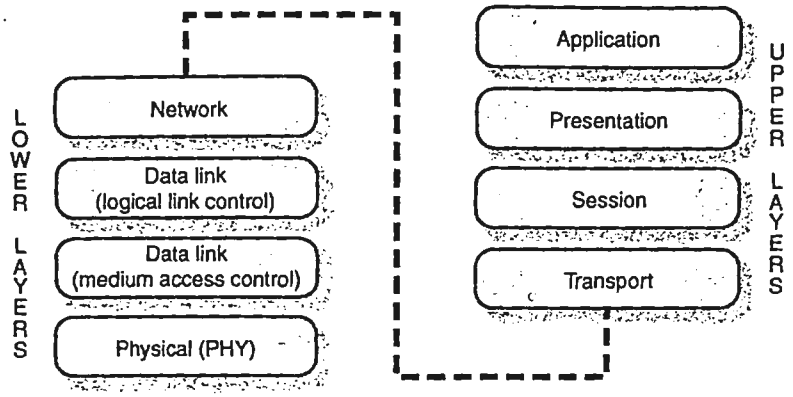


Figure 1-10
The OSI reference model provides a standardized, layered approach to computer communications.

The *physical layer* (PHY) contains the actual physical interface and the rules for its use. In Bluetooth, the PHY is RF and the modulation, and detection processes are listed in the specification. The PHY is made reliable, and link connection and detachment rules are provided by the *data-link layer*. This layer contains *Media Access Control* (MAC), which is a set of rules that determine the structure of basic data packets and how they are sent, and the *Logical Link Control* (LLC), which provides the protocol for link establishment and detachment. The *network layer* provides transparent transfer of data between transport entities on each end of the communication link. A properly implemented network layer relieves the transport layer from requiring any knowledge of the method by which data moves from source to destination. In other words, a Bluetooth device can appear to be a serial cable to the transport layer.

Layers four and above in the OSI model are called *higher layers* (profound, yes?), and their functions start to become less well defined. The *transport layer* includes optimization routines and other *quality of service* (QoS) methods for efficient data exchange, and the *session layer* contains the method for controlling dialog between applications on either end of the link. Finally, the *presentation layer* resolves differences between format and data representation between entities, and the *application layer* provides the means by which applications can access the OSI environment. As we move up the OSI layers, their implementation gradually changes from hardware, to firmware, and finally into software. The Bluetooth protocol stack exhibits the same behavior.

Bluetooth Protocols

Now that we've studied the OSI model, we can move on to the Bluetooth protocol stack shown in Figure 1-11. It's at once apparent that the Bluetooth protocol stack doesn't conform to the OSI model exactly, but the layers are still there and gradually transition from implementation in hardware and firmware (lower layers) to software (higher layers). If each of these groups of layers are separate entities, such as a PC card and laptop computer, then they can communicate with each other through the HCI. HCI provides paths for data, audio, and control signals between the Bluetooth module and host.

The radio completes the physical layer by providing a transmitter and receiver for two-way communication. Data packets are assembled and fed to the radio by the baseband state machine. The link controller provides more complex state operations, such as the standby, connect, and low-power modes. The baseband and link controller functions are combined into one layer in Figure 1-11 to be consistent with their treatment in the Bluetooth Specification 1.1. The link manager provides link control and configuration through a low-level language called the *link manager protocol* (LMP).

The *logical link control and adaptation protocol* (L2CAP) establishes virtual channels between hosts that can keep track of several simultaneous sessions such as multiple file transfers. L2CAP also takes application data and breaks it into Bluetooth-size morsels for transmission, and reverses the process for received data. *Radio frequency communication* (RFCOMM) is the Bluetooth serial port emulator, and its main purpose is to trick an application into thinking that a wired serial port exists instead of an RF link. Finally, the various software programs that are needed for the different Bluetooth usage models enable a familiar application to use Bluetooth. These include *service discovery protocol* (SDP), *object exchange* (OBEX), *telephony control protocol specification* (TCS), and *Wireless Application Protocol* (WAP).

Aside from data communications, Bluetooth has a special provision for real-time, two-way, digitized voice as well. Once these voice packets are created by an application, they bypass most of the data protocol stack and are handled directly by the baseband layer. This prevents unacceptable delay between the time the packets are created and the time they arrive at their destination. Control of the Bluetooth module usually proceeds from the application through HCI to the module, also bypassing the protocol layers used for handling the data communication process itself.

The Bluetooth radio and the baseband/link controller consist of hardware that is typically available as one or two integrated circuits. The firmware-based link manager and one end of the host controller interface

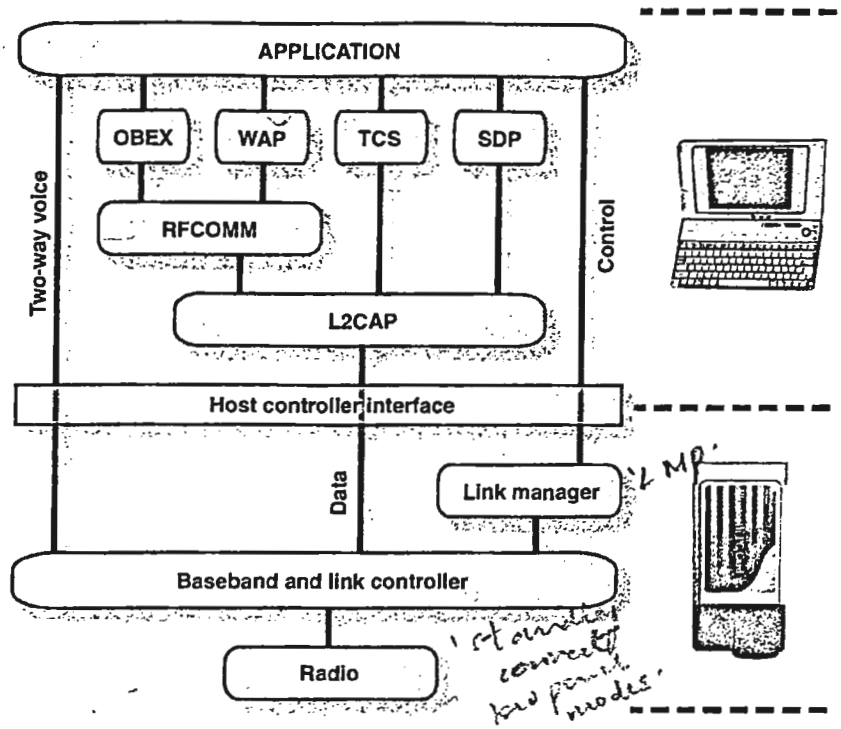


Figure 1-11
The Bluetooth protocol stack

perhaps with a bus driver for connection to the host, complete the Bluetooth module shown in Figure 1-9. The remaining parts of the protocol stack and the host end of HCI can be implemented in software on the host itself.

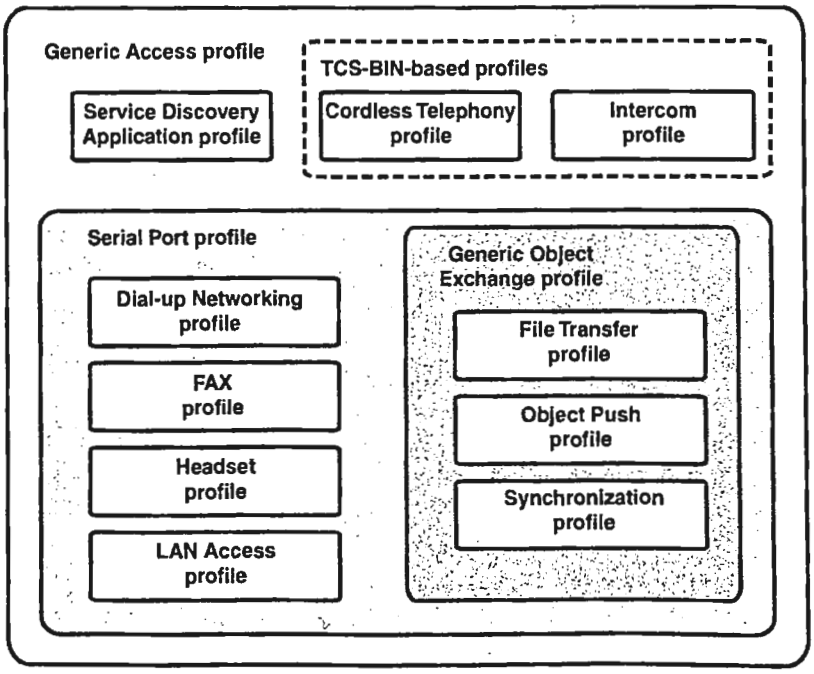
Bluetooth Profiles

Whereas protocols provide the basic building blocks for Bluetooth operation, the profile is what gives a Bluetooth-equipped device its personality. Do you want the device to be a headset? Use the headset profile. A cordless phone? Use the cordless telephony profile. The purpose of profiles is threefold:

- Lots of options are reduced to those needed for a specific function.
- Procedures for a specific function can be taken from a set of base standards.
- A common user experience is provided across devices from different manufacturers.

Several profiles existed in the first release of the Bluetooth specification, and these original profiles and their interaction with each other are shown in Figure 1-12. For example, if a Bluetooth-equipped device is to have the ability to perform automatic file synchronization, then the Generic Access profile, Serial Port profile, Generic Object Exchange profile, and Synchronization profile will all play a role in the device's capabilities. Profiles can be envisioned as a "vertical slice" through the Bluetooth protocol stack, in which a subset of capabilities in each layer is selected for the particular Bluetooth function being developed. Automatic file synchronization, for example, doesn't require the use of two-way real-time audio, so implementing audio isn't necessary for that application. Each usage model has its own corresponding set of profiles. Other profiles continue to be added as they attain SIG approval. We will take a more detailed look at profiles and how they are constructed in Chapter 11.

Figure 1-12
Examples of Bluetooth profile interaction



DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.