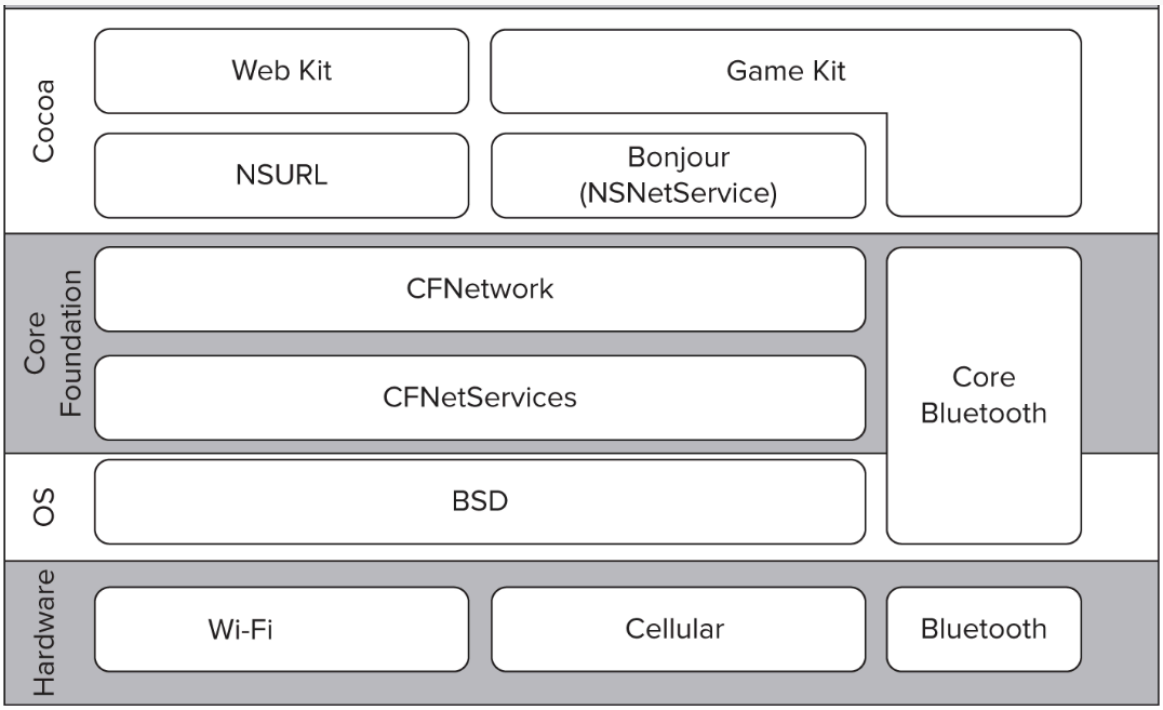
iOS中的网络框架



- Cocoa层，包含了用于URL加载的OC API、Bonjour、Web Kit与Game Kit
- Core Foundation层，是一套C API，其中包含了CFNetwork，这是大多数应用级别的网络代码的基础。CFNetwork在CFStream与CFSocket之上提供了一个简单地网络接口，是针对BSD socket的轻量级封装
- OS层的BSD socket严格使用C来实现，提供了与远端设备与服务器通信的完全控制能力

> 从上层框架到下层移动，会获得更为严密的控制，但却也失去了上层提供的易用性与抽象能力。比如，BSD层的socket是无法访问系统范围的VPN，也无法激活Wi-Fi和蜂窝无线网络

NSURLConnection

NSURLConnection是个Cocoa级别的API，构建在NSSream之上，设计时针对4个常见的URI模式进行了优化支持，文件、HTTP、HTTPS与FTP。NSURLConnection支持同步和异步两种请求

Game Kit

Game Kit并不需要网络基础设施就能使用，能够创建自组（ad-hoc）的Bluetooth Personal Area Networks（PAN）来实现点对点的通信

Bonjour

Bonjour是Apple对零配置（zeroconf）网络的实现，通过名字、服务类型与域这几个元组来引用服务

NSStream

NSStream是一个Cocoa级别的API，构建在CFNetwork之上，作为NSURLConnection的基础，旨在完成一些底层的网络任务。NSStream可以支持使用telnet或SMTP等NSURLConnection不支持的协议进行通信。NSStream是异步的，通过NSStreamDelegate实现通信更新

CFNetwork

CFNetwork API位于基础的BSD socket之上，与BSD socket之间的主要差别在于CFNetwork集成了运行循环。CFNetwork构建在Core Foundation层的CFSocket和CFStream API之上

HTTP协议

协议

协议是指计算机通信网络中两台计算机之间进行通信所必须共同遵守的规定或规则

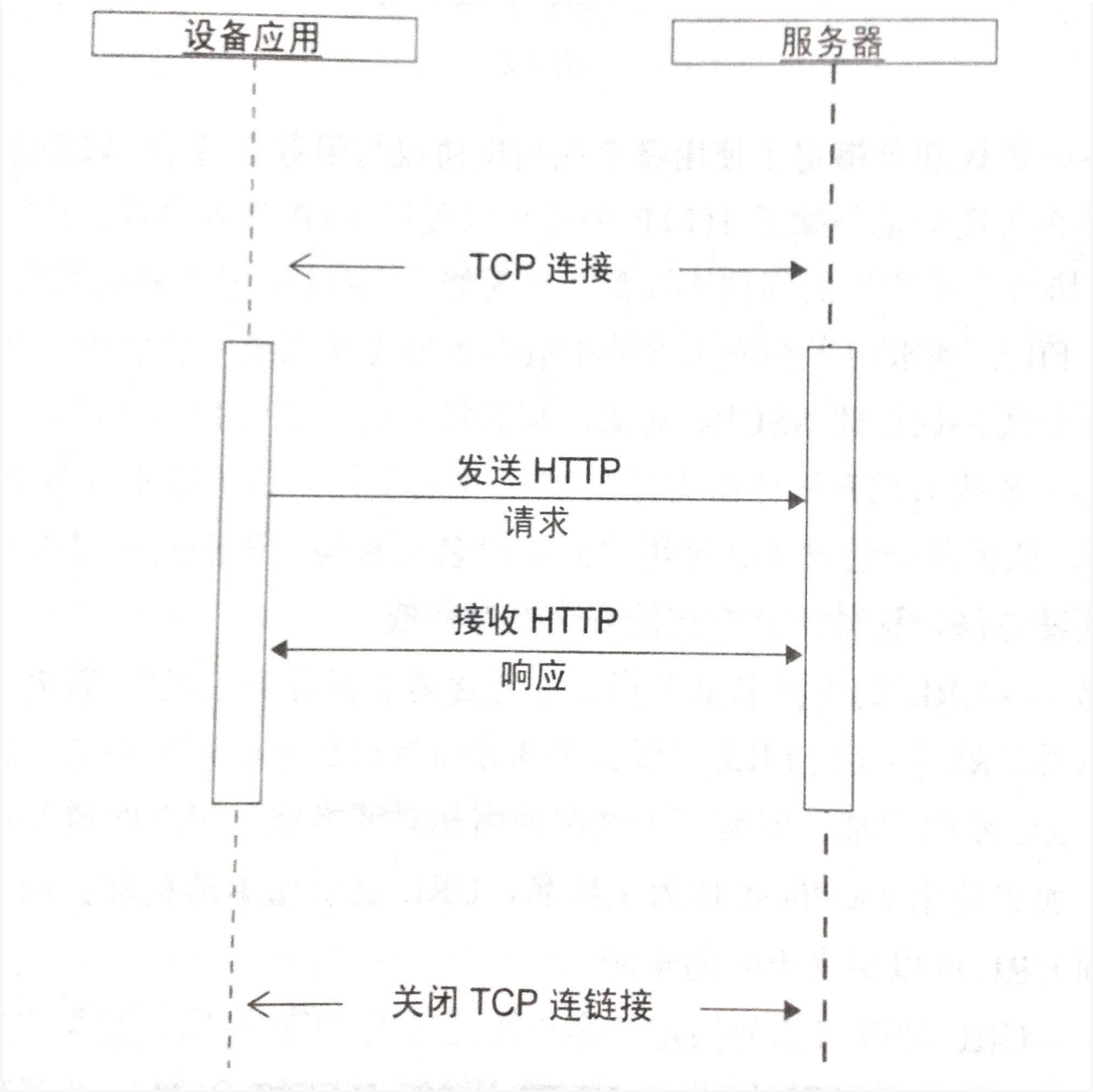
HTTP协议介绍

HTTP(Hyper Text Transfer Protocol)超文本传输协议

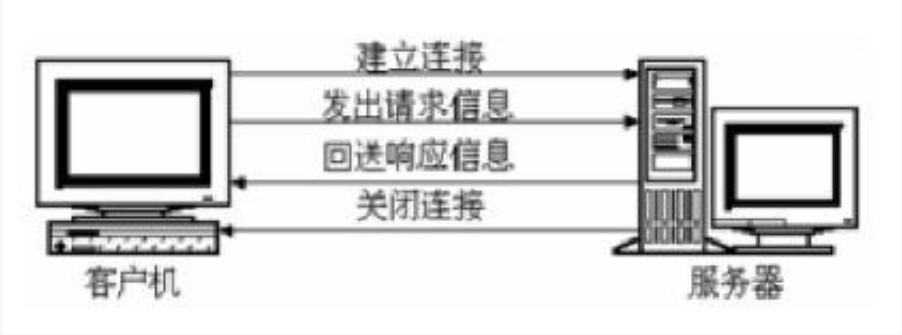
Tim Berners-Lee于1990年创造了HTTP协议的首个版本，最初的的提案中有3个主要的创新：HTML、HTTP与URL。

- HTML定义了向文本添加样式的一种方式
- HTTP定义了在服务端与客户端之间传输数据的一种方式
- URL定义了在网络机器中定位唯一资源的一种方式

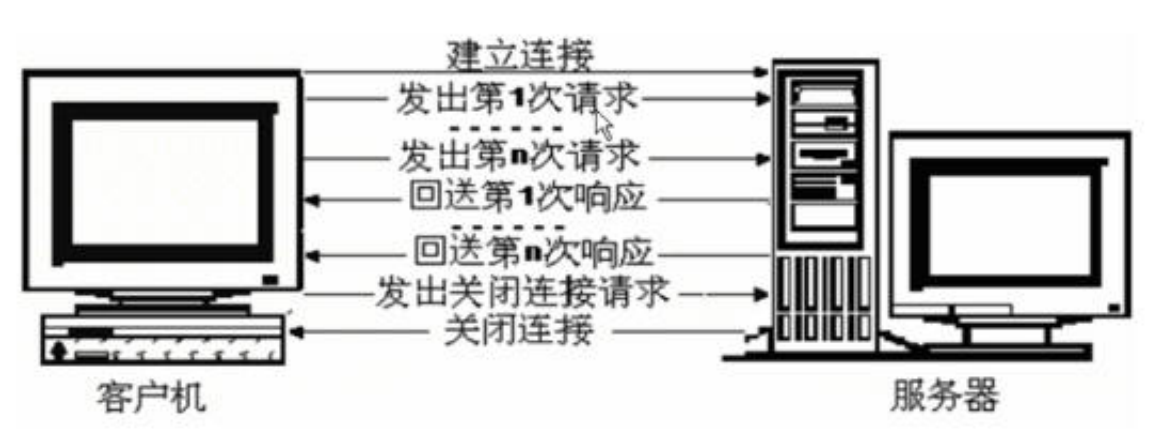
HTTP请求与响应



http1.0



HTTP1.1

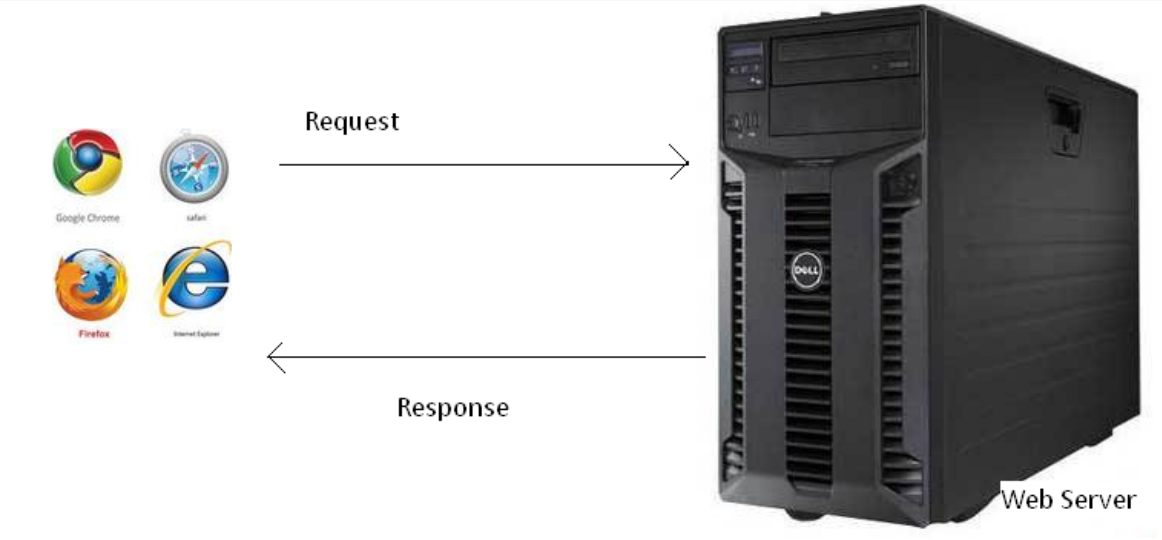


HTTP的规范是IETF FRF 2616 http://www.ietf.org/rfc/rfc2616.txt

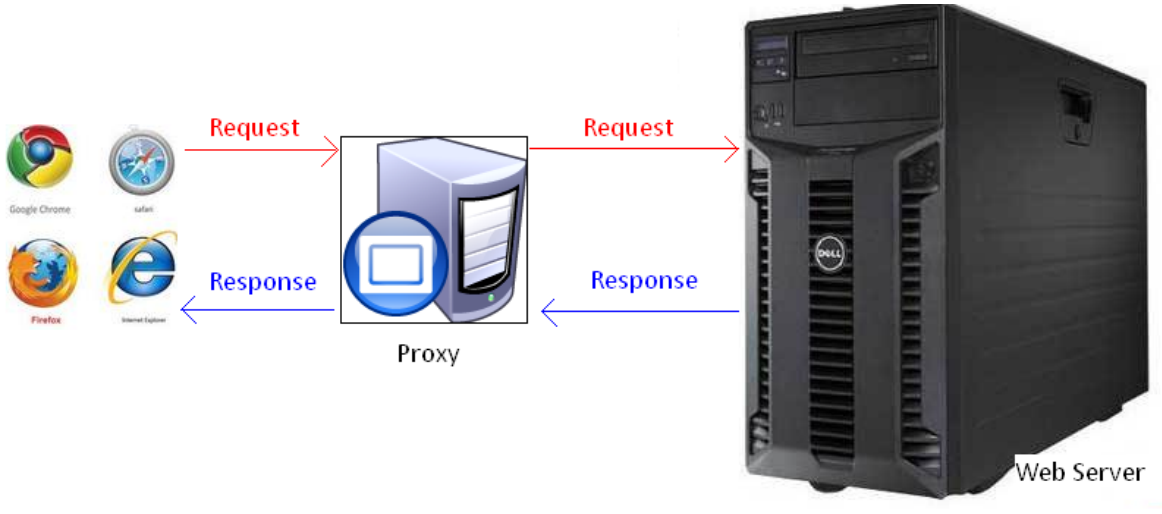
Web服务器，浏览器，代理服务器

当我们打开浏览器，在地址栏中输入URL，然后我们就看到了网页。 原理是怎样的呢?

1. 实际上我们输入URL后，我们的浏览器给Web服务器发送了一个Request
2. Web服务器接到Request后进行处理，生成相应的Response，然后发送给浏览器， 浏览器解析Response中的HTML,这样我们就看到了网页，过程如下图所示:



我们的Request有可能是经过了代理服务器，最后才到达Web服务器的



代理服务器就是网络信息的中转站，有什么功能呢?

1. 提高访问速度， 大多数的代理服务器都有缓存功能
2. 突破限制，也就是FQ了
3. 隐藏身份

URL结构

schema://host[:port]/path/.../[?query-string][#anchor] 协议://主机名[:端口]/绝对路径/[?查询字符串1&查询字符串2][锚]

eg:

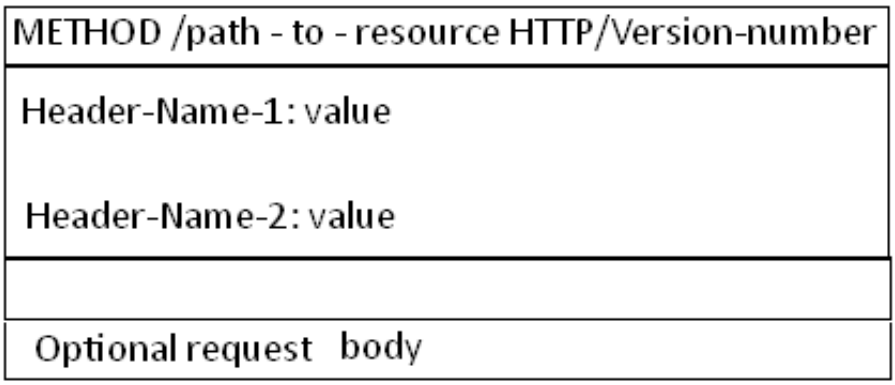
```
http://www.mytestsite.com/dorayo/test/test.aspx?name=dorayo&x=true#stuff

Schema:                 http
host:                   www.mytestsite.com
path:                   /dorayo/test/test.aspx
Query String:           name=dorayo&x=true
Anchor:                 stuff
```

绝对路径和查询字符串不可以包含空格、回车与换行符，因此，URL通常会使用百分号进行编码，RFC 3986(http://tools.ietf.org/html/rfc3986)规定了URL百分号编码的详细信息

请求内容

HTTP请求包含3部分：请求行、请求头与请求体



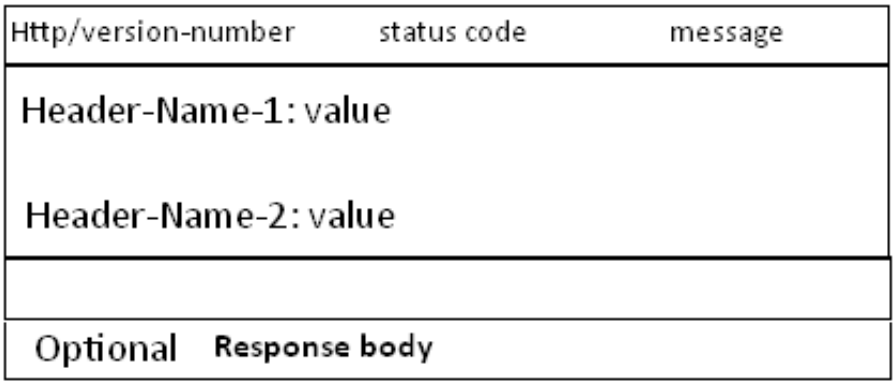
- Method表示请求方法,比如"POST","GET"
- Path-to-resoure表示请求的资源
- Http/version-number 表示HTTP协议的版本号
- 当使用的是"GET" 方法的时候， body是为空的

HTTP是无状态的协议，即HTTP服务器不会保留关于某个请求的任何信息以用在未来的请求中。Cookie机制提供了一种方式，可以将一些简单地状态信息存储在客户端，并在后续的请求中与服务器进行通信

请求体必须遵循客户端与服务器之间预先确定的数据编码，对于Web浏览器来说，这通常是表单编码数据，但对于移动应用来说，该编码通常是XML或JSON数据

响应内容

HTTP响应包含3部分：状态行、响应头与相应体



- HTTP/version-number表示HTTP协议的版本号
- status code 和 message 分别是状态码和状态消息

状态码用来告诉HTTP客户端,HTTP服务器是否产生了预期的Response

HTTP/1.1中定义了5类状态码， 状态码由三位数字组成，第一个数字定义了响应的类别

1XX 提示信息 - 表示请求已被成功接收，继续处理

2XX 成功 - 表示请求已被成功接收，理解，接受

3XX 重定向 - 要完成请求必须进行更进一步的处理

4XX 客户端错误 - 请求有语法错误或请求无法实现

5XX 服务器端错误 - 服务器未能实现合法的请求

常见的状态码:

200 OK

这表明该请求被成功地完成，所请求的资源发送回客户端

302 Found

重定向，新的URL会在response 中的Location中返回，浏览器将会自动使用新的URL发出新的Request

400 Bad Request

客户端请求与语法错误，不能被服务器所理解

403 Forbidden

服务器收到请求，但是拒绝提供服务

404 Not Found

请求资源不存在（输错了URL）

503 Server Unavailable

服务器当前不能处理客户端的请求，一段时间后可能恢复正常

在iOS的URL加载系统中，NSURLResponse及其子类NSHTTPURLResponse封装了请求返回的数据

HTTP协议是无状态的和Connection: keep-alive的区别

无状态是指协议对于事务处理没有记忆能力，服务器不知道客户端是什么状态。从另一方面

讲，打开一个服务器上的网页和你之前打开这个服务器上的网页之间没有任何联系

HTTP是一个无状态的面向连接的协议，无状态不代表HTTP不能保持TCP连接，更不能代表HTTP使用的是UDP协议（无连接）

从HTTP/1.1起，默认都开启了Keep-Alive，保持连接特性，简单地说，当一个网页打开完成后，客户端和服务器之间用于传输HTTP数据的TCP连接不会关闭，如果客户端再次访问这个服务器上的网页，会继续使用这一条已经建立的连接 Keep-Alive不会永久保持连接，它有一个保持时间，可以在不同的服务器软件（如Apache）中设定这个时间

抓包展示三次握手

Cocoa层 iOS HTTP API

在URL加载系统中，有3个主要的方式可以执行HTTP请求和接收响应

```
- 同步       （线程会被阻塞）
- 队列式异步          （创建一个队列并放入后台线程执行）
- 异步       （调用委托方法）
```

所有的URL加载请求方式都会用到这4类对象

```
NSURL、NSURLRequest、NSURLConnection与NSURLResponse对象
```