[Bluetooth GATT介绍](http://blog.csdn.net/hzl6255/article/details/41930457)

**1. 介绍**

GATT(Generic Attribute Profile)，描述了一种使用ATT的服务框架   
该框架定义了服务(Server)和服务属性(characteristic)的过程(Procedure)及格式   
Procedure定义了characteristic的发现、读、写、通知(Notifing)、指示(Indicating)   
及配置characteristic的广播

GATT可以被Application或其他Profile使用   
其协议栈如下图   
---------------        ---------------   
| Application | <----> | Application |   
---------------        ---------------   
|  Attribute  |        | Attribute   |   
|  Protocol   | <----> | Protocol    |   
---------------        ---------------   
|    L2CAP    | <----> |    L2CAP    |   
---------------        ---------------   
|  Controller | <----> |  Controller |   
---------------        ---------------

GATT可以配置为如下两种角色(Role)   
- Client : 命令、请求发起方   
- Server : 命令、请求接收方

角色配置实例如下

\_\_\_\_\_\_\_\_\_

/ ======= \

/ \_\_\_\_\_\_\_\_\_\_\

| \_\_\_\_\_\_\_\_\_\_\_ | Request /\

| |Computer | | ----------> / /

| |(Client) | | <---------- / / Sensor(Server)

| |\_\_\_\_\_\_\_\_\_| | Response / /

\=\_\_\_\_\_\_\_\_\_\_\_\_/ \/

/ """"""""""" \

/ ::::::::::::: \

(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

Computer是一个温度服务客户端, Sensor是温度服务服务器   
Computer向Sensor发起Procedure来读Sensor的值

GATT对下层的需求如下   
- Physical Link : 使用GAP Channel Establishment建立的ATT Bearer   
- GATT Role     : 不依赖于Coontroller角色(Master/Slave)   
- Security      : 对于LE,Security Features(Authorization、Authentication、Encryption)是可选的   
                  对于BR/EDR, Encryption是强制的   
- TX order      : GATT中的多字节字段，采用Least Significant octet先发送(Little Endian)

**2 内容**

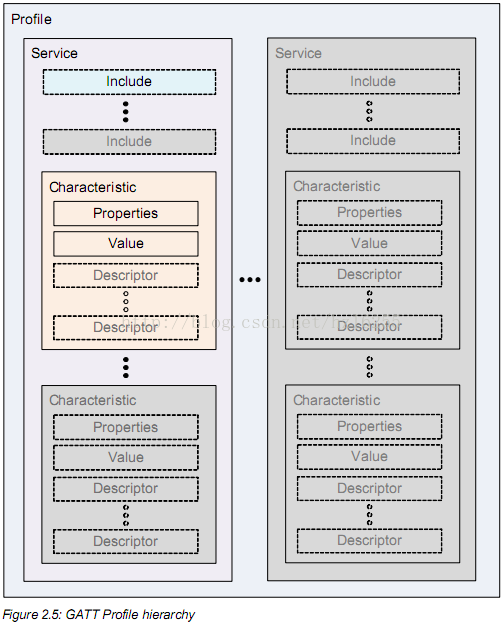
**2.1 Configured Broadcast**

对于LE物理链路，在Server广播模式过程中   
Client通过Configured Broadcast告知Server应该在advertising data加入Characteristic Value

方法是Client设置指定bit位   
广播频率则是Service、Characteristic行为定义的一部分

**2.2 GATT Profile Hierarchy**

GATT指定了数据交互的结构(Structure)   
这个结构体定义了一些基本元素，如Service、Characteristic   
这些元素存在于Attribute中



GATT中最上层是Profile，Profile由一个或多个服务(Service)组成   
服务是由Characteristics组成，或是其他服务的引用(Reference)   
Characteristic包含一个值(Value)，可能包含该Value的相关信息

**2.2.1 Service**

Service是[数据]和与之关联的[完成某个特定功能的行为]/[特性]的集合   
在GATT中，一个服务由服务定义(Service Defintion)来实现   
一个服务定义可能包含引用服务(Reference Service)、强制Characteristic和可选Characteristic

为了实现旧版本的兼容，新版本中服务定义只能增加新的引用服务或可选Characteristic   
新版本中的服务定义禁止从旧的服务定义中改变行为

Server有两类   
- Primary Service   : 拥有基本功能的服务,可被其他服务包含,可以通过Primary Service Discovery过程来发现   
- Secondary Service : 仅用来被Primary/Other Secondary Service、高层协议引用的服务

判读一个服务是Primary or Secondary Service可通过高层协议强制规定

**2.2.2 Included Service**

一个Included Service是一种引用已存在服务的方法，   
具体办法为在服务定义的开始加上Included Service的引用，   
这样整个Included Service定义成为新服务定义的一部分

**2.2.3 Characteristic**

Characteristic由Characteristic Definition定义，   
包含一个Characteristic声明、Characteristic属性、值、值的描述(Optional)

**3 Service Interoperability Requirements**

**3.1 Service Definition**

服务定义(Service Definition)包含一个服务申明(Service Declaration)   
    可能包含Include Definitions和Characteristic Definitions   
在下一个服务申明前或到达*Maximum Attribute Handle*时结束   
服务定义在服务端上基于*Attribute Handle*顺序呈现

服务定义中的Include Definitions和Characteristic Definitions被认为是服务的一部分   
服务定义中的顺序为   
Service Declaration ~ Include Definitions(>=0) ~ Characteristic Definitions(>=0) 

Service Declaration如下

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Handle | Attribute Type | Attribute Value | Attribute Permission |
| 0xNNNN | 0x2800 – UUID for <Primary Service>  0x2801 – UUID for <Secondary Service> | 16-bit Bluetooth UUID  128-bit UUID for Service | Read Only,  No Authentication,  No Authorization |

规则如下:   
- 当多个服务存在时   
      使用16-bit Bluetooth UUID服务定义的服务应该分组(如按顺序排列)   
      同理，使用128-bit UUID的服务定义的服务也分组   
- 一个设备或高层协议可能有多个服务定义，同时多个服务定义含有相同的Service UUID   
- 服务端的所有Attributes应该包含一个服务声明或存在一个服务定义   
- 服务端的服务定义可能无序;Client不应该认为服务端的服务定义一定是有序的

**3.2 Include Definition**

一个Include Definition只包含一个Include Declaration

Include Declaration如下

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute Handle | Attribute Type | Attribute Value | | | Attribute Permission |
| 0xNNNN | 0x2802 – UUID  for<Include> | Included Service  Attribute Handle | End Group  Handle | Service UUID | Read Only,  No Authentication,  No Authorization |

其中，仅当UUID是16-bit Bluetooth UUID时才存在

如果一个Service的Include Definition(A)是引用其他Server的Include Definition(B)   
那么Include Definition(B)不应该引用Include Definition(A)，否则就是循环引用(Circular Reference)

当一个Client检测到循环引用或detects nested include declarations to a greater level than it expects   
Client应当终止本次通信(ATT Bearer)

**3.3 Characteristic Definition**

Characteristic Definition包含如下内容   
- Characteristic Declaration                       : First   
- Characteristic Value declaration                 : Second   
- Characteristic Descriptor Declarations(Optional) : Last(含多个时顺序不关紧要)

Characteristic Definitions在服务端以*Attribute Handle*排序

以上每个Declaration包含在一个单独的Attribute中

**3.3.1 Characteristic Declaration**

Characteristic Declaration如下

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute Handle | Attribute Type | Attribute Value | | | Attribute Permission |
| 0xNNNN | 0x2803  – UUID for Characteristic | Characteristic Properties | Characteristic Value Attribute Handle | Characteristic UUID | Read Only,  No Authentication,  No Authorization |

其中，Attribute Value只读   
Attribute Value字段如下

|  |  |  |
| --- | --- | --- |
| Attribute Value | Size | Description |
| Characteristic Properties | 1 octets | Bit field of characteristic properties |
| Characteristic Value Handle | 2 octets | Handle of the Attribute containing the value of this characteristic |
| Characteristic UUID | 2/16 octets | 16-bit Bluetooth UUID or 128-bit UUID for Characteristic Value |

一个Service可能含有相同Characteristic UUID的Characteristic Definitions

在一个服务定义中，一些强制的Characteristics应该位于Include Declarations之后、其他可选characteristics之前   
而Client不应该认为Characteristic是有序的   
同时，16-bit Bluetooth UUIDs和128-bit Bluetooth UUIDs应该分别分组

Characteristic Properties以bit位的方式决定如下内容   
- Characteristic Value的使用方式   
- Characteristic Descriptors的访问方式

Characteristic Properties bit field的详细内容如下   
Notice: 多个比特位可以同时设置

|  |  |  |
| --- | --- | --- |
| Properties | Value | Description |
| Broadcast | 0x01 | If set, permits broadcasts of the Characteristic Value  using Characteristic Configuration Descriptor |
| Read | 0x02 | If set, permits reads of the Characteristic  Value using procedures defined in Section 4.8 |
| Write Without  Response | 0x04 | If set, permit writes of the Characteristic Value  without response using procedures  defined in Section 4.9.1 |
| Write | 0x08 | If set, permits writes of the Characteristic Value  with response using procedures  defined in Section 4.9.3 or Section 4.9.4 |
| Notify | 0x10 | If set, permits notifications of a Characteristic Value  without acknowledgement using the procedure  defined in Section 4.10 |
| Indicate | 0x20 | If set, permits indications of a Characteristic Value  with acknowledgement using the procedure  defined in Section 4.11 |
| Authenticated  Signed Writes | 0x40 | If set, permits signed writes to the Characteristic Value  using the procedure defined in Section 4.9.2 |
| Extended  Properties | 0x80 | If set, additional characteristic properties are  defined in the Characteristic Extended Properties Descriptor  defined in Section 3.3.3.1 |

**3.3.2 Characteristic Value Declaration**

Characteristic Value Declaration如下

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute Handle | Attribute Type | Attribute Value | Attribute Permissions |
| 0xNNNN | 0xuuuu – 16-bit Bluetooth UUID  or  128-bit UUID for Characteristic UUID | Characteristic Value | Higher layer profile  or  implementation specific |

**3.3.3 Characteristic Descriptor Declarations**

Characteristic Descriptor Declarations包含了*Characteristic Value*相关信息

GATT定义了一系列的标准Characteristic Descriptors供高层协议使用   
高层协议也可以定义协议相关的Characteristic Descriptors

Characteristic Descriptors在服务端上是无序的，Client不应该理所当然   
Characteristic Descriptors Declarations Permissions由高层协议定义或协议相关的   
Client不应该理所当然地认为是可读的

Characteristic Descriptor Declarations包括如下内容(*详细见规范*)

- Characteristic Extended Properties   
- Characteristic User Description   
- Client Characteristic Configuration   
- Server Characteristic Configuration   
- Characteristic Presentation Format   
- Characteristic Aggregate Format

**3.4 Summary of GATT Profile Attribute Types**

|  |  |  |
| --- | --- | --- |
| Attribute Type | UUID | Description |
| <Primary Service> | 0x2800 | Primary Service Declaration |
| <Secondary Service> | 0x2801 | Secondary Service Declaration |
| <Include> | 0x2802 | Include Declaration |
| <Characteristic> | 0x2803 | Characteristic Declaration |
| <Characteristic Extended Properties> | 0x2900 | Characteristic Extended Properties |
| <Characteristic User Description> | 0x2901 | Characteristic User Description Descriptor |
| <Client Characteristic Configuration> | 0x2902 | Client Characteristic Configuration Descriptor |
| <Server Characteristic Configuration> | 0x2903 | Server Characteristic Configuration Descriptor |
| <Characteristic Format> | 0x2904 | Characteristic Format Descriptor |
| <Characteristic Aggregate Format> | 0x2905 | Characteristic Aggregate Format Descriptor |

**4. GATT Feature Requirements**

**4.1 Overview**

GATT中定义了11项Feature   
1.  Server Configuration   
2.  Primary Service Discovery   
3.  Relationship Discovery   
4.  Characteristic Discovery   
5.  Characteristic Descriptor Discovery   
6.  Reading a Characteristic Value   
7.  Writing a Characteristic Value   
8.  Notification of a Characteristic Value   
9.  Indication of a Characteristic Value   
10. Reading a Characteristic Descriptor   
11. Writing a Characteristic Descriptor

每个Feature都有对应的过程和子过程，这些过程描述了如何使用ATT来实现各自的功能

**4.2 Feature Support and Procedure Mapping**

详细的对应列表如下表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Feature | Sub-Procedure | Ref | Client  Support | Server  Support |
| 1 | Server Configuration | Exchange MTU | 4.3.1 | O | O |
| 2 | Primary Service  Discovery | Discover All Primary Services | 4.4.1 | O | M |
| Discover Primary Services By  Service UUID | 4.4.2 | O | M |
| 3 | Relationship Discovery | Find Included Services | 4.5.1 | O | M |
| 4 | Characteristic  Discovery | Discover All Characteristic of a Service | 4.6.1 | O | M |
| Discover Characteristic by UUID | 4.6.2 | O | M |
| 5 | Characteristic  Descriptor Discovery | Discover All Characteristic Descriptors | 4.7.1 | O | M |
| 6 | Characteristic  Value Read | Read Characteristic Value | 4.8.1 | O | M |
| Read Using Characteristic UUID | 4.8.1 | O | M |
| Read Long Characteristic Values | 4.8.2 | O | O |
| Read Multiple Characteristic Values | 4.8.3 | O | O |
| 7 | Characteristic  Value Write | Write Without Response | 4.9.1 | O | C.1 |
| Signed Write Without Response | 4.9.2 | O | O |
| Write Characteristic Value | 4.9.3 | O | C.2 |
| Write Long Characteristic Values | 4.9.4 | O | O |
| Characteristic Value Reliable Writes | 4.9.5 | O | O |
| 8 | Characteristic  Value Notification | Notifications | 4.10.1 | O | O |
| 9 | Characteristic  Value Indication | Indications | 4.11.1 | M | C.3 |
| 10 | Characteristic  Descriptor Value Read | Read Characteristic Descriptors | 4.12.1 | O | O |
| Read Long Characteristic Descriptors | 4.12.2 | O | O |
| 11 | Characteristic  Descriptor Value Write | Write Characteristic Descriptors | 4.12.3 | O | O |
| Write Long Characteristic Descriptors | 4.12.4 | O | O |
| C1: Write Without Response is mandatory if Signed Write Without Response is supported  otherwise optional  C2: Write Characteristic Value is mandatory if Write Long Characteristic Values is supported  otherwise optional  C3: If Service Change Characteristic is present, this feature is mandatory, otherwise optional. | | | | | |

**4.3 Server Configuration**

该过程可被Client用来配置Attribute Protocol的MTU大小

**4.3.1 Exchange MTU**

Client使用该子过程来设置适配双方均支持的最大ATT\_MTU

在BR/EDR物理链路中不应该使用该过程，而应该使用L2CAP Channel Configuration Procedures

该过程对应于ATT的*MTU Exchange Request/Response*见<[Bluetooth ATT介绍](http://www.cnblogs.com/hzl6255/p/4141505.html) - 4.2 MTU Exchange>

**4.4 Primary Service Discovery**

Client使用该过程来发现服务端的Primary Services  
一旦发现服务存在，可通过**其他过程**来访问Primary Services的**附加信息**(关联主服务和次服务)  
可使用的**其他过程**包括Characteristic Discovery和Relationship Discovery

该过程包括两个子过程:  
- Discover All Primary Services  
- Discover Primary Services by Service UUID

在BR/EDR物理链路上则使用SDP service discovery来发现服务

**4.4.1 Discover All Primary Services**

Client使用该子过程来发现服务端的所有Primary Services

该子过程使用ATT的*Read By Group Type Request*,同时设置如下参数  
- Starting Handle : 0x0001  
- Ending Handle   : 0xFFFF  
- Attribute Type  : UUID for <Primary Service>

可能的回应有  
- Read By Group Type Response  
- Error Response

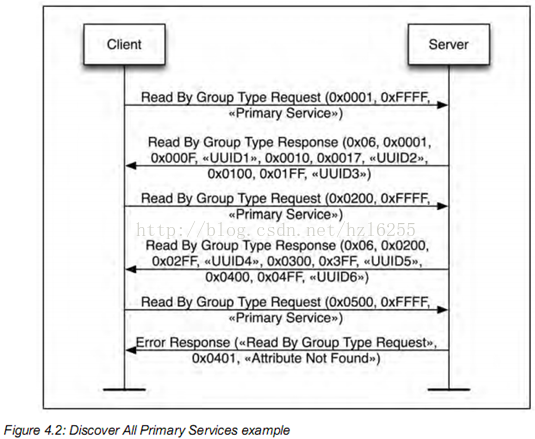
Read By Group Type Response返回三元组列表  
三元组包括  
- Attribute Handle : 服务声明的Handle  
- End Group Handle : 服务定义中最后一个Attribute的Handle  
- Attribute Value  : 服务端支持的服务的Service UUID

当收到Error Response<Error Code: Attribute Not Found>时  
则表明该过程已经完成

当Client找到自己所需要的服务时,可以终止该过程

*Note: 3.1中已指出Service Declaration是可读,并且不需要认证或授权  
         因此权限相关的错误不会发生*

下图是一个实例图



**4.4.2 Discover Primary Service by Service UUID**

当Client只知道Service UUID时,可以使用该子过程来发现对应的主服务

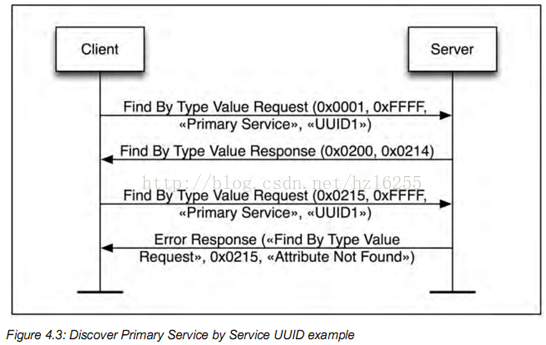
该子过程使用ATT的*Find By Type Value Request*,同时设置参数如下  
- Starting Handle : 0x0001  
- Ending Handle   : 0xFFFF  
- Attribute Value : 16-bit Bluetooth UUID or 128-bit UUID  
- Attribute Type  : UUID for <Primary Service>

可能的回应有  
- Find By Type Value Response  
- Error Response

Find By Type Value Response返回*Attribute Handle* ranges列表  
Attribute Handle range即服务定义的*Starting Handle*和*Ending Handle*如果Attribute Handle range中的End Found Handle不是0xFFFF  
那么Client将会再请求一次Req  
同时将*Starting Handle*设置为收到的最后一个Attribute Handle**+1**

终止规则和权限问题同Discover All Primary Services

下图是一个实例图



**4.5 Relationship Discovery**

Client使用该过程来发现和其他服务的服务关系

**4.5.1 Find Include Services**

Client使用该子过程来发现一个服务定义包含的服务申明

该子过程使用ATT的Read By Type Request,同时设置参数如下  
- Starting Handle : 所要查找服务的Starting Handle  
- Ending Handle   : 所要查找服务的Ending Handle  
- Attribute Type  : UUID for <Include>

可能的回应有  
- Find By Type Response  
- Error Response

Find By Type Response返回[*Attribute Handle, Attribute Value*]集合对  
Attribute Value由所包含服务申明的*Attribute Handle*和*End Group Handle*组成  
当UUID为16-bit Bluetooth UUID时,那么它也将包含在Rsp中

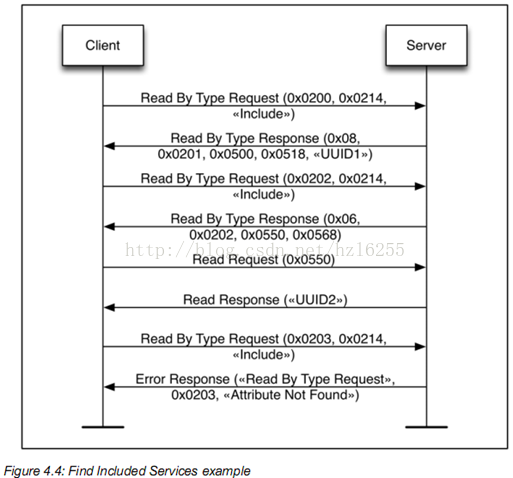
该Req应该被再次请求，同时设置*Starting Handle*为为收到的最后一个Attribute Handle**+1**

当Rsp中包含的服务申明中Attribute Handle等于Req的Ending Handle时,  
该子过程被认为完成(当然Attribute Not Found-Error Rsp也是)

当Include Service使用128-bit UUID时  
使用Read Request来获取Include Service UUID  
其中Attribute Handle参数设置为Include Service的*Attribute Handle*

权限规则同上面

下图是一个实例图



========================================================