IEEE 1888 SDK开发编程指南

（for C）

V1.0

北京天地互连信息技术有限公司
http://www.biigroup.com
技术热线：01058678188-161

使用说明手册更新记录

时间	版本号	更新内容	责任人
2014 年 8 月 9 日	V0.1	新完成	张兵涛
2015 年 7 月 7 日	V1.0	增加 SDK 介绍、将 client 和 serve 端分开介绍、修改文档格式	常琳

声　明

非常感谢您使用IEEE 1888 SDK进行项目开发，如果您有什么疑问或需要请随时联系我们。

- 我们已尽量保证手册内容的完整性与准确性，但也不免出现技术上不准确、与产品功能及操作不相符或印刷错误等情况出现，如有任何疑问或争议，请以天地互连最终解释为准。
- 产品和手册将实时进行更新，恕不另行通知。
- 本手册中内容仅为用户提供参考指导作用，请以SDK 实际内容为准

目录

1. SDK 简介

IEEE 1888 智慧能源标准开发 SDK 主要是基于通用的 SOAP 协议开发，为 IEEE 1888 的网关、存储 Storage、应用 APP 等组件提供开放调用服务，可用于智慧能源、智能建筑、智能电网等行业应用中组件的二次开发。

IEEE 1888 SDK 包括客户端和服务器端两部分，其中客户端主要功能为：

- 发送客户端的 FETCH 请求给服务器端，获取点数据并解析请求结果返回给 SDK 调用方；
- 发送客户端的 WRITE 请求给服务器端，解析请求结果返回给 SDK 调用方；

服务器端主要功能为：

- 解析客服端发来的请求，若为 FETCH 请求，将客户端要求的数据以 IEEE 1888 的格式发送给客户端；若为 WRITE 请求，将客户端发来的 IEEE 1888 格式的数据处理，如数据存储、反向控制等，返回客户端状态消息；

2. 客户端函数介绍

2.1. 功能接口简介

2.1.1. FETCH 客户端接口

功能	接口	相关参数
从远程组件获取数据	IEEE1888_ClientBasicFetch	serverUrl：服务端地址
		pBuffer：查询条件
		pCount：查询点的个数
获取数据操作结束后，回收内存	IEEE1888_DestoryObject	p_value：需要回收内存的地址
		n_value：需要回收的数组数量

2.1.2. WRITE 客户端接口

功能	接口	相关参数
将点的数据写入指定服务端	IEEE1888_ClientBasicWrite	serverUrl：服务端地址
		pBuffer：点要写入的数据内容
		pCount：写入点的数量

2.2. 函数说明

2.2.1. IEEE1888_ClientBasicFetch

函数功能	从服务端查询指定点的指定数据
函数	int IEEE1888_ClientBasicFetch(char* serverUrl,fetch_point * pBuffer,int nCount)
参数	char* serverUrl：需要查询的远端服务端地址； fetch_point * pBuffer：所查询点的信息以及查询条件； int nCount：查询的点的个数；
返回值	0：查询成功，查询的数据通过指针存储在 ieee1888_value_ext p_value 中； 1：查询失败；

说明	fetch_point 结构定义如下：
	typedef struct fetch_point {
	char point_id[MAX_POINT_ID_LEN];//要查询对象的 ID
	int select；// 0:查询最大值 1:查询最小值 2:指定查询范围
	char pGt[MAX_TIME_LEN]; //大于某值
	char pLt[MAX_TIME_LEN]; //小于某值
	int n_value; //此 ID 查询结果的数量
	ieee1888_value_ext *p_value; //此 ID 的查询结果
	}fetch_point;
	typedef struct

	ieee1888_value_ext { char*
	time; //数据
	写入时间
	char* content; //数据
	值
	}ieee1888_value_ext;

2.2.2. IEEE1888_ClientBasicWrite

函数功能	向服务端写入指定点的数据
函数	int IEEE1888_ClientBasicWrite(char *serverUrl, write_point *pBuffer,int pCount)
参数	char* serverUrl：写入服务端的地址； write_point * pBuffer：点需要写入的数据信息； int pCount：本次写入点的个数；
返回值	0：写入成功； 1：写入失败；

说明	Write_point 结构定义如下：
	typedef struct write_point{
	char point_id[MAX_POINT_ID_LEN]; //上报的管控点 ID
	int n_value; //点上报的数据条数
	ieee1888_write_ext * p_value; //点上报的数据内容
	}write_point;
	Typedef struct ieee1888_write_ext{
	Time_t time; //数据写入时间
	Char * content; //数据值
	}ieee1888_write_ext;

2.2.3. IEEE1888_DestoryObject

函数功能	内存回收
函数	int IEEE1888_DestoryObject(ieee1888_value_ext* p_value, int n_value);

参数	ieee1888_value_ext* p_value：需要回收内存的地址 int n_value：需要回收的数组数量
返回值	0：回收成功 1：回收失败

说明	ieee1888_value_ext 结构定义如下：
	typedef struct ieee1888_value_ext{
	char* time; //数据写入时间
	char* content; //数据值
	}ieee1888_value_ext;

3. 服务器端函数介绍

3.1. 功能接口简介

功能	接口	相关函数
开启 server 端,开始监听端口	IEEE1888_ServerCreate	Port：指定监听的端口
当监听的端口有数据传入,此函数指定数据的处理方式	IEEE1888_SetServerHandler	Query：来源是 query 请求时的处理方式
		Data：来源是data 请求时 的处理方式
		Fault：客户端请求错误时 server 端的处理方式

3.2. 函数说明

3.2.1. IEEE1888_ServerCreate

函数功能	开启server 端，开始监听端口
函数	int IEEE1888_ServerCreate(int port)
参数	int port：指定监听的端口
返回值	返回监听结果，正常情况线程等待数据，不返回 错误返回值： 11：DNS 解析失败 （ERROR_DNS_RESOLVE）

	12：SOCKET 创建失败（ERROR_SOCKET_CREATE） 13：SOCKET 监听失败（ERROR_SOCKET_LISTEN） 14：SOCKET 选择失败（ERROR_SOCKET_SELECT） 15：SOCKET 接受失败（ERROR_SOCKET_ACCEPT）
说明	此函数一般最后调用，需要先指定收到数据的处理方式

3.2.2. IEEE1888_SetServerHandler

函数功能	当监听的端口有数据传入，此函数指定数据的处理方式
函数	void IEEE1888_SetServerHandler (query_handler query, data_handler data, fault_handler fault)
参数	query_handler query：指定query 接口的处理函数 data_handler data：指定data 接口的处理函数 fault_handler fault：客户端请求错误时server 端的处理方式
返回值	无

说明	query_handler 和data_handler 为函数指针,此接口接收处理逻辑， 在端口接收到数据后，调用此处传入的函数进行处理
	typedef points_value *(*query_handler)(fetch_points* request);
	typedef int(*data_handler)(points_value* request);
	typedef void(*fault_handler)(fault* fault);
	query_handler 指向的函数，传入参数为 fetch_points，传出参数 为points_value*,常用场景为接收到point 的相关信息，在数据库 中查询此 point 的最新数据，再赋值到 points_value。 data_handler 指向的函数，传入参数为 points_value，传出参数为 int。常用场景为收到 point 的数据之后，存储到数据库，并返回 操作结果。

4. 数据结构说明

4.1 数字整型

```
#define IEEE1888_DATATYPE_NONE          0
#define IEEE1888_DATATYPE_KEY           1
#define IEEE1888_DATATYPE_OK            2
#define IEEE1888_DATATYPE_ERROR         3
#define IEEE1888_DATATYPE_QUERY         4
#define IEEE1888_DATATYPE_HEADER        5
#define IEEE1888_DATATYPE_VALUE         6
#define IEEE1888_DATATYPE_POINT         7
#define IEEE1888_DATATYPE_POINTSET      8
#define IEEE1888_DATATYPE_BODY          9
#define IEEE1888_DATATYPE_TRANSPORT     10
```

4.2 字符串型

```
typedef char    ieee1888_uuid;
typedef char    ieee1888_queryType;
typedef char    ieee1888_attrNameType;
typedef char    ieee1888_selectType;
typedef char    ieee1888_trapType;
typedef char    ieee1888_time;
typedef char    ieee1888_uri;
typedef char    ieee1888_url;
```

4.3 数据结构

```
typedef struct  _ieee1888_key       ieee1888_key
typedef struct  _ieee1888_OK        ieee1888_OK
typedef struct  _ieee1888_error     ieee1888_error
typedef struct  _ieee1888_query     ieee1888_query
typedef struct  _ieee1888_header    ieee1888_header
```

typedef struct	_ieee1888_value	ieee1888_value
typedef struct	_ieee1888_point	ieee1888_point
typedef struct	_ieee1888_pointSet	ieee1888_pointSet
typedef struct	_ieee1888_body	ieee1888_body
typedef struct	_ieee1888_transport	ieee1888_transport
typedef struct	_ieee1888_object	ieee1888_object

以上数据结构按照soap 包嵌套关系顺序定义，详见ieee1888.h 文件