说明：该文档以修改ffe0服务下的ffe1通道属性为例。

1. 在ffe0s.c文件下的ffe0s_init函数里做如下图修改。

```
static uint8_t ffe0s_init (struct prf_task_env* env, uint16_t* start_hdl, uint16_t app
{
    uint16_t shdl;
    struct ffe0s_env_tag* ffe0s_env = NULL;
    // Status
    uint8_t status = GAP_ERR_NO_ERROR;
    //-------------------- allocate memory required for the profile  ----------------
    ffe0s_env = (struct ffe0s_env_tag* ) ke_malloc(sizeof(struct ffe0s_env_tag), KE_ME
    memset(ffe0s_env, 0 , sizeof(struct ffe0s_env_tag));
    // Service content flag
    uint16_t cfg_flag = 0x1ff;
    // Save database configuration
    ffe0s_env->features |= (params->features) ;
    shdl = *start_hdl;
    //Create ffe0 in the DB
    //------------------ create the attribute database for the profile ---------------
    status = attm_svc_create_db(&(shdl), ATT_USER_SERVER_ffe0, (uint8_t *)&cfg_flag,
            ffe0S_IDX_NB, NULL, env->task, &ffe0_att_db[0],
            (sec_lvl & (PERM_MASK_SVC_DIS | PERM_MASK_SVC_AUTH | PERM_MASK_SVC_EKS)));
    //Set optional permissions
    if (status == GAP_ERR_NO_ERROR)
    {
        //Set optional permissions
        if((params->features & 0x01) == ffe0_FEE1_LVL_NTF_SUP)
        {
            // Battery Level characteristic value permissions
            uint16_t perm = PERM(RD, ENABLE) | PERM(IND, ENABLE);
            attm_att_set_permission(shdl + ffe0S_IDX_FEE1_LVL_VAL, perm, 0);
        }
        if((params->features & 0x02) == ffe0_FEE3_LVL_NTF_SUP)
        {
```

2. 在ffe0s.c文件下的ffe0s_notify_fee1_lvl函数里做如下图修改。

```
void ffe0s_notify_fee1_lvl(struct ffe0s_env_tag* ffe0s_env, struct ffe0s_fee1_level_upd_req co
{
    // Allocate the GATT notification message
    struct gattc_send_evt_cmd *fee1_lvl = KE_MSG_ALLOC_DYN(GATTC_SEND_EVT_CMD,
            KE_BUILD_ID(TASK_GATTC, 0), prf_src_task_get(&(ffe0s_env->prf_env),0),
            gattc_send_evt_cmd, sizeof(uint8_t)* (param->length));
    // Fill in the parameter structure
    fee1_lvl->operation = GATTC_INDICATE;//GATTC_NOTIFY;
    fee1_lvl->handle = ffe0s_get_att_handle(ffe0S_IDX_FEE1_LVL_VAL);
    // save current to g_handle
    g_hande_ffe0 = fee1_lvl->handle;
    // pack measured value in database
    fee1_lvl->length = param->length;
    memcpy(&fee1_lvl->value[0],&param->fee1_level[0],param->length);
    // send notification to peer device
    ke_msg_send(fee1_lvl);
}
```

3. 在ffe0s_task.c文件下的gattc_write_req_ind_handler函数里做如下图修改。

```
static int gattc_write_req_ind_handler(ke_msg_id_t const msgid, struct gattc_write_req_ind const *param,
                                       ke_task_id_t const dest_id, ke_task_id_t const src_id)
{
    UART_PRINTF("ffe0s_task.c:%s   line:%d\r\n",__func__,__LINE__);
    struct gattc_write_cfm * cfm;
    uint8_t att_idx = 0;
    uint8_t conidx = KE_IDX_GET(src_id);
    // retrieve handle information
    uint8_t status = ffe0s_get_att_idx(param->handle,  &att_idx);
    // If the attribute has been found, status is GAP_ERR_NO_ERROR
    if (status == GAP_ERR_NO_ERROR)
    {
        struct ffe0s_env_tag* ffe0s_env = PRF_ENV_GET(ffe0S, ffe0s);
        // Extract value before check
        uint16_t ntf_cfg = co_read16p(&param->value[0]);
        // Only update configuration if value for stop or notification enable
        UART_PRINTF("att_idx=%d ntf_cfg=%d\r\n",att_idx,ntf_cfg);
        if ((att_idx == ffe0S_IDX_FEE1_LVL_NTF_CFG)
                && ((ntf_cfg == PRF_CLI_STOP_NTFIND) || (ntf_cfg == PRF_CLI_START_IND)))
        {
            UART_PRINTF("ffe0S_IDX_FEE1_LVL_NTF_CFG\r\n");
            // Conserve information in environment
            if (ntf_cfg == PRF_CLI_START_IND)
            {
                // Ntf cfg bit set to 1
                ffe0s_env->ffe1_ntf_cfg[conidx] = (ffe0_FEE1_LVL_NTF_SUP );
            }
```

4. 在ffe0s_task.c文件下的gattc_read_req_ind_handler函数里做如下图修改。

```
static int gattc_read_req_ind_handler(ke_msg_id_t const msgid, struct gattc_read_req_ind const *param,
                                      ke_task_id_t const dest_id, ke_task_id_t const src_id)
{
    UART_PRINTF("ffe0s_task.c:%s   line:%d\r\n",__func__,__LINE__);
    struct gattc_read_cfm * cfm;
    uint8_t  att_idx = 0;
    uint8_t conidx = KE_IDX_GET(src_id);
    // retrieve handle information
    uint8_t status = ffe0s_get_att_idx(param->handle, &att_idx);
    uint16_t length = 0;
    struct ffe0s_env_tag* ffe0s_env = PRF_ENV_GET(ffe0S, ffe0s);
    // If the attribute has been found, status is GAP_ERR_NO_ERROR
    if (status == GAP_ERR_NO_ERROR) ...
    //Send write response
    cfm = KE_MSG_ALLOC_DYN(GATTC_READ_CFM, src_id, dest_id, gattc_read_cfm, length);
    cfm->handle = param->handle;
    cfm->status = status;
    cfm->length = length;
    if (status == GAP_ERR_NO_ERROR)
    {
        // read notification information
        if (att_idx == ffe0S_IDX_FEE1_LVL_VAL)
        {
            cfm->value[0] = ffe0s_env->fee1_lvl[0];
        }
        #if 1
        // retrieve notification config
        else if (att_idx == ffe0S_IDX_FEE1_LVL_NTF_CFG)
        {
            uint16_t ntf_cfg = PRF_CLI_START_IND ;//(ffe0s_env->ffe1_ntf_cfg[conidx] & ffe0_FEE1_LVL_NTF_SU
            UART_PRINTF("ntf_cfg=%d  ffe0s_env->ntf_cfg[conidx]=%d\r\n",ntf_cfg,ffe0s_env->ffe1_ntf_cfg[con
            co_write16p(cfm->value, ntf_cfg);
        }
        #endif
        else if(att_idx == ffe0S_IDX_FEE3_LVL_VAL)
        {
            cfm->value[0] = ffe0s_env->fee3_lvl[0];
        }
        #if 1
        else if(att_idx == ffe0S_IDX_FEE3_LVL_NTF_CFG)
        {
            //uint16_t ntf_cfg = (ffe0s_env->ntf_cfg[conidx] & ffe0_FEE3_LVL_NTF_SUP) ? PRF_CLI_START_NTF :
            uint16_t ntf_cfg =  PRF_CLI_START_IND ;
```

5. 修改前ffe0服务下的ffe1通道为notify属性，修改后ffe0服务下的ffe1通道为indicate属性