GATT Interface Specification

Interface Specification

RW-BLE-GATT-IS

Version 8.05

2019-01-30

RivieraWaves
http://www.rivierawaves.com/

Revision History

Version	Date	Revision Description	Author
1.00	2012-12-18	GATT 1.0 API	KYAP/ LT/ FBELOUIN
2.00	2013-01-31	GATT 2.0 API	FBELOUIN/LT
7.00	2014-06-30	BLE 4.1	FBELOUIN
7.01	2014-08-05	Attribute value handle is just after characteristic handle	FBELOUIN
7.02	2014-12-01	Add Sequence number in GATTC commands return back in completed event	FBELOUIN
7.03	2015-01-06	Add a message to inform application about a ATT Transaction timeout.	FBELOUIN
8.00	2015-01-16	Updated to BLE 4.2 API	FBELOUIN
8.01	2015-07-29	Naming mismatches corrected	CMORAL
8.02	2015-10-22	GATT Service Permission updated, editorial	KYAP
8.03	2017-11-09	Usage of char descriptor discovery	KYAP
8.04	2018-08-30	GATTC_READ_REQ_IND and GATTC_MTU_CHANGED_IND parameters alignment 4.3.6.2 GATTC_SDP_SVC_IND updated	FBELOUIN/KYAP
8.05	2019-01-30	4.2.1 GATTC_CMP_EVT description updated	KYAP

Table of Contents

List of Tables

1 Overview

The RW-BLE Generic Attribute Profile (GATT) defines the service framework using the Attribute Protocol for discovering services, reading and writing characteristic values on a peer device (See [1]).

1.1 Document Overview

This document describes the non-standard interface of the RW-BLE Generic Attribute Profile implementation. Along this document, the interface messages will be referred to as API messages for the profile block(s).

Their descriptions will include their utility and reason for implementation for a better understanding of the user and the developer that may one day need to interface them from a higher application.

Moreover, it is recommended that the user check the html-based documentation of the RW-BLE Host, which is derived from actual RW-BLE host code and formatted via *Doxygen*. This material can further provide information on RW-BLE GATT implementation (e.g. data structures, states, message calling).

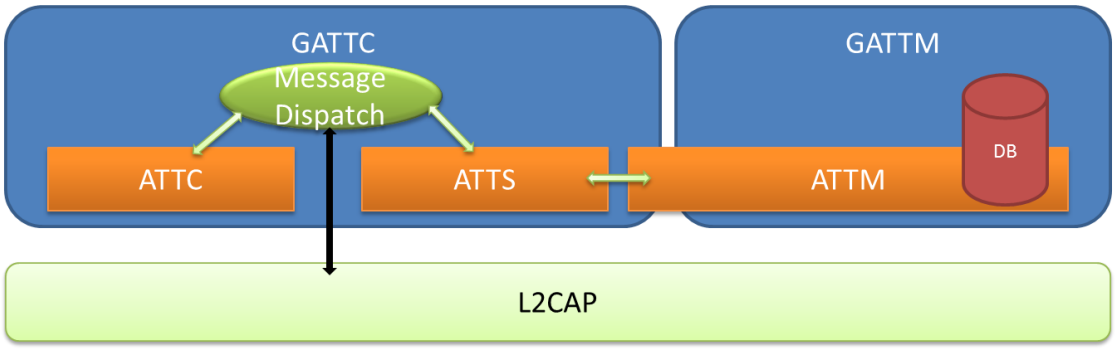
1.2 Protocol Overview

The RW-BLE GATT has complete and substantial support of the LE GATT (Core 4.2):

- ✓ Two Roles – client and server
- ✓ Configuration Exchange
- ✓ Attribute Discovery
- ✓ Reading/Writing Characteristic
- ✓ Indicating/Notifying Characteristic
- ✓ Profile Interface
- ✓ Service Discovery Procedure

1.3 Implementation Overview

The RW-BLE GATT is divided in two parts. First task is mono instantiated and manages all application requests not related to a link (mainly access and modification of local attribute database). This task is the GATT Manager (called GATTM) and it also manages creation and/or deletion of the second type of GATT task, the GATT Controller (called GATTC). This task is multi instantiated; one instance of GATTC is created when a connection to a peer device is created and deleted when connection is terminated. Index of the created task is related to connection index created for connection in General Access Profile (GAP see [3]).



GATT interface schema representing internal tasks.

2 Profile Roles

The RW-BLE GATT supports the two roles of GATT (See [2]).

GATT Client

This is the device that initiates commands and requests towards the GATT server. It can receive responses, indications and notifications sent by the GATT server.

GATT Server

This is a device that accepts incoming commands and requests from the GATT client. It can send responses, indications and notifications to the GATT client.

3 GATT Manager (GATTM)

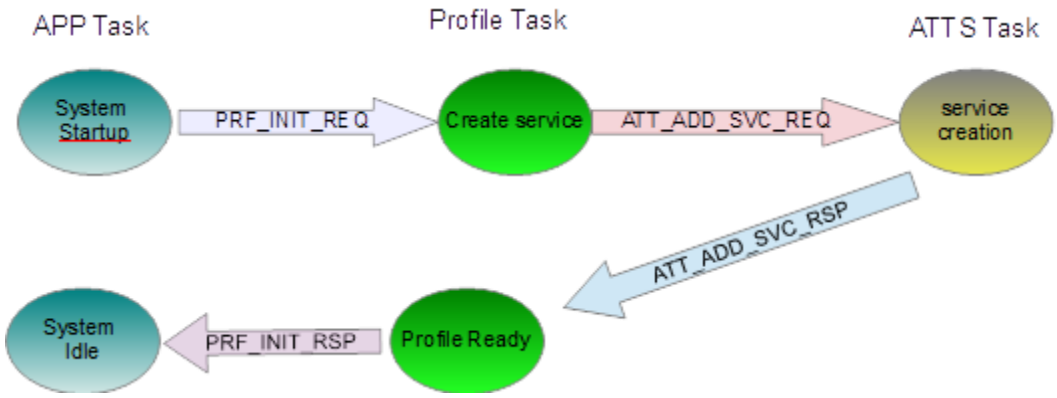
Mono-instantiated task, the GATT Manager provides a message API used to manage the internal attribute database.

The GATT Manager block has handlers for these messages, defined in gattm_task files (.h/.c).

3.1 Database Creation

At system start-up, application shall activate profiles in order to create attribute database. If a GAP Reset is requested, or if GAP configuration is updated (shall be done once at system start-up), database is cleared and default GAP and GATT services are inserted in the database. It means also that after a GAP Reset or GAP Device Configuration, profile entries in the database are removed and shall be re-created.

Database can be also created and managed by the application using the following kernel messages.



Database creation at start-up

The Attribute database is highly configurable according to profile requirements:

- Attribute value can be managed by database to simplify profile implementation.
- Attribute value can be managed by profile to optimize RAM memory usage.
- Legacy Attribute sizes are optimized into the database.
- Characteristic value handle is put just after characteristic definition handle.

❖ **attm_svc_perm_mask**

7	6	5	4	3	2	1	0
SEC	UUID_LEN		DIS	AUTH		EKS	MI

Value	Flag	Description
0x01	PERM_MASK_SVC_MI	Task that manage service is multi-instantiated
0x00	PERM_POS_SVC_MI	
0x02	PERM_MASK_SVC_EKS	Check Encryption key size for service Access
0x01	PERM_POS_SVC_EKS	(Encryption key Size must be 16 byte)
0x0C	PERM_MASK_SVC_AUTH	Service Permission authentication
0x02	PERM_POS_SVC_AUTH	(0 = Disable, 1 = Enable, 2 = UNAUTH, 3 = AUTH)
0x10	PERM_MASK_SVC_DIS	Service Disable
0x04	PERM_POS_SVC_DIS	
0x60	PERM_MASK_SVC_UUID_LEN	Service UUID Length
0x05	PERM_POS_SVC_UUID_LEN	(0 = 16 bits, 1 = 32 bits, 2 = 128 bits, 3 = RFU)
0x80	PERM_MASK_SVC_SECONDARY	Secondary Service present
0x07	PERM_POS_SVC_SECONDARY	

Table 1: Service Permission bit field

❖ **attm_perm_mask**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXT	WS	I	N	WR	WC	RD	B	NP		IP		WP		RP	

Value	Flag	Description
0x0003	PERM_MASK_RP	Read Access Mask:
0x00	PERM_POS_RP	(0 = NO_AUTH, 1 = UNAUTH, 2 = AUTH, 3 = SEC_CON)
0x000C	PERM_MASK_WP	Write Access Mask
0x02	PERM_POS_WP	(0 = NO_AUTH, 1 = UNAUTH, 2 = AUTH, 3 = SEC_CON)
0x0030	PERM_MASK_IP	Indication Access Mask
0x04	PERM_POS_IP	(0 = NO_AUTH, 1 = UNAUTH, 2 = AUTH, 3 = SEC_CON)
0x00C0	PERM_MASK_NP	Notification Access Mask
0x06	PERM_POS_NP	(0 = NO_AUTH, 1 = UNAUTH, 2 = AUTH, 3 = SEC_CON)
0x0100	PERM_MASK_BROADCAST	Broadcast descriptor present
0x08	PERM_POS_BROADCAST	
0x0200	PERM_MASK_RD	Read Access Mask
0x09	PERM_POS_RD	
0x0400	PERM_MASK_WRITE_COMMAND	Write Command Enabled attribute Mask
0x0A	PERM_POS_WRITE_COMMAND	
0x0800	PERM_MASK_WRITE_REQ	Write Request Enabled attribute Mask

0x0B	PERM_POS_WRITE_REQ	
0x1000	PERM_MASK_NTF	Notification Access Mask
0x0C	PERM_POS_NTF	
0x2000	PERM_MASK_IND	Indication Access Mask
0x0D	PERM_POS_IND	
0x4000	PERM_MASK_WRITE_SIGNED	Write Signed Enabled attribute Mask
0x0E	PERM_POS_WRITE_SIGNED	
0x8000	PERM_MASK_EXT	Extended properties descriptor present
0x0F	PERM_POS_EXT	

Table 2: Attribute Permission bit field

❖ **attm_ext_perm_mask**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RI	UUID_LEN		EKS	RFU											

Value	Flag	Description
0x1000	PERM_MASK_EKS	Check Encryption key size Mask
0x0C	PERM_POS_EKS	
0x6000	PERM_MASK_UUID_LEN	UUID Length
0x0D	PERM_POS_UUID_LEN	
0x8000	PERM_MASK_RI	Read trigger Indication
0x0F	PERM_POS_RI	

Table 3: Attribute Extended Permission bit field

❖ gattm_att_desc

Type	Parameters	Description
uint8_t [16]	uuid	Attribute UUID (LSB First)
uint16_t	perm	**Attribute Permission bit field: (see** Table 2**)**
uint16_t	max_len	**Attribute Value Length** - Maximum Attribute Length **Note**: - For Included Services and Characteristic Declarations, this field contains targeted handle. - For Characteristic Extended Properties, this field contains 2 byte value - Not used Client Characteristic Configuration and Server Characteristic Configuration, this field is not used.
uint16_t	ext_perm	**Attribute Extended Permission bit field: (see** Table 3**)**

Table 4: Attribute Description structure

3.1.1 GATTM_ADD_SVC_REQ

Parameters:

Type	Parameters	Description
uint16_t	start_hdl	Attribute Start Handle (0 = dynamically allocated)
uint16_t	task_id	Task identifier that manages the service
uint8_t	perm	Service Permission (see 1)
uint8_t	nb_att	Number of attribute(s) in service
uint8_t[16]	uuid	Service UUID
struct gattm_att_desc [nb_att]	atts	List of attribute description present in service. (see Table 4)

Response:

GATTM_ADD_SVC_RSP

Description:

Add Service into database request. This message shall be used to allocate a buffer that will be used to describe a service in attribute database.

If start handle is set to zero (invalid attribute handle), ATTM search a free handle block matching with number of attributes to reserve. Else, according to start handle, GATTM checks if attributes to reserve are not overlapping part of existing database

Finally it allocates buffer that

- Describe the database (Block insert in database linked list sorted by start handle)
- Contains attributes configurations and optionally their values.

3.1.2 GATTM_ADD_SVC_RSP

Parameters:

Type	Parameters	Description
uint16_t	start_hdl	Start handle of allocated service in attribute database
uint8_t	status	Return status of service allocation in attribute database (See [4]).

Description:

Message sent back to requester task. It informs about service allocation response. If allocation succeeds, it returns attribute start handle of first attribute.

Status code:

- ATT_ERR_NO_ERROR: If service allocation succeeds.
- ATT_ERR_INVALID_HANDLE: If start_hdl given in parameter and number of attribute overrides some existing services handles.
- ATT_INSUFF_RESOURCE: There is not enough memory to allocate service buffer.

3.1.3 GATTM_DESTROY_DB_REQ – debug only

Parameters:

Type	Parameters	Description
uint16_t	gap_hdl	New GAP Start Handle
uint16_t	gatt_hdl	New GATT Start Handle

Response:

GATTM_DESTROY_DB_RSP

Description:

Fully destroy the attribute database. **This message shall be used only for debug purpose** in order to change database.

3.1.4 GATTM_DESTROY_DB_RSP – debug only

Parameters:

Type	Parameters	Description
uint8_t	status	Return status (See [4])

Description:

Attribute database fully destroyed.

Status code:

- ATT_ERR_NO_ERROR: If request succeeds.

3.2 Service management

This message API shall be used to manage service permissions:

3.2.1 GATTM_SVC_GET_PERMISSION_REQ

Parameters:

Type	Parameters	Description
uint16_t	start_handle	Service start handle

Response:

GATTM_SVC_GET_PERMISSION_RSP

Description:

Get permission settings of service request

3.2.2 GATTM_SVC_GET_PERMISSION_RSP

Parameters:

Type	Parameters	Description
uint16_t	start_handle	Service start handle
uint8_t	perm	Service permissions (see Table 1) **Note:** UUID length not present
uint8_t	status	Command status (See [4])

Description:

Get permission settings of service response

Status code:

- ATT_ERR_NO_ERROR: Command succeeds.
- ATT_ERR_INVALID_HANDLE: Service Handle not available in database

3.2.3 GATTM_SVC_SET_PERMISSION_REQ

Parameters:

Type	Parameters	Description
uint16_t	start_handle	Service start handle
uint8_t	perm	Service permissions (see Table 1)

Response:

GATTM_SVC_SET_PERMISSION_RSP

Description:

Set permission settings of service request.

3.2.4 GATTM_SVC_SET_PERMISSION_RSP

Parameters:

Type	Parameters	Description
uint16_t	start_handle	Service start handle
uint8_t	status	Command status (See [4])

Description:

Set permission settings of service response

Status code:

- ATT_ERR_NO_ERROR: Command succeeds.
- ATT_ERR_INVALID_HANDLE: Service Handle not available in database

3.2.5 GATTM_ SVC _GET_LIST_REQ – debug only

Parameters:

None

Response:

GATTM_GET_SVC_LIST_RSP

Description:

DEBUG ONLY: Retrieve list of services request

3.2.6 GATTM_ SVC_GET_LIST_RSP – debug only

Parameters:

Type	Parameters	Description
uint8_t	nb_svc	Service start handle
uint8_t	status	Command status (See [4])
struct gattm_svc_info []	svc	Array of information about services

(struct gattm_svc_info)

Type	Parameters	Description
uint16_t	start_hdl	Service start handle
uint16_t	end_hdl	Service end handle
uint16_t	task_id	Service task_id
uint8_t	perm	Service permissions (see Table 1)

Description:

DEBUG ONLY: Retrieve list of services response

Status code:

- ATT_ERR_NO_ERROR: Command succeeds.
- ATT_ERR_INVALID_HANDLE: Service Handle not available in database

3.3 Attribute management

This message API shall be used to manage attribute:

- Permissions
- Value

3.3.1 GATTM_ATT_GET_PERMISSION_REQ

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle

Response:

GATTM_ATT_GET_PERMISSION_RSP

Description:

Retrieve permission settings of attribute.

3.3.2 GATTM_ATT_GET_PERMISSION_RSP

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle
uint16_t	perm	Attribute Permission (see Table 2)
uint16_t	ext_perm	Attribute Extended Permission bit field: (see Table 3**)**
uint8_t	status	Command status (See [4])

Description:

Returns permission value of attribute (see Table 2)

Status code:

- ATT_ERR_NO_ERROR: Command succeeds.
- ATT_ERR_INVALID_HANDLE: Handle not available in database

3.3.3 GATTM_ATT_SET_PERMISSION_REQ

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle
uint16_t	perm	Attribute Permission (see Table 2)
uint16_t	ext_perm	Attribute Extended Permission bit field: (see Table 3**)**

Response:

GATTM_ATT_SET_PERMISSION_RSP

Description:

Changes attribute permission (see Table 2)

3.3.4 GATTM_ATT_SET_PERMISSION_RSP

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle
uint8_t	status	Command status (See [4])

Description:

Status code:

- ATT_ERR_NO_ERROR: Command succeeds.
- ATT_ERR_INVALID_HANDLE: Handle not available in database

3.3.5 GATTM_ATT_GET_VALUE_REQ

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle

Response:

GATTM_ATT_GET_VALUE_RSP

Description:

Retrieve attribute value.

3.3.6 GATTM_ATT_GET_VALUE_RSP

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle
uint16_t	length	Value length
uint8_t	status	Command status (See [4])
uint8_t[length]	value	Attribute value

Description:

Returns value of attribute. Reading value didn't require any permission, it directly copy attribute value in a kernel message

Status code:

- ATT_ERR_NO_ERROR: Command succeeds.
- ATT_ERR_INVALID_HANDLE: Handle not available in database

3.3.7 GATTM_ATT_SET_VALUE_REQ

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle
uint16_t	length	Value length
uint8_t[length]	value	Attribute value

Response:

GATTM_ATT_SET_VALUE_RSP

Description:

Changes attribute value.

This kernel message change attributes value, but it doesn't trigger any notification or indication message to peer device. This shall be done in addition using GATT message API.

3.3.8 GATTM_ATT_SET_VALUE_RSP

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle
uint8_t	status	Command status (See [4])

Description:

Status code:

- ATT_ERR_NO_ERROR: Command succeeds.
- ATT_ERR_INVALID_HANDLE: Handle not available in database
- ATT_ERR_INVALID_ATTRIBUTE_VAL_LEN: Length parameter exceeds maximum attribute length

3.3.9 GATTM_ATT_GET_INFO_REQ – Debug Only

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle

Response:

GATTM_ATT_GET_INFO_RSP

Description:

Retrieve information of attribute request.

3.3.10 GATTM_ATT_GET_INFO_RSP – Debug Only

Parameters:

Type	Parameters	Description
uint8_t	status	Return status (See [4])
uint8_t	uuid_len	UUID Length
uint16_t	handle	Attribute handle
uint16_t	perm	Attribute permissions (see Table 2)
uint16_t	ext_perm	Attribute Extended Permission bit field: (see Table 3**)**
uint8_t[uuid_len]	uuid	UUID Value

Description:

Retrieve information of attribute request

Status code:

- ATT_ERR_NO_ERROR: Command succeeds.
- ATT_ERR_INVALID_HANDLE: Handle not available in database
- ATT_ERR_INVALID_ATTRIBUTE_VAL_LEN: Length parameter exceeds maximum attribute length

4 GATT Controller (GATTC)

Multi-instantiated, GATT controller task is related to a BLE connection. Instance number of the task is related to connection index provided by GAP at link establishment.

This interface is used in client role to discover, read and write attribute of the peer device. Moreover, It can receive notification or indication events.

On Server role, this interface is used to be notified when modification of database is requested by peer device, and to send indication or notification events to peer.

The GATT controller block has handlers for these messages, defined in gattc_task files (.h/.c).

4.1 Request Flags

The block uses request flag options embedded in the interface message sent to GATT. This flag ensures correct handling of the request from the application or profile for request interfaces that handle multiple types of operations.

Value	Flag	Description
0x00	GATTC_NO_OP	No operation
	MTU Negotiation	
0x01	GATTC_MTU_EXCH	Perform MTU exchange
	Attribute Discovery	
0x02	GATTC_DISC_ALL_SVC	Discover all services
0x03	GATTC_DISC_BY_UUID_SVC	Discover services by UUID
0x04	GATTC_DISC_INCLUDED_SVC	Discover included services
0x05	GATTC_DISC_ALL_CHAR	Discover all characteristics
0x06	GATTC_DISC_BY_UUID_CHAR	Discover characteristic by UUID
0x07	GATTC_DISC_DESC_CHAR	Discover characteristic descriptor
	Read Attribute	
0x08	GATTC_READ	Read attribute
0x09	GATTC_READ_LONG	Read long attribute
0x0A	GATTC_READ_BY_UUID	Read attribute by UUID
0x0B	GATTC_READ_MULTIPLE	Read multiple attribute
	Write Attribute	
0x0C	GATTC_WRITE	Write attribute
0x0D	GATTC_WRITE_NO_RESPONSE	Write no response
0x0E	GATTC_WRITE_SIGNED	Write signed
0x0F	GATTC_EXEC_WRITE	Execute write
	Registering to peer device event	
0x10	GATTC_REGISTER	Register to peer device events
0x11	GATTC_UNREGISTER	Unregister from peer device events
	Sending events to peer device	
0x12	GATTC_NOTIFY	Send an attribute notification
0x13	GATTC_INDICATE	Send an attribute indication
0x14	GATTC_SVC_CHANGED	Send a service changed indication
	Service Discovery Procedure	
0x15	GATTC_SDP_DISC_SVC	Search specific service
0x16	GATTC_SDP_DISC_SVC_ALL	Search for all services
0x17	GATTC_SDP_DISC_CANCEL	Cancel Service Discovery Procedure

Table 5: GATTC Operation Flags

4.2 Generic Interface

The generic GATT Controller interface includes commands and events common to GATT server and client.

4.2.1 GATTC_CMP_EVT

Parameters:

Type	Parameters	Description
uint8_t	operation	GATTC request type (see Table 5)
uint8_t	status	Status of the operation (See [4])
uint16_t	seq_num	Operation sequence number - provided when operation is started

Description:

Complete event for GATT operation. This is the generic complete event for GATT operations. All operation triggers this event when operation is finished.

It is **strongly recommended** that the application/upper layer should wait for the GATTC_CMP_EVT of the current GATT request before making additional request. This ensures proper and sequential execution of attribute operations/requests by the BLE stack.

Note: The seq_num parameter is a sequence number provided by tasks using different GATTC commands. This sequence number is never modified by GATTC task but only a copy of this parameter is performed when operation is completed.

4.2.2 GATTC_TRANSACTION_TO_ERROR_IND

Parameters:

None

Description:

Message triggered to main application when an Attribute transaction has timed out. This means that no more attribute transaction will be accepted by device on current connection.

Note: Disconnection of the link must be performed by application.

4.3 GATT Client

Client side of the API, it provides operation to discover, read and modify peer device attribute database and convey value modification events to registered profiles/applications.

4.3.1 Configuration

This is intended for setting the Maximum Transmission Unit (MTU) of the link for GATT transactions. The client and the server will exchange this information to inform the peer of their sending bandwidth.

4.3.1.1 GATT_EXC_MTU_CMD

Parameters:

Type	Parameters	Description
uint8_t	operation	GATT request type (see Table 5) - GATTC_MTU_EXCH
uint16_t	seq_num	Operation sequence number

Response:

GATTC_MTU_CHANGED_IND: triggered when MTU has been negotiated

GATTC_CMP_EVT: when command is proceed

Description:

Start the MTU exchange procedure. The MTU sent by the device will be the MTU set during the configuration of the device.

4.3.1.2 GATTC_ MTU_CHANGED_IND

Parameters:

Type	Parameters	Description
uint16_t	operation	Exchanged MTU value
uint16_t	seq_num	Operation sequence number

Description:

Event triggered when attribute MTU value changed due to an MTU negotiation over ATT has been performed.

4.3.2 Discovery Procedure

Discovery of services exposed by the GATT server to the GATT client is an important interface for the RW-BLE GATT.

Once the primary services are discovered, additional information can be accessed including characteristic and relationship discovery. RW-BLE GATT provides means for the user to discover the services by group type and by UUID.

4.3.2.1 GATTC_DISC_CMD

Parameters:

Type	Parameters	Description
uint8_t	operation	GATTC request type (see Table 5) - GATTC_DISC_ALL_SVC - GATTC_DISC_BY_UUID_SVC - GATTC_DISC_INCLUDED_SVC - GATTC_DISC_ALL_CHAR - GATTC_DISC_BY_UUID_CHAR - GATTC_DISC_DESC_CHAR
uint8_t	uuid_len	UUID length (2, 4 or 16 bytes)
uint16_t	seq_num	operation sequence number
uint16_t	start_hdl	Discovery Start handle range
uint16_t	end_hdl	Discovery End handle range
uint8_t[uuid_len]	uuid	UUID searched - LSB first

Response:

GATTC_CMP_EVT: when command is accepted and processed.

GATTC_DISC_SVC_IND: Triggered during a service discovery.

GATTC_DISC_SVC_INCL_IND: Triggered during an included service discovery.

GATTC_DISC_CHAR_IND: Triggered during a characteristic discovery.

GATTC_DISC_CHAR_DESC_IND: Triggered during a characteristic descriptor discovery.

Description:

Discover services, included services, characteristics or characteristic descriptor exposed by peer device in its attribute database.

- **Service Discovery:** Triggers GATTC_DISC_SVC_IND events when a service is founded.
    - GATTC_DISC_ALL_SVC: This operation should be used to discover all services in given handle range. This operation stops when searched UUID is found or if no more services are available in peer device. To find all services, UUID searched shall be set to 0x0000.
    - GATTC_DISC_BY_UUID_SVC: This operation should be used to discover all services corresponding to search UUID in given handle range. This operation stops when s no more services are available in peer device.

- **Included Service discovery:** Triggers GATTC_DISC_SVC_INCL_IND events when an included service is founded within provided handle range
    - GATTC_DISC_INCLUDED_SVC operation, set UUID to 0x0000.

- **Characteristic discovery:** Triggers GATTC_DISC_CHAR_IND event when characteristic is founded within provided handle range.
    - GATTC_DISC_ALL_CHAR: This operation should be used to discover all characteristic in given handle range. Set UUID to 0x0000.
    - GATTC_DISC_BY_UUID_CHAR: This operation should be used to discover specific searched UUID in provided handle range.

- **Characteristic Descriptor discovery:** Triggers GATTC_DISC_CHAR_DESC_IND event when characteristic descriptor is found within the provided handle range.

  The application should know the location (attribute handles) of the characteristic descriptors within the discovered characteristic. The start and end handles provided in the command must be the **actual range of the characteristic descriptors**.

  It is assumed that a characteristic discovery was performed for this particular characteristic prior to discovering its characteristic descriptors.

  - GATTC_DISC_DESC_CHAR operation, set UUID to 0x0000.

4.3.2.2 GATTC_DISC_SVC_IND

Parameters:

Type	Parameters	Description
uint16_t	start_hdl	Service start handle
uint16_t	end_hdl	Service end handle
uint8_t	uuid_len	UUID length (2, 4 or 16 bytes)
uint8_t[uuid_len]	uuid	UUID searched - LSB first

Description:

Indication triggered when service(s) is/are found during discovery operation.

4.3.2.3 GATTC_DISC_SVC_INCL_IND

Parameters:

Type	Parameters	Description
uint16_t	attr_hdl	Element Handle
uint16_t	start_hdl	Service start handle
uint16_t	end_hdl	Service end handle
uint8_t	uuid_len	UUID length (2, 4 or 16 bytes)
uint8_t[uuid_len]	uuid	UUID searched - LSB first

Description:

Indication triggered when included service(s) is/are found during discovery operation.

4.3.2.4 GATTC_DISC_CHAR_IND

Parameters:

Type	Parameters	Description
uint16_t	attr_hdl	Element Handle
uint16_t	pointer_hdl	pointer attribute handle to UUID
uint8_t	prop	Characteristic properties
uint8_t	uuid_len	UUID length (2, 4 or 16 bytes)
uint8_t[uuid_len]	uuid	UUID searched - LSB first

Description:

Indication triggered when characteristic(s) is/are found during discovery operation.

4.3.2.5 GATTC_DISC_CHAR_DESC_IND

Parameters:

Type	Parameters	Description
uint16_t	attr_hdl	Element Handle
uint8_t	uuid_len	UUID length (2, 4 or 16 bytes)
uint8_t[uuid_len]	uuid	UUID searched - LSB first

Description:

Indication triggered when characteristic descriptor(s) is/are found during discovery operation.

4.3.3 Read Characteristic

RW-BLE GATT provides a way for a peer characteristic to be read. The interface supports characteristic read in different formats and lengths.

4.3.3.1 GATTC_READ_CMD

Parameters:

Type	Parameters	Description
uint8_t	operation	GATTC request type (see Table 5) - GATTC_READ - GATTC_READ_LONG - GATTC_READ_BY_UUID - GATTC_READ_MULTIPLE
uint8_t	nb	number of read (only used for multiple read)
uint16_t	seq_num	operation sequence number
union gattc_read_req	req	request union according to read type (see union gattc_read_req)

Type	Parameters	Description
struct gattc_read_simple	simple	Simple Read (GATTC_READ or GATTC_READ_LONG)
struct gattc_read_by_uuid	by_uuid	Read by UUID (GATTC_READ_BY_UUID)
struct gattc_read_multiple[nb]	multiple	Read Multiple short characteristic (GATTC_READ_MULTIPLE)

Table 6: union gattc_read_req

Type	Parameters	Description
uint16_t	handle	Attribute handle
uint16_t	offset	Start offset in data payload
uint16_t	length	Length of data to read (0 = read all)

Table 7: struct gattc_read_simple

Type	Parameters	Description
uint16_t	start_hdl	Searched start handle
uint16_t	end_hdl	Searched end handle
uint8_t	uuid_len	UUID length (2, 4 or 16 bytes)
uint8_t[uuid_len]	uuid	UUID searched

Table 8: struct gattc_read_ by_uuid

Type	Parameters	Description
uint16_t	handle	Attribute handle
uint16_t	len	Known Handle length (shall be > 0)

Table 9: struct gattc_read_multiple

Response:

GATTC_CMP_EVT: when command is proceed

GATTC_READ_IND: Triggered when an attribute has been read

Description:

Read characteristic(s) from peer attribute database.

- **Simple Read**: GATTC_READ or GATTC_READ_LONG (read or read long request). Just set the handle to read. It's possible to start reading from an offset and limit read size to a specific length. To read all attribute, those variables shall be set to 0

- **Read by UUID**: GATTC_READ_BY_UUID. Read first variable found with searched UUID in provided handle range. Note that if it's a long attribute it will return only first read bytes returned by ATT_READ_BY_TYPE_RESP.

- **Read Multiple**: GATTC_READ_MULTIPLE. Read multiple handle in same time using ATT_READ_MULTIPLE_REQ. Size of peer attribute shall be known, else GATTC_CMP_EVT will return a status error.

4.3.3.2 GATTC_READ_IND

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute Handle
uint16_t	offset	Read Offset
uint16_t	length	Read data length
uint8_t[length]	value	Read data value

Description:

Event triggered when the requested attribute handle has been read.

4.3.4 Write Characteristic

RW-BLE GATT provides a way for a peer characteristic to be written. The interface supports characteristic write in different formats and lengths.

4.3.4.1 GATTC_WRITE_CMD

Parameters:

Type	Parameters	Description
uint8_t	operation	GATTC request type (see Table 5) - GATTC_WRITE - GATTC_WRITE_NO_RESPONSE - GATTC_WRITE_SIGNED
uint8_t	auto_execute	Perform automatic execution (only relevant for GATTC_WRITE) If 0, an ATT Prepare Write will be used and GATTC_EXECUTE_WRITE_CMD will be sent to execute write request.
uint16_t	seq_num	operation sequence number
uint16_t	handle	Attribute Handle
uint16_t	offset	Write offset
uint16_t	length	Write length
uint16_t	cursor	Internal write cursor **shall be initialized to 0**
uint8_t[length]	value	Data value to write

Response:

GATTC_CMP_EVT: when command is proceed

Description:

This command shall be used to modify peer device attribute handle.

- **Write Characteristic Value:** If operation is GATTC_WRITE, auto_execute set to 1 (enabled), offset to 0 and data length ≤ (ATT_MTU - 3), in that case, ATT_WRITE_REQUEST will be sent to peer device.

- **Write Long Characteristic Value:** If operation is GATTC_WRITE, auto_execute set to 1 (enabled), offset to 0 and data length > (ATT_MTU - 3), in that case, several ATT_PREPARE_WRITE_REQUEST will be sent to peer device and finally an ATT_EXECUTE_WRITE_REQUEST will be sent to peer device.

- **Write Without Response Value:** If operation is GATTC_WRITE_NO_RESPONSE, in that case, ATT_WRITE_COMMAND will be sent to the peer device and GATTC_CMP_EVT will be triggered as soon as packet has been sent over the air.

- **Write Signed Value:** If operation is GATTC_WRITE_SIGNED, in that case, ATT_WRITE_SIGNED_COMMAND will be sent to the peer device and GATTC_CMP_EVT will be triggered as soon as packet has been sent over the air.

- **Reliable Writes:** If operation is GATTC_WRITE with auto_execute set to 0 (disabled), several ATT_PREPARE_WRITE_REQUEST PDUs will be sent to the peer device. Requesting this command several times before sending GATTC_EXECUTE_WRITE_CMD is considered as doing a Reliable writes.

4.3.4.2 GATTC_EXECUTE_WRITE_CMD

Parameters:

Type	Parameters	Description
uint8_t	operation	GATTC request type (see Table 5) - GATTC_EXEC_WRITE only
uint8_t	Execute	- **1:** perform pending write operations - **0:** cancel pending write operations
uint16_t	seq_num	Operation sequence number

Response:

GATTC_CMP_EVT: when command is proceed

Description:

This command is used to execute or cancel pending prepare write operations on peer device attributes.

4.3.5 Event Interface

Characteristics can be notified and indicated by peer device.

Interface for the profiles or higher layer is necessary to have efficient connection to GATT.

4.3.5.1 GATTC_REG_TO_PEER_EVT_CMD

Parameters:

Type	Parameters	Description
uint8_t	operation	GATTC request type (see Table 5) - GATTC_REGISTER - GATTC_UNREGISTER
uint8	padding	Padding unused
uint16_t	seq_num	Operation sequence number
uint16_t	svc_shdl	Service start handle
uint16_t	svc_ehdl	Service end handle

Response:

GATTC_CMP_EVT: when command is proceed

Description:

Register or unregister from peer device events such as indication or notifications on a specific service attribute handle range on dedicated connection.

- **Register**: Set operation to GATTC_REGISTER
- **Unregister**: Set operation to GATTC_UNREGISTER

4.3.5.2 GATTC_EVENT_IND

Parameters:

Type	Parameters	Description
uint8_t	type	GATTC request type (see Table 5) - GATTC_NOTIFY
uint8	length	Data length
uint16_t	handle	Attribute handle
uint8_t[length]	value	New attribute value

Description:

This message is triggered to registered task (see 4.3.5.1). This event contains new value of peer attribute handle.

4.3.5.3 GATTC_EVENT_REQ_IND

Parameters:

Type	Parameters	Description
uint8_t	type	GATTC request type (see Table 5) - GATTC_INDICATE
uint8	length	Data length
uint16_t	handle	Attribute handle
uint8_t[length]	value	New attribute value

Description:

This message is triggered to registered task (see 4.3.5.1). This event contains new value of peer attribute handle. This event shall be acknowledged by GATTC_EVENT_CFM message.

4.3.5.4 GATTC_EVENT_CFM

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute handle

Description:

This message shall be sent after receiving a GATTC_EVENT_REQ_IND to trigger ATT_HANDLE_VALUE_CONFIRMATION PDU in order to acknowledge received indication.

4.3.6 Service Discovery Procedure

Characteristics can be notified and indicated by peer device.

Interface for the profiles or higher layer is necessary to have efficient connection to GATT.

❖ **gattc_sdp_att_type**

Value	Flag	Description
0x00	GATTC_SDP_NONE	No Attribute Information
0x01	GATTC_SDP_INC_SVC	Included Service Information
0x02	GATTC_SDP_ATT_CHAR	Characteristic Declaration
0x03	GATTC_SDP_ATT_VAL	Attribute Value
0x04	GATTC_SDP_ATT_DESC	Attribute Descriptor

Table 10: Service Discovery Attribute type

4.3.6.1 GATTC_SDP_SVC_DISC_CMD

Parameters:

Type	Parameters	Description
uint8_t	operation	GATTC request type (see Table 5) - GATTC_SDP_DISC_SVC - GATTC_SDP_DISC_SVC_ALL - GATTC_SDP_DISC_CANCEL
uint8_t	uuid_len	Service UUID Length (2, 4 or 16 bytes)
uint16_t	seq_num	operation sequence number
uint16_t	start_hdl	Service start handle
uint16_t	end_hdl	Service end handle
uint8_t[16]	uuid	Service UUID to search

Response:

GATTC_SDP_SVC_IND: triggered when a service has been discovered

GATTC_CMP_EVT: when command is proceed

Description:

This command can be used to perform a service discovery using GATTC discovery procedures.

This discovery is able to search all (GATTC_SDP_DISC_SVC_ALL) or a specific (GATTC_SDP_DISC_SVC) service.

This discovery automatically searches for included services, characteristics and descriptors.

This procedure can be canceled using GATTC_SDP_DISC_CANCEL operation code.

4.3.6.2 GATTC_SDP_SVC_IND

Parameters:

Type	Parameters	Description
uint8_t	uuid_len	Service UUID Length (2, 4 or 16 bytes)
uint8_t[16]	uuid	Service UUID found
uint16_t	start_hdl	Service start handle
uint16_t	end_hdl	Service end handle
union gattc_sdp_att_info	info	Attribute information present in the service (see Table 11) (length = end_hdl - start_hdl)

Type	Parameters	Description
uint8_t	att_type	Attribute Type (first octet of following structure, see Table 10)
struct gattc_sdp_att_char	att_char	Information about attribute characteristic (see Table 12)
struct gattc_sdp_include_svc	inc_svc	Information about included service (see Table 13)
struct gattc_sdp_att	att	Information about attribute (see Table 14)

Table 11: union gattc_sdp_att_info

Type	Parameters	Description
uint8_t	att_type	Attribute Type (see Table 10) - GATTC_SDP_ATT_CHAR: Characteristic Declaration
uint8_t	prop	Value property
uint16_t	handle	Value Handle

Table 12: struct gattc_sdp_att_char

Type	Parameters	Description
uint8_t	att_type	Attribute Type (see Table 10) - GATTC_SDP_INC_SVC: Included Service Information
uint8_t	uuid_len	Included service UUID Length (2, 4 or 16 bytes)
uint8_t[16]	uuid	Included Service UUID
uint16_t	start_hdl	Included service Start Handle
uint16_t	end_hdl	Included service End Handle

Table 13: struct gattc_sdp_include_svc

Type	Parameters	Description
uint8_t	att_type	Attribute Type (see Table 10) - GATTC_SDP_ATT_VAL: Attribute Value - GATTC_SDP_ATT_DESC: Attribute Descriptor
uint8_t	uuid_len	Attribute UUID Length (2, 4 or 16 bytes)
uint8_t[16]	uuid	Attribute UUID

Table 14: struct gattc_sdp_att

Description:

This event is triggered when Service Discovery procedure has found and browse a service.

This structure can be big due to memory allocated for 128bits UUIDs and should be free as soon as possible by profile/application.

4.4 GATT Server

Server side of the API, provides procedure to inform profile service about read or modification request of attributes, and for profile to inform peer device that an attribute value changes.

4.4.1 Notify and Indication Characteristic

Characteristics can be notified and indicated. These actions originate from GATT server.

Notification would not expect attribute protocol layer acknowledgement.

4.4.1.1 GATTC_SEND_EVT_CMD

Parameters:

Type	Parameters	Description
uint8_t	operation	GATTC request type (see Table 5) - GATTC_NOTIFY - GATTC_INDICATE
uint16_t	seq_num	operation sequence number
uint16_t	handle	Characteristic handle
uint16_t	length	Length of attribute value
uint8_t[length]	value	Attribute data value

Response:

GATTC_CMP_EVT: when command is proceed

Description:

This request triggers a notification or indication event on specified characteristic.

- **Sending Notification**: operation shall be set to GATTC_NOTIFY. GATTC_CMP_EVT message is sent back as soon as notification PDU has been sent over the air.

- **Sending Indication**: operation shall be set to GATTC_INDICATE. . GATTC_CMP_EVT message is sent back as soon as ATT_HANDLE_VALUE_CONFIRMATION PDU is received by device confirming that indication has been correctly received by peer device.

Note: If attribute value is present in database, it's role of profile to update it.

4.4.2 Read request from peer device

A read request for an attribute that is not currently available in database would trigger a request to the profile task that manages the service.

4.4.2.1 GATTC_READ_REQ_IND

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute Handle that has to be read

Response:

GATTC_READ_CFM message shall be send back by upper layer to confirm requested read value execution

Description:

Reception of peer device read request. This message is convey to profile in charge of the service handle. It is Profile role provide the attribute value.

Note: This message isn't triggered if value is present in attribute database.

4.4.2.2 GATTC_READ_CFM

Parameters:

Type	Parameters	Description
uint16_t	handle	Handle of the attribute written
uint16_t	length	Attribute data length
uint8_t	status	Status of write command execution by upper layers (see [4])
uint8_t[length]	value	Attribute data value

Description:

Read confirmation of upper layer that will trigger an answer to peer device read request.

4.4.3 Write request from peer device

Characteristics value modification is handled by profile.

4.4.3.1 GATTC_WRITE_REQ_IND

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute Handle that has to be written
uint16_t	offset	Offset at which the data has to be written
uint16_t	length	Data length to be written
uint8_t[length]	value	Data value to write

Response:

GATTC_WRITE_CFM message shall be send back by upper layer to confirm write execution

Description:

Reception of peer device write request. This message is convey to profile in charge of the service handle. Attribute database isn't modified when receiving this message; it is Profile role to modify it if value is present in the database.

4.4.3.2 GATTC_WRITE_CFM

Parameters:

Type	Parameters	Description
uint16_t	handle	Handle of the attribute written
uint8_t	status	Status of write command execution by upper layers (see [4])

Description:

Write confirmation of upper layer that will trigger an answer to peer device write request.

4.4.3.3 GATTC_ATT_INFO_REQ_IND

Parameters:

Type	Parameters	Description
uint16_t	handle	Attribute Handle that has to be written

Response:

GATTC_ATT_INFO_CFM message shall be send back by upper layer to confirm attribute info

Description:

Request Attribute info to upper layer - could be trigger during prepare write to check if attribute modification is authorized by profile/application or not and to get current attribute length.

4.4.3.4 GATTC_ATT_INFO_CFM

Parameters:

Type	Parameters	Description
uint16_t	handle	Handle of the attribute written
uint16_t	length	Current length of the attribute
uint8_t	status	Status of write command execution by upper layers (see [4])

Description:

Attribute Information confirmation message to inform if peer device is authorized to modify attribute value, and to get current attribute length.

4.4.4 Service Changed

Event triggered when Peer Device set Service Changed Client Configuration.

4.4.4.1 GATTC_SVC_CHANGED_CFG_IND

Parameters:

Type	Parameters	Description
uint16_t	ind_cfg	Client Characteristic Configuration Descriptor value

Description:

This message is sent to the application each time the Client Characteristic Configuration descriptor value is written by the peer device.

It contains the new configuration set by the device. The application shall take care of the given configuration before sending an indication. It will also keep this configuration disconnection of a link with a bonded device in order to remind it once a new connection occurs.

This value is part of GATT bond data and should be store in a non-volatile memory. This value can be restored with GAPC_CONNECTION_CFM message (see [3])

4.4.4.2 GATTC_SEND_SVC_CHANGED_CMD

Parameters:

Type	Parameters	Description
uint8_t	operation	GATTC request type (see Table 5) - GATTC_SVC_CHANGED
uint16_t	seq_num	operation sequence number
uint16_t	svc_shdl	Start handle of the affected part of the database
uint16_t	svc_ehdl	End handle of the affected part of the database

Description:

This message is used by the application to require sending of a Service Change Characteristic Indication.

The Service Changed characteristic is used to indicate to connected devices that services in the database have changed (i.e added, removed, modified). It shall be used to indicate to a bonded device that the database content has been modified between the last disconnection and the reconnection.

References

[1]	**Title**	Specification of the Bluetooth System		
	Reference	Bluetooth Specification		
	Version	4.2	**Date**	2014-12-02
	Source	Bluetooth SIG		

[2]	**Title**	RW-BLE Host Functional Specification		
	Reference	RW-BLE-SW-HOST-FS		
	Version	8.00	**Date**	2015-07-17
	Source	RivieraWaves SAS		

[3]	**Title**	GAP Interface Specification		
	Reference	RW-BLE-GAP-IS		
	Version	8.00	**Date**	2014-07-17
	Source	RivieraWaves SAS		

[4]	**Title**	RW BLE Host Error Code Interface Specification		
	Reference	RW-BLE-HOST-ERR-CODE-IS		
	Version	8.00	**Date**	2014-07-17
	Source	RivieraWaves SAS		