



CoAP 协议一致性测试

帮助文档

Beijing SmeshLink Technology Co., Ltd.

<http://www.smeshlink.com>

2014-09-09

美信凌科享有解释权，任何更新以最新日期为准。

目 录

1 CoAP 协议介绍.....	4
2 CoAP 协议测试软件操作方法.....	5
2.1 快速开始	5
2.2 用例管理	6
2.2.1 默认用例	6
2.2.2 导入用例	7
2.2.3 添加用例	8
2.2.4 修改用例	8
2.2.5 导出用例	9
2.2.6 删除用例	9
2.3 修改参数设置	10
2.3.1 请求部分	10
2.3.2 响应部分	11
2.4 查看并导出测试结果	12
2.4.1 直接从软件中查看测试结果	12
2.4.2 导出测试用例结果	13
2.5 详细示例	14
2.5.1 示例一	14
2.5.2 示例二	16
3 CoAP 浏览器.....	19
3.1 详细示例	20
3.1.1 示例一	20

3.1.2 示例二	23
4 附录	27

1 CoAP 协议介绍

受约束的应用协议（CoAP）是一种软件协议，旨在应用于非常简单的电子设备中，从而使他们能够在互联网上进行交互式通信。CoAP 协议可以通过标准的因特网重点应用于小型低功率传感器、开关、阀门和这种类似需要被控制或远程监督的组件。CoAP 是一个应用层协议，该协议用于在资源受限的网络设备，例如无线传感器网络节点。CoAP 协议被设计得很容易转换成 HTTP 协议，同时也能满足一些特殊的要求，例如可支持多播，降低开销，和实现协议的简化。多播，低开销，以及简单性对于物联网（IoT）和机器对机器（M2M）设备来说是非常重要的，因为他们需要更深层次的嵌入性，并且他们需要比传统的互联网设备有更低的成本和功率。因此，CoAP 协议的效率非常重要。

互联网工程任务组（IETF）的 RESTful 约束环境（CORE）工作组已经做了许多这个协议的标准化工作，工作组在设计 COAP 协议时充分考虑了以下功能：

- 基于 REST 协议的设计最大限度地减少了 HTTP 映射的复杂性
- 报头开销少和分析的复杂度低
- URI 和内容类型的支持
- 支持由已知 COAP 服务提供的资源发现

COAP 与 HTTP 的映射也被定义，从而允许代理来建立提供通过 HTTP 访问 COAP 资源以统一的方式。

2 CoAP 协议一致测试软件操作方法

2.1 快速开始

首先下载“CoAP 协议一致性测试软件”，下载地址为 <http://www.smeshlink.com/app/coapterster>，点击“安装”即可。安装完成后自动进入如图 2-1 所示操作界面（已安装软件的可直接打开“CoAPTester.exe”）：

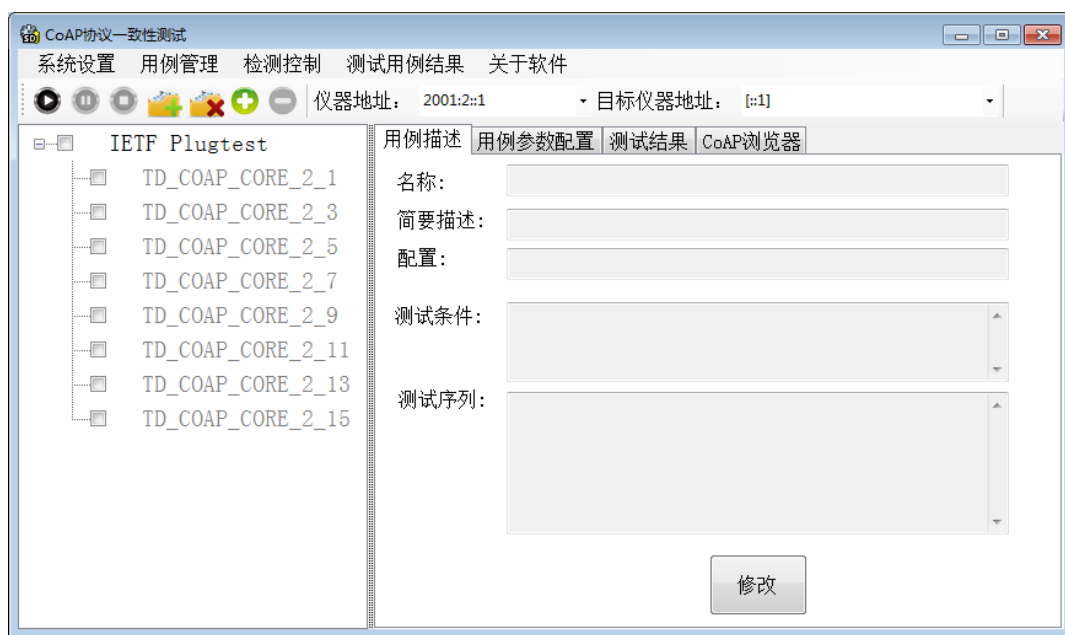


图 2-1

其中，状态栏中各个按钮依次为，“运行用例”“暂停运行用例”“停止运行用例”“添加组”“删除组”“添加用例”“删除用例”。

左下方区域为用例列表，刚打开软件时该区域显示的是自带的默认用例。

右下方 tab 标签页依次为“用例描述”“用例参数配置”“测试结果”“CoAP 浏览器”。其中，“用例描述”可以修改左侧选中用例的一些说明信息，“用例参数配置”可以对协议测试的参数进行设置，“测试结果”可以查看用例测试结果，“CoAP 浏览器”可以以更明显直观的方式访问服务端资源。

接下来便可以在该操作界面中进行各项操作，但在运行用例前必须打开菜单栏中“系统设置”下的“打开服务器端”（如图 2-2 所示），该程序负责提供资源供客户端请求，并且该窗口在运行过程中不能关闭。

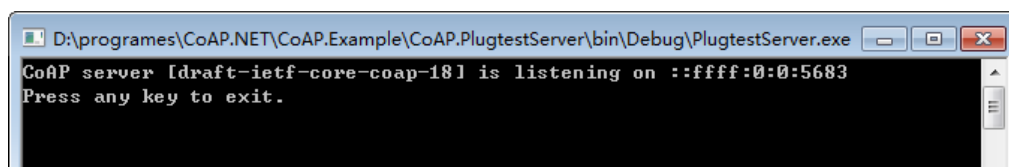


图 2-2

● **注意！** 在运行用例前必须打开该窗口，且测试协议一致性过程中，该窗口不能被关闭。

2.2 用例管理


软件中的测试用例是以 xml 格式存储的，用户可以在软件中添加、修改、删除用例并将最终用例导出保存为“*.xml”文件，以便下次测试时可以快速、方便的从已有测试用例文件中导入测试用例。

选择菜单栏中的“用例管理”，会出现一个下拉菜单，其子菜单项包括“默认用例”、“导入用例”、“导出用例”、“添加用例”、“删除用例”。

选择“默认用例”会自动导入名为“standard.xml”的默认用例，用户可对默认用例进行各种操作及查看结果从而对该软件进行初步了解。

选择“导入用例”可从其他地方导入测试用例，从而对导入的用例进行协议一致性测试。

选择“导出用例”可将用例列表中所有的用例导出保存下来。

选择“添加用例”和“删除用例”即可添加和删除所选用例，并且这两项功能可通过按钮得到快速实现。

2.2.1 默认用例

打开“CoAPTester.exe”后可以看到界面左侧有一组用例，即为“默认用例”。正常情况下，运行默认用例的结果将全部为“通过”。



图 2-3

若打算对默认用例进行修改等操作，敬请参考以下普通用例操作方法。

2.2.2 导入用例

点击菜单栏中的“用例管理”，选择“导入用例”。

例如，当用户打算导入名为“测试用例.xml”的用例时，在菜单栏的“用例管理”下面选择“导入用例”，之后在“打开”对话框中找到存放用例的位置，选择要导入的用例“测试用例.xml”，点击“打开”，新用例便被成功导入左下方用例列表中。

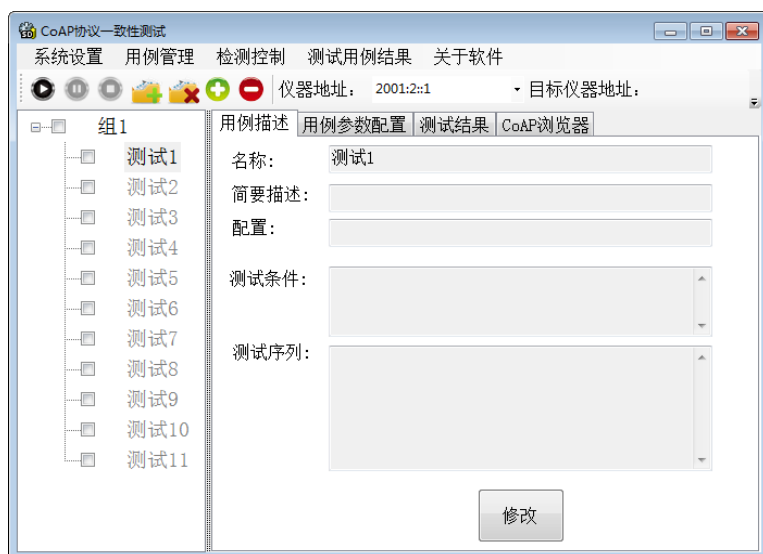


图 2-4

成功导入后，便可以对用例进行其他操作，需要注意的是，如需对其中某一个测试进行操作，则必须使该项用例处于选中状态。

2.2.3 添加用例


点击“用例管理”下的“添加用例”或者工具栏中相应的快捷按钮，可以在所选的组内添加新用例，新生成的用例默认名为“New Test”（如图 2-5 所示）。选中新用例并再次单击，可以修改新用例的名称，或者选中新用例然后在右侧“用例描述”中修改新用例名称。



图 2-5

2.2.4 修改用例

选中某一用例后，可以在右下方的标签页中对该用例进行修改。修改可分为简单的描述性信息修改，和用例参数设置的修改，本部分主要讲述第一种修改，用例参数配置的修改方法敬请参考 2.3。

例如，当用户打算将组 1 中的测试 1 的名称改为“测试用例一”，可先选中该用例，然后在右侧选中“用例描述”。点击下方“修改”按钮，以上所有文本框将会由不可编辑状态变为可编辑状态，此时方可对名称及其他各个选项进行修改。

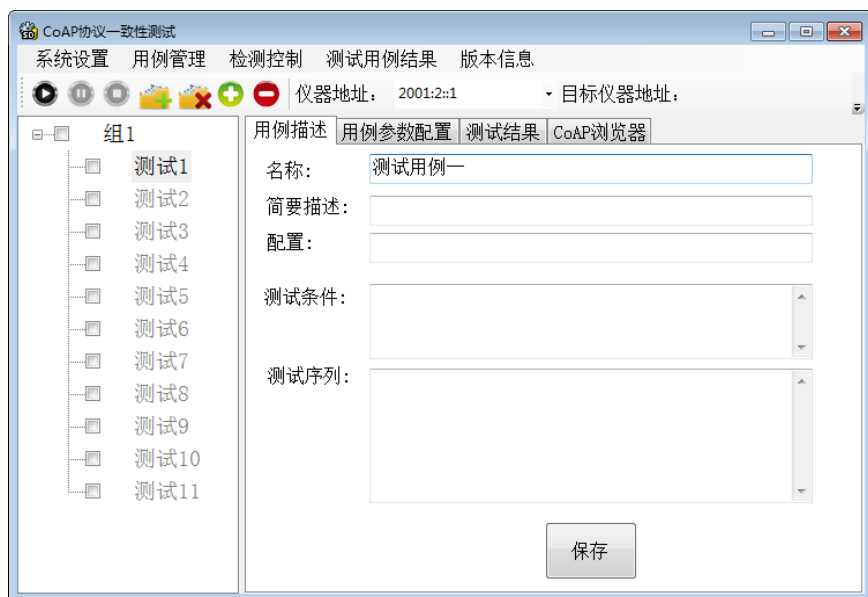


图 2-6

修改完成后点击“保存”按钮，以上输入框将会重新变为不可编辑状态。需要注意的是，只在软件中点击“保存”是不能将数据保存到磁盘中的，这时修改过的数据只是被保存在内存中，下次再打开该用例其实仍然打开的是之前未修改的用例。若想要达到长久保存的效果，必须对已修改用例进行“导出用例”操作。

- **注意！** 修改用例描述时，必须先点击下方“修改”按钮，当描述后的方框由不可编辑状态变成可编辑状态，下方按钮由“修改”变为“保存”后，才能修改用例的描述性信息。


2.2.5 导出用例

点击“用例管理”中的“导出用例”可以导出保存新生成的用例和修改过的用例。

例如，前面用户已经成功修改了“测试用例一”，接着可点击“导出用例”，在“另存为”对话框中选择保存位置并写入将要保存的文件名，保存类型默认为“XML 文件”。

- **注意！** 不管是否修改是否选中，都将被一起导出保存，即“导出用例”操作是将所有的用例都导出保存。

2.2.6 删除用例



点击“用例管理”下的“删除用例”或者点击“删除用例”的快捷按钮，可以删除某一选中的测试用例。需要注意的是，该项操作一旦进行，不可恢复，故操作前请谨慎考虑。建议添加、修改完用例后及时导出用例，以防各种原因造成用例数据丢失。

2.3 修改参数设置

测试用例可包括“请求”(request)和“响应”(response)。其中,“请求”可简单分为“GET Request”、“POST Request”、“PUT Request”和“DELETE Request”,“响应”可简单分为“ACK Response”、“CON Response”、“NON Response”和“RST Response”。



图 2-7

在实际测试当中,可根据需要添加相应的请求和响应。选择“用例参数配置”,可发现下方有两个白色方框,右侧方框里为可添加的请求和响应,若需添加则先在右侧选中相应选项,再点击按钮,若需移除则先在左侧选中相应选项,再点击按钮.

添加合适的请求和响应后,可根据实际需要对他们的参数进行选择 and 设置,一旦被选择,在运行用例时会对该参数对应的期望数据和实际接收数据进行比较,不一致的用例判为“未通过”。

● **注意!** 换行符不同也会导致“未通过”结果,敬请参考 3.1.1 CoAP 浏览器示例一。

2.3.1 请求部分

双击“GET Request”(在实际测试中也可能为“PUT Request”“POST Request”“DELETE Request”),双击后将自动弹出“Request Build Step”对话框,此时可在对话框中对以下选项进行逐一设置。

图 2-8

其中，部分参数的含义如下：

- Method 可以根据实际测试需要选择为“GET”“PUT”“POST”“DELETE”
- Type 可以根据实际测试需要选择为“CON”“NON”“ACK”“RST”
- Payload 为有效负载
- Uri-Path 为请求资源的地址
- Uri-Query 后填写的内容为需要搜索的参数
- Block2 为资源块的序号

2.3.2 响应部分

双击“ACK Response”（在实际测试中也可能为“CON Response”、“NON Response”和“RST Response”），双击后将自动弹出“Response Check Step”对话框，此时可在对话框中对以下选项进行逐一设置。

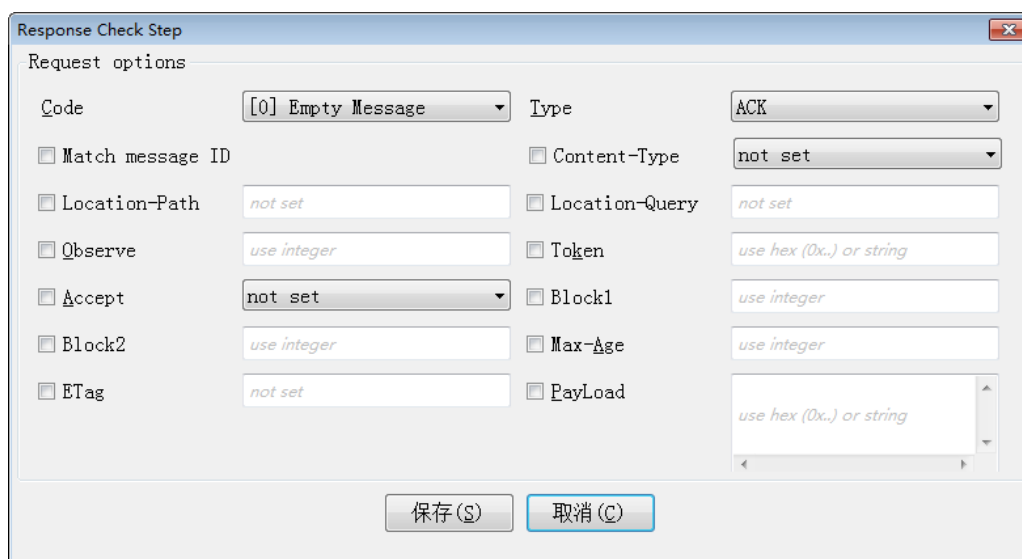


图 2-9

其中，部分 Code 的含义如下：

- **【65】** 2.01 created 通常为进行 POST Request 成功后的期望结果代码
- **【66】** 2.02 deleted 通常为进行 DELETE Request 成功后的期望结果代码
- **【69】** 2.05 content 通常为进行 GET Request 和 PUT Request 成功后的期望结果代码
- **【132】** 4.04 not found 当 Uri-Path 填写错误时通常返回该结果，即无法获取该资源

2.4 查看并导出测试结果

完成以上设置后便可运行用例，得到测试结果。测试结果有两种查看方式，一为直接在该软件中选择“测试结果”查看，二为在“测试用例结果”的下拉菜单中选择“导出测试用例结果”，其中导出结果默认保存格式为 XML 文件，但会同时生成 HTML 格式的测试结果。

2.4.1 直接从软件中查看测试结果

运行用例后会在界面右侧自动显示出测试结果，结果可分为“通过”和“未通过”两种，“通过”的结果不再显示详细信息，“未通过”的结果则会显示出具体的错误信息。

例如，选中导入用例“测试用例.xml”中的最后三个测试“测试 9”“测试 10”“测试 11”，点击“运行用例”按钮，测试结果为：

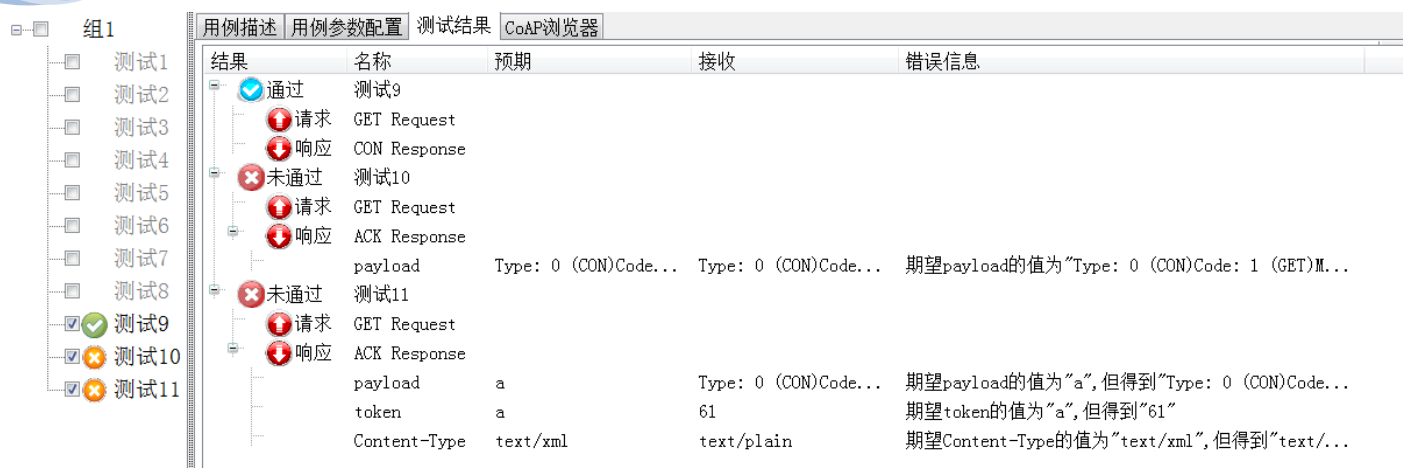


图 2-10

由上图可知，测试结果对于通过测试的用例运行结果不予以详细信息说明，只会显示出未通过测试的具体错误信息，并且当错误信息多于一处时，对各个错误信息并列显示具体错误信息。

2.4.2 导出测试用例结果

除了可以直接从软件中查看测试结果，还可以将测试结果导出查看。

例如，勾选“测试用例.xml”中的最后三个测试，运行后选择“测试用例结果”中的“导出用例测试结果”可将这三个测试的测试结果导出，在“另存为”对话框中输入要保存的文件名“测试用例结果 9-11”后点击“保存”（保存类型为 XML 文件）。

实际上在导出用例结果时，会同时自动生成 HTML 格式的测试结果，并且点击“保存”按钮后会自动打开（如图 2-11 所示），这种格式的测试用例结果更为直观明了。

其中，绿色背景的结果为“通过”，红色背景的结果为“未通过”，并且在表格最后会统计出成功通过测试和未通过测试的个数，同样对于通过测试的用例运行结果不予以详细信息说明，只会显示出未通过的具体错误信息，并且当错误信息多于一处时，对各个错误信息并列显示具体错误信息。

CoAP协议测试结果

2014/9/9 10:14:50				
测试用例	测试步骤/错误项名称	期望	接收	错误信息
测试9				
步骤	GET Request			
	CON Response			
测试10				
步骤	GET Request			
	ACK Response			
错误项	payload	Type: 0 (CON) Code: 1 (GET) MID: 1956 Token: 68656C6C6F	Type: 0 (CON) Code: 1 (GET) MID: 30172 Token: 68656C6C6F	期望payload的值为"Type: 0 (CON) Code: 1 (GET) MID: 1956 Token: 68656C6C6F",但得到"Type: 0 (CON) Code: 1 (GET) MID: 30172 Token: 68656C6C6F"
测试11				
步骤	GET Request			
	ACK Response			
错误项	payload	a	Type: 0 (CON) Code: 1 (GET) MID: 30173 Token: 61	期望payload的值为"a",但得到"Type: 0 (CON) Code: 1 (GET) MID: 30173 Token: 61"
	token	a	61	期望token的值为"a",但得到"61"
	Content-Type	text/xml	text/plain	期望Content-Type的值为"text/xml",但得到"text/plain"
成功1个, 失败2个				

图 2-11

2.5 详细示例

2.5.1 示例一

打开软件后, 点击“创建组”按钮, 新建“组 2”, 选中“组 2”再点击“创建用例”按钮, 新建用例“示例一”。在右侧选择“用例参数配置”标签页, 添加“GET Request”和“ACK Response”, 并分别按图 2-12 和图 2-13 进行参数配置, 保存。



图 2-12

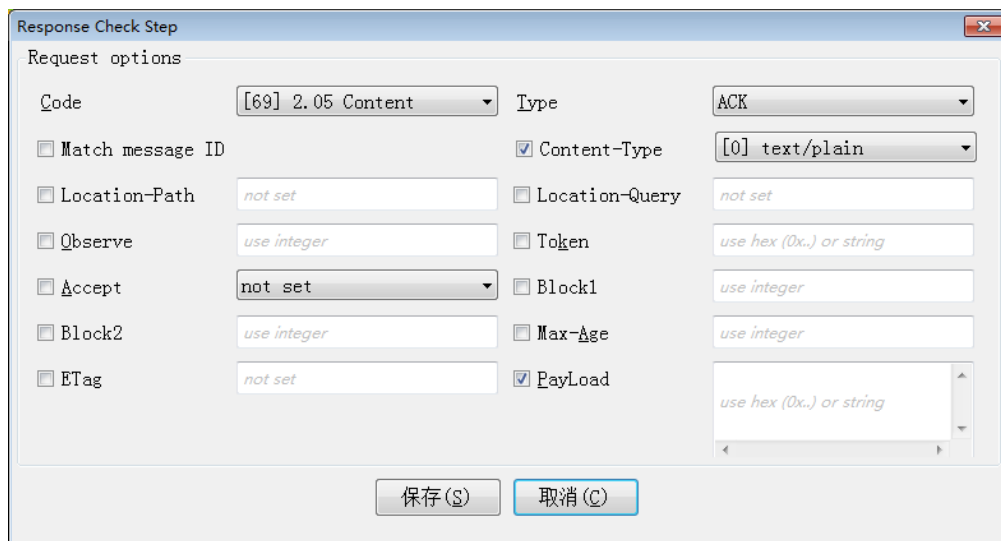


图 2-13

配置完成后，在左侧选中“示例一”，点击“运行用例”按钮，得到测试结果如图 2-14 所示。

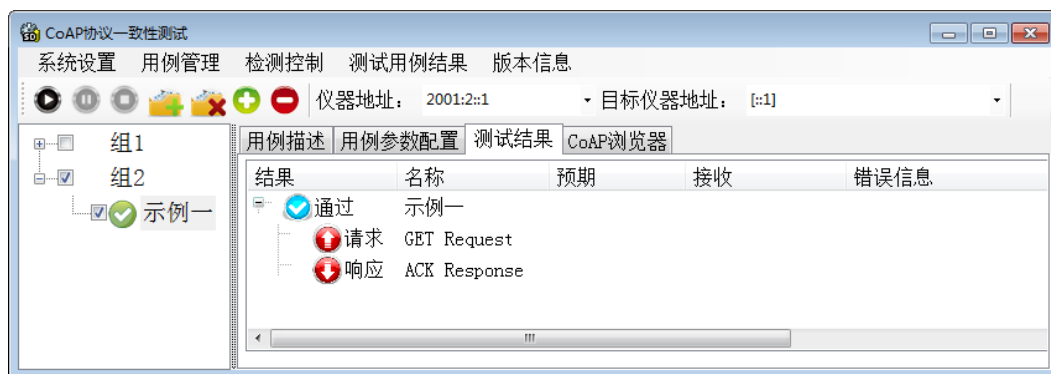
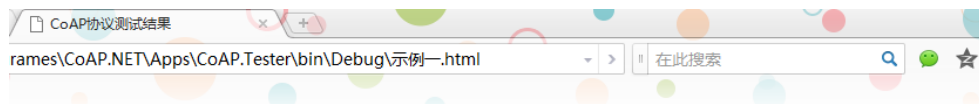


图 2-14

导出测试用例结果，得到 HTML 格式（图 2-15）的测试结果。



测试用例	测试步骤/错误项名称	期望	接收	错误信息
示例一				
步骤	GET Request			
	ACK Response			
成功1个，失败0个				

图 2-15

注：建议尝试修改部分参数配置，再次运行并查看结果。例如，在“Response Check Step”中勾选 Payload，并在后面输入框中填写“Hello World”，然后勾选 Content-Type 并在后方下拉子菜单中选择“[40]application/link-format”。运行用例后会发现测试结果未通过，因为预期得到的 Payload 和实际得到的 Payload 不一致，并且预期得到数据的 Content-Type 和实际得到的数据的 Content-Type 也不一致。

2.5.2 示例二

选中“组 2”后，点击创建用例按钮，新建用例“示例二”。在右侧选择“用例参数配置”标签页，添加“GET Request”和“ACK Response”，并分别按图 2-16 和图 2-17 进行参数配置，保存。



Request Build Step

Request options

Method	GET	Type	CON
Payload	use hex (0x..) or string	Token	use hex (0x..) or string
Uri-Path	/query	Uri-Query	?first=1&second=2&third=3
Accept	not set	Content-Type	not set
Block2	block number	Observe	use integer
Etag	use hex (0x..) or string	Uri-Host	not set
Uri-Port	n/s	Proxy-Uri	use absolute URI
Max-Age	use integer	If-Match	use an ETag

☐ If-None-Match ☐ Use Proxy-Scheme

保存(S) 取消(C)

图 2-16

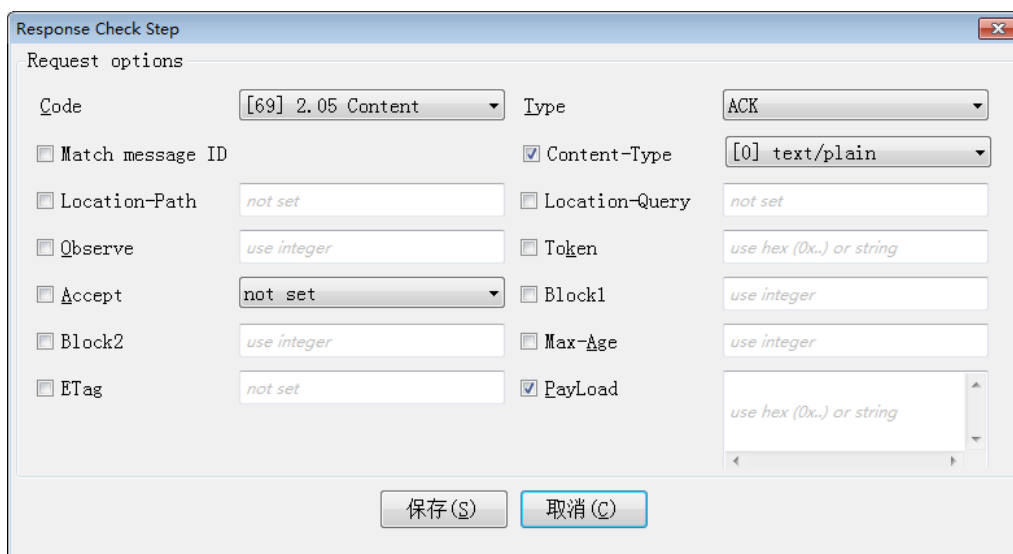


图 2-17

配置完成后，在左侧选中“示例二”，点击“运行用例”按钮，得到测试结果。

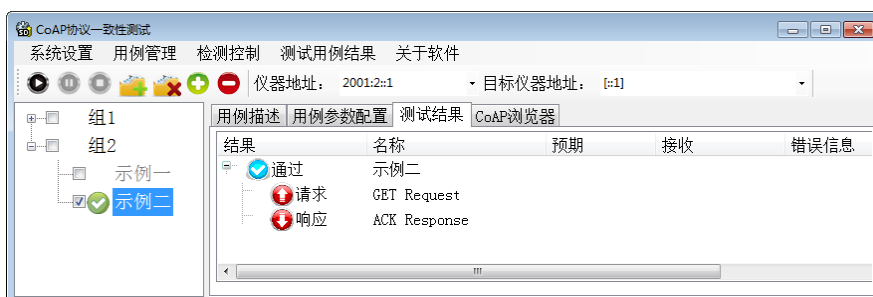
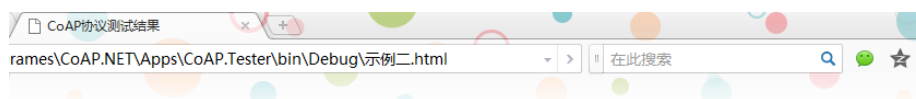


图 2-18

导出测试用例结果，得到 HTML 格式（如图 2-22 所示）的测试结果。



CoAP协议测试结果

2014/9/4 9:54:17				
测试用例	测试步骤/错误项名称	期望	接收	错误信息
示例二				
步骤	GET Request			
	ACK Response			
成功1个，失败0个				

图 2-22

注：建议尝试修改部分参数配置，再次运行并查看结果。例如，在“Response Check Step”中勾选 Payload，并在后面输入框中填写“Hello World”，然后勾选 Content-Type 并在后方下拉子菜单中

选择 “[40]application/link-format” 。运行用例后会发现测试结果未通过，因为预期得到的 Payload 和实际得到的 Payload 不一致，并且预期得到数据的 Content-Type 和实际得到的数据的 Content-Type 也不一致。

3 CoAP 浏览器

CoAP 浏览器可以有效支持 CoAP 协议，并且通过 CoAP 浏览器可以更加清楚和快速了解协议的测试过程及结果。

打开软件右下方标签页“CoAP 浏览器”（如图 3-1 所示），发现工具栏中有如下按钮

GET POST PUT DELETE，分别对应着“GET Request”、“POST Request”、“PUT Request”和“DELETE Request”。在这些按钮右边有 Observe Discover，点击“Discover”按钮可以获取 Resources（资源列表）。

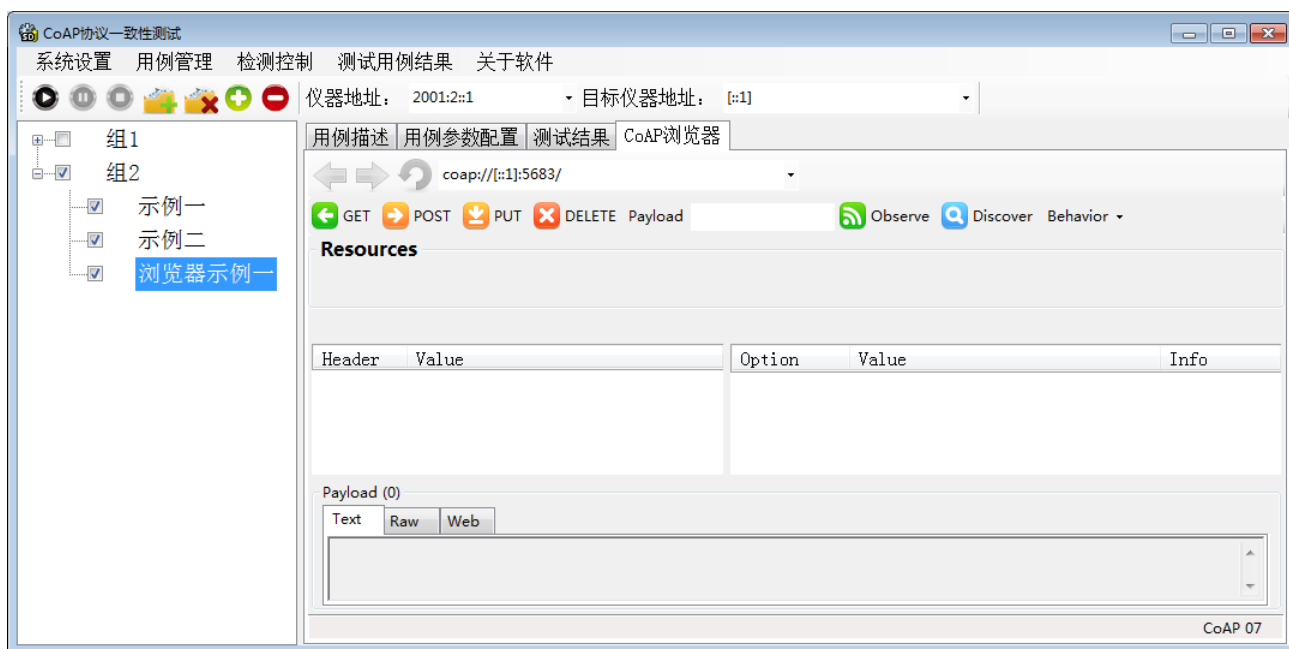


图 3-1

如图 3-2 所示，点击“Discover”按钮，获取资源列表（下图中红色方框内），然后便可以选择资源，并对各资源进行相关操作。

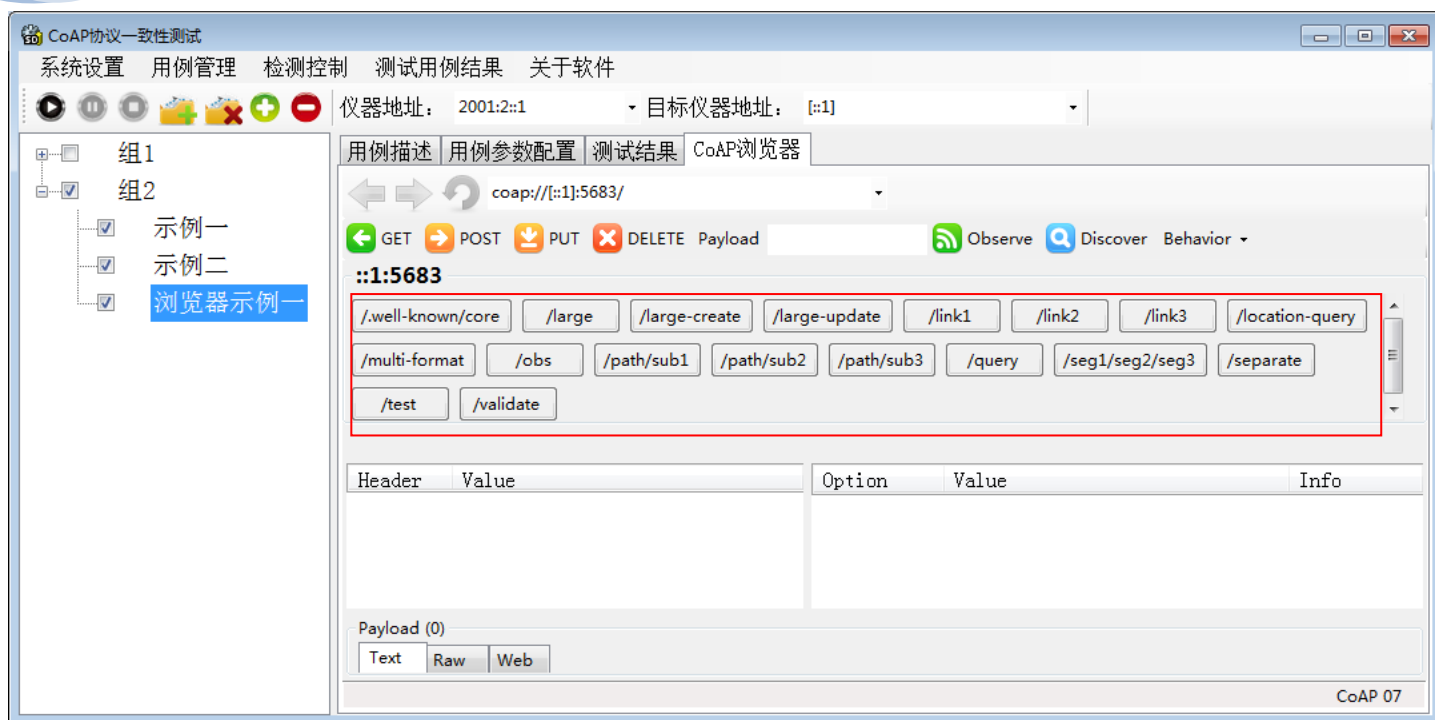


图 3-2

3.1 详细示例

3.1.1 示例一

选择资源/*large*，点击“GET”按钮，可得到如图 3-3 所示结果。

下方 Payload 即为资源/*large* 中的数据，本软件支持以 Text 格式，Raw 格式和 Web 格式查看数据。

注：可试着点击“POST”、“PUT”和“DELETE”按钮，得到结果为“4.05 Method Not Allowed”。

用例描述 用例参数配置 测试结果 CoAP浏览器

coap://[::1]:5683/large

GET POST PUT DELETE Payload Observe Discover Behavior

::1:5683

/well-known/core /large /large-create /large-update /link1 /link2 /link3 /location-query

/multi-format /obs /path/sub1 /path/sub2 /path/sub3 /query /seg1/seg2/seg3 /separate

/test /validate

2.05 Content

Header	Value	Option	Value	Info
Type	ACK	Content-Type	text/plain	0
Code	2.05 Content	Block2	2 (512B/block [5])	1 byte(s)
ID	1766			
Options	2			

Payload (1280 bytes)

Text Raw Web

```

[each line contains 64 bytes]
[each line contains 64 bytes]
[each line contains 64 bytes]
[each line contains 64 bytes]
[each line contains 64 bytes]
  
```

RESOURCE BLOCK NO. 1 OF 5
RESOURCE BLOCK NO. 2 OF 5
RESOURCE BLOCK NO. 3 OF 5
RESOURCE BLOCK NO. 4 OF 5
RESOURCE BLOCK NO. 5 OF 5

CoAP 07

图 3-3

若不采用 CoAP 浏览器，可按照如下操作方法完成 3.1.1 示例一。

添加用例“浏览器示例一”，在用例参数配置中添加“GET Request”和“ACK Response”，其中请求和响应的参数配置分别如图 3-4 和图 3-5 所示：



Request Build Step

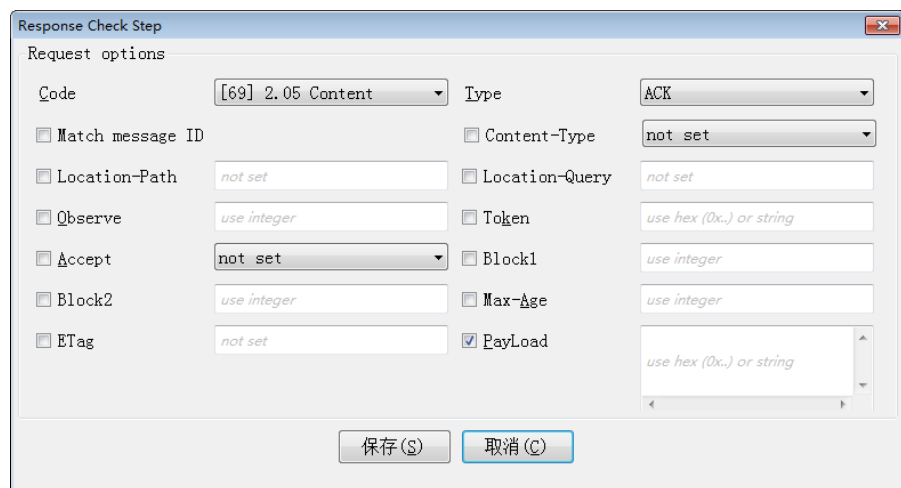
Request options

Method	GET	Type	CON
Payload	use hex (0x..) or string	Token	use hex (0x..) or string
Uri-Path	/large	Uri-Query	not set
Accept	not set	Content-Type	not set
Block2	block number	Observe	use integer
Etag	use hex (0x..) or string	Uri-Host	not set
Uri-Port	n/s	Proxy-Uri	use absolute URI
Max-Age	use integer	If-Match	use an ETag

☐ If-None-Match ☐ Use Proxy-Scheme

保存(S) 取消(C)

图 3-4



Response Check Step

Request options

Code	[69] 2.05 Content	Type	ACK
<input type="checkbox"/> Match message ID		<input type="checkbox"/> Content-Type	not set
<input type="checkbox"/> Location-Path	not set	<input type="checkbox"/> Location-Query	not set
<input type="checkbox"/> Observe	use integer	<input type="checkbox"/> Token	use hex (0x..) or string
<input type="checkbox"/> Accept	not set	<input type="checkbox"/> Block1	use integer
<input type="checkbox"/> Block2	use integer	<input type="checkbox"/> Max-Age	use integer
<input type="checkbox"/> ETag	not set	<input checked="" type="checkbox"/> Payload	use hex (0x..) or string

保存(S) 取消(C)

图 3-5

设置完成后，运行用例，通过。

注：如果从 CoAP 浏览器中将资源/large 获取的 Payload 复制下来，然后粘贴在“Response Check Step”中的 Payload 后面方框中，运行用例，可得到如图 3-6 和图 3-7 所示结果。

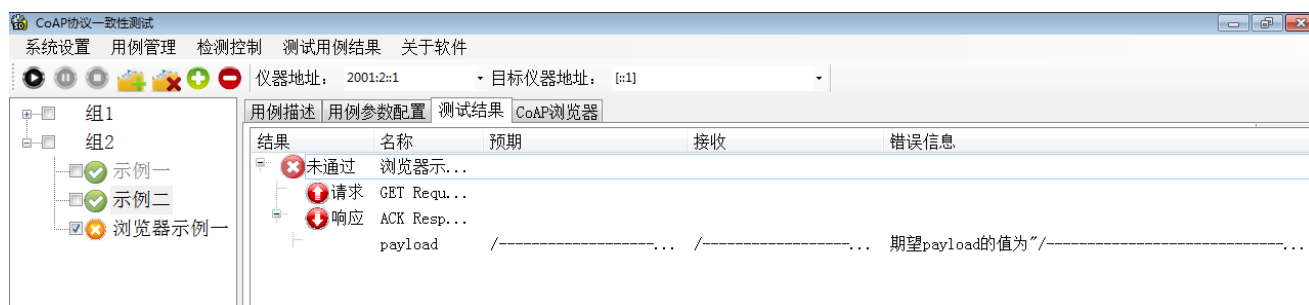


图 3-6

CoAP协议测试结果

2014/9/4 12:42:58

测试用例	期望	接收	错误信息
GET Request			
ACK Response			
payload	RESOURCE BLOCK NO. 1 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 2 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 3 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 4 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 5 OF 5 [each line contains 64 bytes]	RESOURCE BLOCK NO. 1 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 2 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 3 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 4 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 5 OF 5 [each line contains 64 bytes]	期望payload的值为"/ OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 2 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 3 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 4 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 5 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 1 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 2 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 3 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 4 OF 5 [each line contains 64 bytes] RESOURCE BLOCK NO. 5 OF 5 [each line contains 64 bytes]

成功0个，失败1个

图 3-7

从上图可看出，虽然预期的数据和实际收到的数据看起来一致，但实际上复制并粘贴之后的换行符会和接收到数据中的换行符不一致，而换行符不同会导致“未通过”结果。

3.1.2 示例二

选择资源/*large-create*，点击“POST”按钮，可得到如图 3-8 所示结果（2.01 Created）：

用例描述 用例参数配置 测试结果 CoAP浏览器

coap://[::1]:5683/

GET POST PUT DELETE Payload Observe Discover Behavior

::1:5683

/well-known/core /large /large-create /large-update /link1 /link2 /link3 /location-query

/multi-format /obs /path/sub1 /path/sub2 /path/sub3 /query /seg1/seg2/seg3 /separate

/test /validate

2.01 Created

Header	Value	Option	Value	Info
Type	ACK	Location-Path	nirvana	7 byte(s)
Code	2.01 Created			
ID	1854			
Options	1			

Payload (0 bytes)

Text Raw Web

CoAP 07

图 3-8

在上方 Payload 后面的方框内填写“smeshlink”，然后再点击“POST”按钮，得到结果同样为“2.01 Created”。然后点击“GET”按钮，可得到如图 3-9 所示结果。

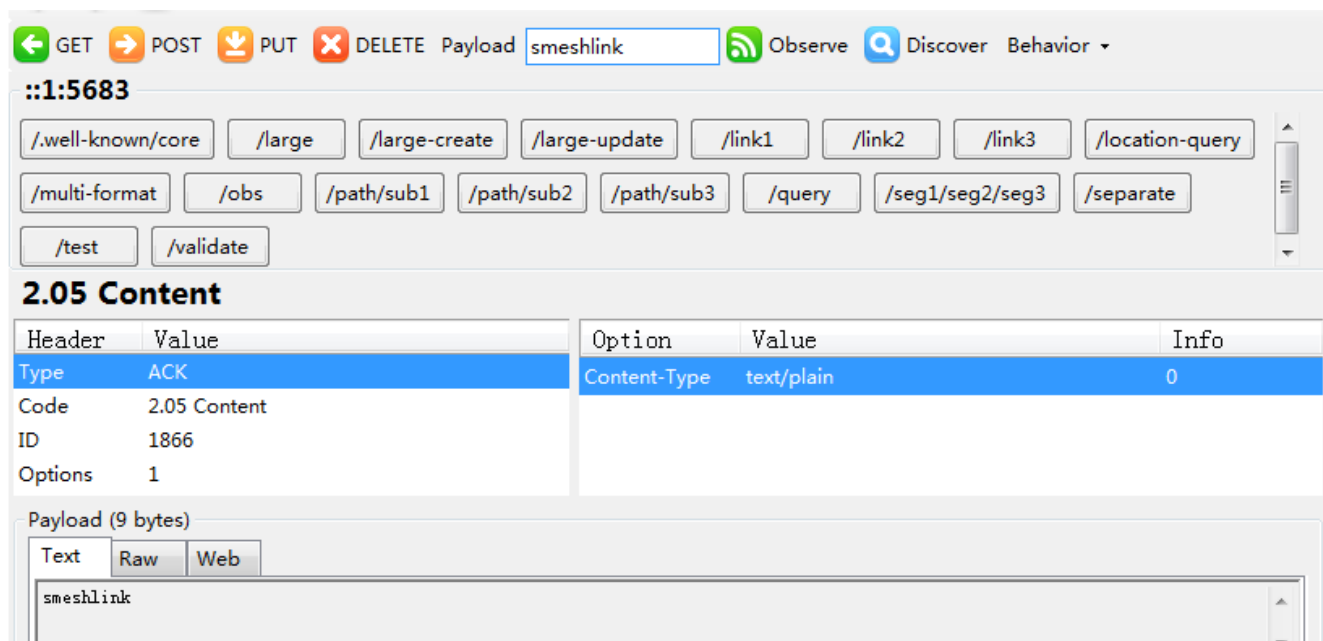


图 3-9

由上图可发现，可以通过“GET”获取刚刚利用“POST”创建的数据“smeshlink”，本软件支持以 Text 格式、Raw 格式、Web 格式查看数据。

注：可试着点击“PUT”按钮，得到结果为“4.05 Method Not Allowed”；点击“DELETE”按钮，得到结果为“2.02 DELETED”，然后再点击“GET”，下方 Payload 中会显示“Nothing POSTed yet”。

若不采用 CoAP 浏览器，可按照如下操作方法完成 3.1.2 示例二。

添加用例“浏览器示例二”，在用例参数配置中添加“GET Request”和“ACK Response”，其中请求和响应的参数配置分别如图 3-10 和图 3-11 所示。



Request Build Step

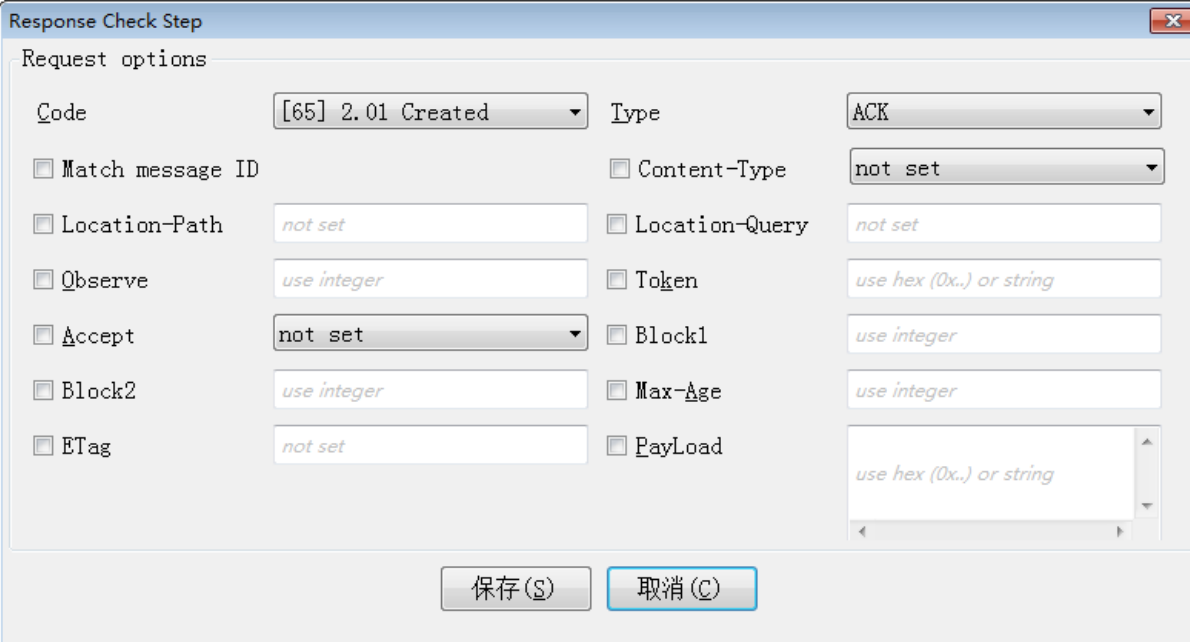
Request options

Method	POST	Type	CON
Payload	smeshlink	Token	use hex (0x..) or string
Uri-Path	/large-create	Uri-Query	not set
Accept	not set	Content-Type	[0] text/plain
Block2	block number	Observe	use integer
ETag	use hex (0x..) or string	Uri-Host	not set
Uri-Port	n/s	Proxy-Uri	use absolute URI
Max-Age	use integer	If-Match	use an ETag

☐ If-None-Match ☐ Use Proxy-Scheme

保存(S) 取消(C)

图 3-10



Response Check Step

Request options

Code	[65] 2.01 Created	Type	ACK
<input type="checkbox"/> Match message ID		<input type="checkbox"/> Content-Type	not set
<input type="checkbox"/> Location-Path	not set	<input type="checkbox"/> Location-Query	not set
<input type="checkbox"/> Observe	use integer	<input type="checkbox"/> Token	use hex (0x..) or string
<input type="checkbox"/> Accept	not set	<input type="checkbox"/> Block1	use integer
<input type="checkbox"/> Block2	use integer	<input type="checkbox"/> Max-Age	use integer
<input type="checkbox"/> ETag	not set	<input type="checkbox"/> Payload	use hex (0x..) or string

保存(S) 取消(C)

图 3-11

设置完成后，运行用例，得到如图 3-12 和图 3-13 所示的结果。

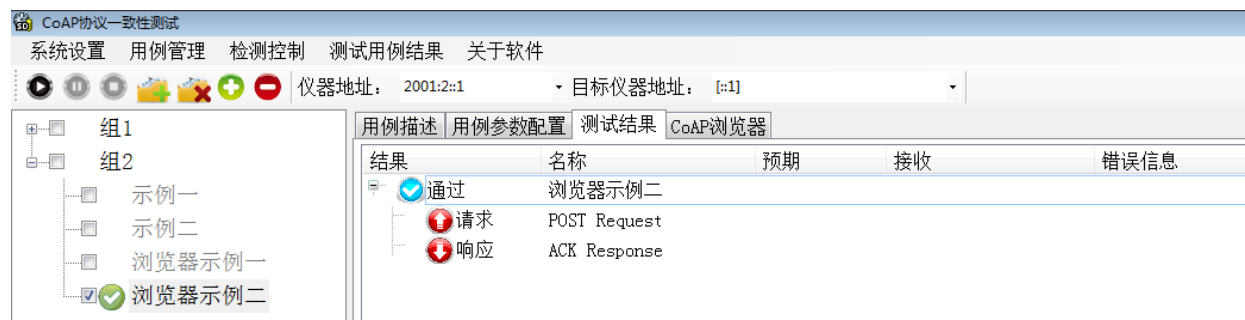
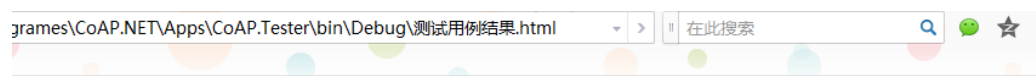


图 3-12



CoAP协议测试结果

2014/9/5 15:41:24				
测试用例	测试步骤错误项名称	期望	接收	错误信息
浏览器示例二				
步骤	POST Request			
	ACK Response			
成功1个，失败0个				

图 3-13

4 附录

在附录中，用户可以发现更多参考用例，以下参考用例皆可通过协议一致性测试。

名称	CoAP 协议测试一	
简要描述	了解 PUT 请求	
配置	CoAP_CFG_01	
测试条件	<ul style="list-style-type: none"> Client supports Block transfers Server supports Block transfers Server offers a large updatable resource /large-update 	
测试序列	请求	Method: PUT Type: CON Payload: smeshlink Uri-Path: /large-update Content-Type: [0]text/plain
	响应	Code: [68]2.04 Changed Type: ACK

名称	CoAP 协议测试二	
简要描述	接入 well-known 接口发现资源	
配置	CoAP_CFG_01	
测试条件	<ul style="list-style-type: none"> Client supports CoRE Link Format Server supports /.well-known/core resource and the CoRE Link Format 	
测试序列	请求	Method: GET

		Type: CON Uri-Path: /.well-known/core
	响应	Code: [69]2.05 Content Type: ACK Content-Type: [40]application/link-format

名称	CoAP 协议测试三	
简要描述	过滤请求访问有限资源	
配置	CoAP_CFG_01	
测试条件	<ul style="list-style-type: none"> Client supports CoRE Link Format Server supports CoRE Link Format Server offers different types of resources (Type1, Type2, ...; see Note) 	
测试序列	请求	Method: GET Type: CON Uri-Path: /.well-known/core Uri-Query: rt=Type1
	响应	Code: [69]2.05 Content Type: ACK Content-Type: [40]application/link-format 勾选 Payload