**Передача данных, используя Web Services в BW**

[02.02.2018](https://helpbw.wordpress.com/2018/02/02/%d0%bf%d0%b5%d1%80%d0%b5%d0%b4%d0%b0%d1%87%d0%b0-%d0%b4%d0%b0%d0%bd%d0%bd%d1%8b%d1%85-%d0%b8%d1%81%d0%bf%d0%be%d0%bb%d1%8c%d0%b7%d1%83%d1%8f-web-services-%d0%b2-bw/) ~ [Vyacheslav](https://helpbw.wordpress.com/author/hazysonic/)

Загрузка данных в BW, используя Web-сервис.

Статьи по использованию Web Serivces есть на help.sap.com, на английском языке.

[Creating InfoPackages for Real-Time Data Acquisition](https://help.sap.com/saphelp_nw74/helpdata/en/4a/2c6be7c16d47dbe10000000a42189c/content.htm)

[Data Transfer Using Web Services](https://help.sap.com/saphelp_nw74/helpdata/en/4a/1dfbbe5c171b40e10000000a42189c/content.htm)

[Transferring Data from SLT Using Web Service](https://help.sap.com/saphelp_nw74/helpdata/en/c5/80b8ba211c4d5993adf04eb2070b4d/content.htm)

[Transferring Transaction Data Using Web Services](https://help.sap.com/saphelp_nw74/helpdata/en/4a/2c6d68c16d47dbe10000000a42189c/frameset.htm)

[How to configure a Service Provider](https://wiki.scn.sap.com/wiki/display/ABAPConn/How+to+configure+a+Service+Provider)

[Настройка Web сервиса между системами SAP, на англ. раз.](https://wiki.scn.sap.com/wiki/display/ABAP/CREATING+OF+WEBSERVICES+AND+CONSUMING+IN+SAP)

[Пошаговое руководство по созданию Web сервиса, на англ. два.](https://blogs.sap.com/2014/05/20/step-by-step-to-create-consume-and-trace-web-service-in-abap-system/)

**Передача данных**

**Использование**

Данные в BW передаются через запрос данных из исходной системы. Запрос посылается из BW (запланирован в планировщике).  Так же возможно использовать Web Services, если вы хотите, что бы загрузка данных контролировалась из внешней системы. В таком случае данные пушатся/проталкиваются (push)

При использовании Web сервисов для передачи данных в BW, возможно использовать экстракцию данных в режиме реального времени. Альтернативный вариант — использовать стандартный DTP.

1. Если данные нужны часто, то используется сбор данных в режиме реального времени.  В первую очередь данных записываются в BW в PSA. Далее с помощью демона или фонового процесса через определённые промежутки времени данные записываются в DSO и становятся доступными для системы отчётов.
2. Если нет необходимости в обновлении данных через определённые короткие интервалы времени, то используется стандартное обновление.

# REST Service Integration in SAP BW

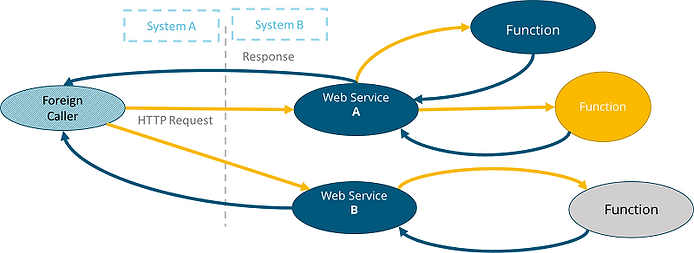
## REST Service Integration in SAP BW

Integration is one of the most important keywords in today’s Information Technology systems. The ability to connect outside of its boundaries and collect data that would be otherwise effortful or not possible to create is one of the utmost important features to an application, with a great number of implications. As web applications gain popularity, so is the need for their integration. **RE**presentational **S**tate **T**ransfer, also known as REST, is a wide-spread Architectural Standard for the creation of Web Services (also encountered as Web APIs). From Social Media to Banking, most web applications/sites we use in our personal and work life rely on Web Services. REST Services, in particular, are employed by the products of a number of today’s software giants. We, the BI consulting company - DAHLBEER, strongly believe in the benefits of automating those user actions that machines can perform better/faster as well as decreasing the number of applications & screens the users are exposed to – two of the numerous use cases for a REST service interface. In today’s article we’ll take a closer look on REST Services and how can we communicate with them using SAP BW.

## Web Service Basics

Web services are a way for a web application to interact with the outside world and send and receive data. Let’s imagine the following case: A Company needs to calculate the amount of labor spent by its employees on different production processes for the sake of a price planning application in its SAP BW system. The Company has already invested in a third-party time tracking web application for the needs of its HR department and the tool has a built-in Service that can provide data to external applications when requested. The BW system can **Call** the **Service** and **Request** the total time spent on a particular activity **(parameter I)** for a given period of time **(parameter II)**. The Service in its place would provide a Response to the caller. This approach is much more efficient than building mirror time tracking solution and asking the users to submit their timesheets in two different systems.





## Rest Specifics

REST is a particular way of building a web service. RESTful services are considered easier to operate with, they are based on Hypertext Transfer Protocol (HTTP) and have similar constructs and same status codes. REST is language independent, meaning that any language that can execute HTTP requests can be used.



The most important elements of the request are:

* URI ( Universal Uniform Identificator = URL + URN): Shows the location and name of the service
* Method: Shows what action should be accomplished with the data. Most widely used are POST and GET methods. Use POST if the use case is to send data to the service and GET to receive data from the service
* Headers: Fields that help in establishing communication between the parties. Passed in “Key” : “Value” format. Example: “Accept: application/json”
* Parameters: Values that show the service details about the requested data or help to determine what internal function needs to be executed.
* Authorization Elements (optional): Contain a form of user authentication. Different types exist, but the most basic form is just supplying username and password.
* Body: Has no meaning for a GET request, however, in case of POST should contain the data to be sent.

Each request is followed by a response that has headers and body of its own, as well as status, In case of a GET request, the requested data would be found in the Body of the response. Example: *“Accept: application/json”*

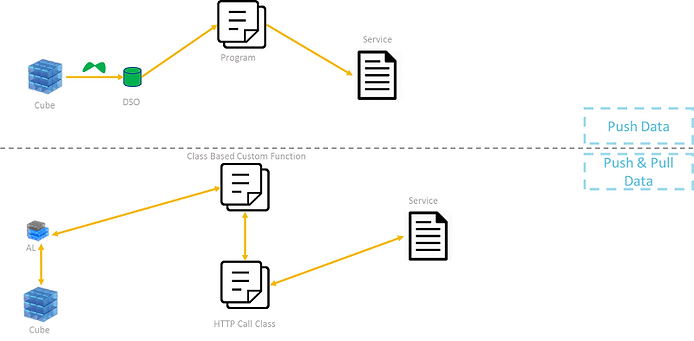
*{* *"ActivityCode": "11",* *"EmployeeCode": "203192",* *"DepartmentCode": "M14",* *"TimeSpent": "15d",* *}*

## Calling a REST Service from within BW

Let’s review the two most common ways to call service via HTTP in BW. One of them is Class-Based and the second one is Functional Module Based. The latter employs standard modules “HTTP\_GET” & “HTTP\_POST” the import and export parameters of which contain the basic elements of the request. Complex calls, however, would require the use of the built-in http classes & interfaces in BW. Here are the basic steps that have to be followed:

* Declare **URI** object referencing interface *if\_http\_client* – Will be used to building up the URI & Authorization Elements
* Declare **Request & Response** objects referencing interface *if\_rest\_entity* – Will be used to building up the request
* Declare **Call** Object referencing class *cl\_rest\_http\_client* - Will be used to executing the request
* Call Method *cl\_http\_client=>create\_by\_url* to Create the **URI** object
* Call Method *lo\_http\_client->request->set\_version* to Set HTTP Version for **URI** Object
* Call Method *lo\_http\_client->authenticate* to Authenticate **URI** Object (optional)
* Call Method *cl\_http\_utility=>set\_request\_uri* to set URI in **URI** Object
* Create Call Object by using CREATE OBJECT statement & exporting the **URI** Object
* Create **Request** object for Call Object using ***[Request Obj.]*** *=* ***[Call Obj.]*** *->if\_rest\_client~create\_request\_entity( )*
* Set Header fields for **Request** Object using ***[Request Obj.]*** *->set\_header\_field( iv\_name = 'Accept' iv\_value = 'application/json' )*
* Set Body for **Request** Object using ***[Request Obj.]****->set\_string\_data( lv\_body\_string)* (optional)
* Execute HTTP Method: ***[Call Obj.]*** ->*if\_rest\_resource~get( Request obj. )*
* Collect Response. ***[Response obj.]*** *=* ***[Call Obj.]*** *->if\_rest\_client~get\_response\_entity( )*

## Architecture & Automation



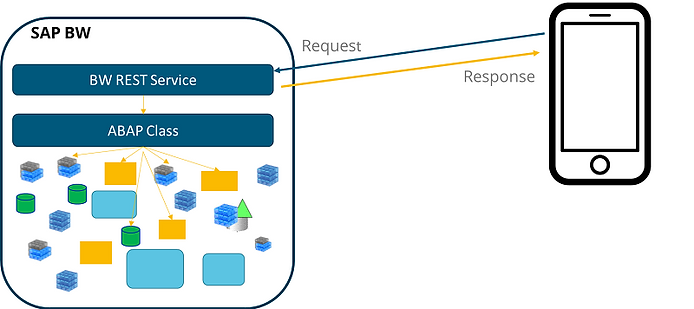
In a productive application, service calls would be rather frequent and they would need a proper mechanism to in order to connect to the application landscape. Among many architecture possibilities, here we show a couple that represents a simple, and a complex planning scenario.

In the first one, the sole goal of the REST service interface is to supply plan data to a subsequent planning step inside a web application via scheduled data pushes.

Data originates from a real-time InfoCube and is loaded in a write-optimized DSO (with the relevant transformations), which has the same granularity as the target on the Web application side. A program reads the DSO into an internal table and then proceeds to execute an HTTP POST to the desired service. A process chain that includes both the data load to the DSO and the program is needed for automation.

In the second approach, data is selected from the Cube via a custom planning function and transformations are executed via ABAP Code. Afterwards, a dedicated helper class takes care of calling the REST Service. Due to the use of a planning function, the reverse action is also possible: User executes a planning sequence in order to get real-time data from a remote system. In essence, this architecture allows both data push and data pull from BW perspective.

## Creating a Native BW REST service



In order to enable initiation of a Push/Pull on the remote side as well, BW has a Web Service Creation as well as Web Request Handling Capabilities Integrated, which make it possible to implement a REST service straight in BW. This way, the system would be open for calls from external web applications requesting data on the fly. This would enable web applications which cannot store large sets to still display relevant user selected data or even do planning. Transaction SICF is responsible for the creation of the Service entity. A custom ABAP class, however, has to be created to read the call and extract/write data from respective InfoProviders. Among other exciting use cases, such an implementation would make planning and reporting easily possible from mobile devices.

In Conclusion, the wide adoption of Rest APIs in combination with matching BW features provides us with an easy gateway option to most modern web applications in order to gather and distribute information, outsource complex calculations, derive pictures and link planning applications to various web services providing valuable input.

## SAP Web Service Введение и метод конфигурации

<https://russianblogs.com/article/86732233208/>

# Consuming Webservice in SAP ABAP

<https://www.samplecodeabap.com/consuming-webservice-sap-abap/>