Measuring WMB from inside is possible thanks to the availability of the new SMF 117 record which provides detailed information about the work performed in a WMB environment.

SMF data collection is not active by default. You have to request it by issuing the following `modify` command:

```plaintext
f broker,cs a=yes,g=yes,j=yes,c=active,t=basic,n=basic,o=smf
```

Figure 6

The `cs` command stands for `changeflowstats` while all the following parameters are the collection options; they have the following meaning:

- **a = yes**: collects archive statistics (snapshot statistics are also available);
- **g = yes**: selects all the execution groups;
- **j = yes**: selects all the message flows;
- **c = active**: activates data collection (use inactive to deactivate it);
- **t = basic**: activates threads data collection; possible values are none and basic;
- **n = basic**: activates basic nodes data collection; possible values are none, basic and advanced; advanced will produce terminal information
- **o=smf**: writes the collected information to SMF.

Other options are available to collect data only from a specific execution group or message flow; more details can be found in the IBM Websphere Message Broker Information Center web site.

By default archive statistics\(^1\) are written at 1 hour interval but you can synchronize them with SMF and RMF interval by using the `changebroker` command with the `v=0` parameter.

```plaintext
f broker,cb v=0
```

Figure 7

---

\(^1\) Snapshot statistics are written every 20 seconds so they are normally not used on a regular basis for performance analysis and capacity planning.
The `rs(mqisireportflowstats)` command displays the current options for accounting and statistics.

```
-F BROKER,RS A=YES,G=YES,J=YES
+BIP8071I MBB1BRK 2 Successful command completion.
```

**Figure 8**

In the above example the status of archive accounting collection has been requested for all the message flows and execution groups.

A separate BIP8187I message is provided, for each message flow, including all the information. The same command shown above, but using the S=YES instead of the A=YES parameter, can be used to retrieve information about the status of the snapshot statistics.

### 6 Most important metrics provided in SMF 117

Two subtypes of SMF 117 records will be written depending on the data currently being collected:

- A single subtype 1 record is produced when only message flow or threads data is collected;
- A single subtype 2 record is produced when only node data is collected; if nodes and terminals\(^2\) data is collected multiple subtype 2 records will be produced.

In the following chapters only message flows and nodes information will be discussed.

#### 6.1 SMF 117 subtype 1 – Message flows

The following metrics allow the identification of a specific message flow and focus on its activity in a specific interval:

- **IMFLBKNM**, broker name;
- **IMFLEXNM**, execution group name;
- **IMFLMFNM**, message flow name;
- **IMFLSTDT**, interval start date;
- **IMFLSTTM**, interval start time;
- **IMFLENDT**, interval end date;
- **IMFLENTM**, interval end time.

\(^2\) A terminal is the point at which one node in a message flow is connected to another node. Terminals have to be used to control the route that a message takes as much as to manage errors in the message flow.
The first information you normally want to know is about throughput: how many messages and how many bytes have been served by the message flow in that interval?
The following metrics provide that information:
- **IMFLTPMG**, total number of messages processed;
- **IMFLTSMG**, total size of input messages (bytes);
- **IMFLMXMG**, maximum input message size (bytes);
- **IMFLMNMG**, minimum input message size (bytes).

The second information you want to know is about performance: how much time it has been needed to process all this data?
The following metrics provide that information:
- **IMFLTPTM**, total elapsed time spent processing input messages;
- **IMFLWTIN**, total elapsed time spent waiting for input messages;
- **IMFLMXTM**, minimum elapsed time spent processing an input message;
- **IMFLMNTM**, maximum elapsed time spent processing an input message.

Obviously you want also to know how much CPU has been used.
The following metrics provide that information:
- **IMFLTPCP**, total processor time spent processing input messages;
- **IMFLWTCP**, total processor time spent waiting for input messages;
- **IMFLMXCP**, maximum processor time spent processing an input message;
- **IMFLMNCP**, minimum processor time spent processing an input message.

It’s important to note that, as we will see in Chapter 7, “polling” techniques are used in the broker environment. IMFLWTCP can be used to evaluate the overhead of this polling activity.

The following additional metrics are provided; they could be very useful to analyse application behaviour and eventual anomalies:
- **IMFLTHDP**, number of threads in pool;
- **IMFLTHDM**, number of times the maximum number of threads is reached;
- **IMFLERMQ**, number of MQGET errors (MQInput node) or Web services errors (HTTPInput node);
- **IMFLERMG**, number of messages that contain errors;
- **IMFLERPR**, number of errors processing a message;
- **IMFLTMOU**, number of timeouts processing a message;
- **IMFLCMIT**, number of transaction commits;
- **IMFLBKOU**, number of transaction backouts.

Finally the **IMFLACCT** (accounting origin) metric is provided. It can be exploited to perform more detailed accounting activities. You have to configure message flows to specify a particular accounting origin.
6.2 SMF 117 subtype 2 – Nodes

The following metrics allow you to identify a specific node and focus on its activity in a specific interval:

- IMFLBKNM, broker name;
- IMFLEXNM, execution group name;
- IMFLMFNM, message flow name;
- INODNDNM, name of node (Label);
- INODTYPE, type of node;
- IMFLSTD, interval start date;
- IMFLSTTM, interval start time;
- IMFLENDT, interval end date;
- IMFLENTM, interval end time.

A reduced set of metrics, compared to message flows, is provided to analyse throughput, performance and CPU usage for each node:

- INODTPMG, total number of messages processed by this node;
- INODTPTM, total elapsed time spent processing input messages;
- INODMXTM, maximum elapsed time spent processing input messages;
- INODMNMTM, minimum elapsed time spent processing input messages;
- INODTPCP, total processor time spent processing input messages;
- INODMXCP, maximum processor time spent processing input messages;
- INODMNCP, minimum processor time spent processing input messages.
7 Using SMF measurements – A real life experience

In this chapter we will show how we used the SMF records to analyze the performance of a simple application prototype.
The application under analysis consists of three message flows each running in a dedicated execution group used to get, parse, validate and save XML messages. The input is composed of single-messages and three types of multiple-message files including 100, 1.000 or 4.000 messages. All the messages are in XML format and their size is about 4K.

In this test the input was: 2945 single-messages, 15 files including 100 messages, 3 files including 1.000 messages and 1 file including 4.000 messages. So the total number of single-messages included in files is 8.500.
The duration of the test was about 600 seconds.

In Figure 9 the usage of CPU and zIIP taken from SMF 30 records is reported.
The first thing to note is that only 2.2% of this work runs on zIIP\(^3\) (this behavior really surprised us especially because we were using an IBM tool to process XML messages).
You can also see how much resources are used by each execution group (identified by the PROCSTEP values starting with EXG) and by the broker control address space (identified by STARTING).

The next figure has been created by using SMF 117; it provides more detailed information at the message flow level.

If you look at the number of message processed (IMFLTPMG) by each message flow (IMFLMFNM) you can understand something about the application design.
All single-messages and all the multiple-message files go through the INP_Sorter message flow. The total number of messages is 2.964 (the sum of 2.945 single-messages and 19 multiple-message files).
The other message flows only process multiple-message files so their IMFLTPMG is only 19.
You can also note that the minimum (IMFLMNNMG) is 4K for the INP_Sorter message flow because it also processes single-messages while it is 431K (about 100 * 4K) for the other message flows which only process multiple-message files.

\(^3\) This customer uses the zAAP on zIIP option.
The maximum (IMFLMXMG) message size is 16MB (about 4,000 * 4K) for all the message flows. The IMFLTPCP column provides the CPU seconds used when processing messages while IMFLWTCP provides the CPU seconds spent in “polling”. In this case the “polling” overhead could be estimated at about 2,5% of the total CPU time used.

The total CPU used seconds is very close to what we got from SMF 30 so it seems the CPU accounting in SMF 117 is very good.

The total elapsed time processing messages (IMFLTPTM) is much lower than the interval duration so it seems there is the possibility to sustain a much bigger throughput.

![Table Image](image)

The above figure highlights some errors in the INP_Sorter message flow probably due to message content or format (IMFLERMG); the number of commits (IMFLCMIT) is the sum of the total single-messages and multiple-message files (2,964) plus the errors (72). This issue is still under investigation.

![Diagram Image](image)

Finally in Figure 12 we present a detailed analysis of the nodes included in the INP_FileSplitter message flow.

You can see that all the nodes are provided in the SMF records even if they are not used. The used nodes have a coloured background: the light yellow background nodes take the multiple-message files from an input queue and split them in single-messages; the light green background nodes process single messages and put them to the output queue.

The total CPU seconds used by all the nodes matches the CPU seconds measured at the message flow level (see Figure 10).
8 Conclusions

The IBM WebSphere Message Broker (WMB) can run on any platform. If for any reason you decide to run it in z/OS you can take advantage of the standard z/OS workload management services and measurements provided by WLM and SMF. A new SMF record type (117) can be written by properly customizing the WMB environment; it provides a lot of additional detailed information about message flows, threads, nodes and terminals. As shown in this paper, making this information available is an essential step in order to analyse and tune application performance and to even get an insight into the application design.