Automotive Industry

Unified Diagnostic Services

Introduction

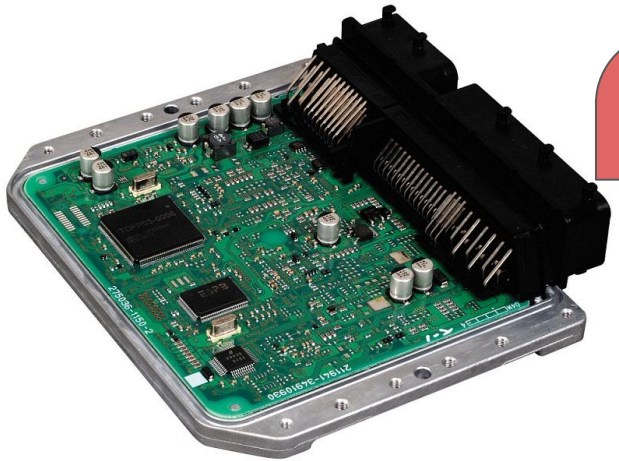
Over time, cars have become more modern and digital. Therefore more digital errors can occur, which can affect the condition and value of the car and in the worst-case scenario, a dangerous accident can occur.

In order to be able to quickly and accurately determine the failure of the vehicle or an element that causes the failure, each ECU has been supplied by a software that can detect and report these failures.

Over voltage, high temperature, short and open circuit conditions, fan lock, unexpected reset and many other failures can be detected by the ECU itself. Once the ECU detect a failure, it would save it in a fault memory for later usage. This fault memory is non volatile memory, to keep this data existing until being removed intentionally after fixing of the error

Diagnostics

Diagnostics is the process of inspection and acquiring these failures. The diagnostic doesn't include the fix, it just gives information about failures detected. Identifying the issue helps us to find the right solution. The diagnostic could be classified into two types, **On Board Diagnostic** and **Off Board Diagnostic**.



Report

Failure Memory

Failure 1
Failure 2
Failure 3
Failure 4
Failure 5

On Board Diagnostic

On-board vehicle diagnostics comes into the picture when the vehicle is moving. The tests are being conducted while the vehicle is on the road. The test results can be seen on the vehicle's dashboard in the form of **MIL (Malfunction indicator light).**

Additionally, a **limp home mode** may be activated in some high-end cars. This mode activates an algorithm to let you drive home or to the service garage without causing more damage to the vehicle.

The error that triggers the MIL is also stored in the automotive ECU which can later be retrieved by the tester tool at the garage.

Off Board Diagnostic

Off-board vehicle diagnostics is normally done using a diagnostic tool that is connected to the car, normally in a maintenance center.

A tester tool is connected to a network of ECUs via a communication channel which can may be CAN, LIN or any other automotive protocol.

Apart from this, there are service protocols which focus on how ECUs and tester should communicate, protocols are released specially for diagnostics, like UDS and KWP.

The off board diagnostic is used for:

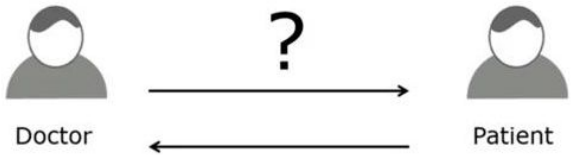
- Reading / Writing vehicle identification data.
- ECU's fault memory check.
- End-of-line programming.

Data Trouble Code

A DTC, short for Diagnostic Trouble Code, is a code used to diagnose malfunctions in a vehicle or heavy equipment. While the malfunction indicator lamp (MIL) simply alerts drivers that there is an issue, a DTC identifies what and where the issue is. DTCs can be read with a scanner that plugs directly into the port of a vehicle.

Through this fault, it can reflect the specific location of the ECU and the type of fault generated. The DTC format is defined in many international standards. For example, UTC defines DTC to be composed of 3 bytes, and ISO 15031-6 defines DTC format to be composed of "two-byte root + one-byte failure type". So what kind of DTC format is used in our commonly used UDS services? 95% of the DTC formats used are DTC failure types and formats defined in ISO 15031-6

Diagnostic Protocol



The concept of diagnostics comes from medicine, doctors use data to make judgments about illnesses by asking and observing patients, or by instrument testing. The purpose of vehicle diagnosis is similar, in order to be able to quickly and accurately determine the failure of the vehicle or a controller and the cause of the failure, so as to provide a reliable basis for maintenance.

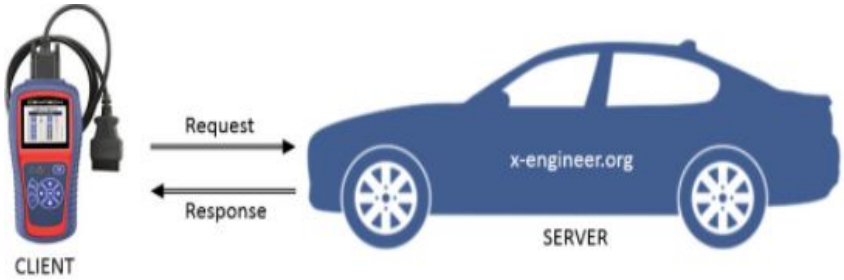
The tester performs the diagnostics on the ECUs by a service request, and these services are performed by the ECU. Since the tester request from the service, tester is considered as a client, since ECU provides the service, ECU is considered as a server

The Tester and ECU communicate with each other in the form of question and answer, therefore, the Tester and ECU must follow the same diagnostic communication protocol.

Commonly used diagnostic protocols are ISO 14230, ISO 15031, ISO 15765, and the familiar ISO 14229 is the UDS protocol, which defines the diagnostic request, the format of the diagnostic response message, and how the ECU handles the diagnostic request message and the application of the diagnostic service.

Unified Diagnostic Services

UDS is a diagnostic communication protocol in the ECUs environment within the automotive electronics which is specified in the ISO 14229-1. Unified in this context means it is an international and not a company specific standard.



The tester performs the diagnostics on the ECUs by a service request, and these services are performed by the ECU. Since the tester request from the service, tester is considered as a client, since ECU provides the service, ECU is considered as a server. The Tester and ECU communicate with each other in the form of question and answer, therefore, the Tester and ECU must follow the same diagnostic communication protocol.

Client requests the server for the service by sending the request message, the server processes the request and sends a response message back to the client. These request messages and response messages are sent between the client and the server by CAN frames or FLEXRAY frames depending on underline communication channel.

Unified Diagnostic Services

The UDS is an application protocol, it uses one of the standard communication protocols as physical connection between server and client. Communication protocols such as CAN, LIN, Ethernet, Flexray, … etc can be used. If the tester and ECU are connected via a CAN channel, then a single frame can carry up to 8 bytes of data, If the tester and ECU are connected via a FLEXRAY channel, then a single frame can carry up to 256 bytes of data.

BUT if the request message or the response message is greater than the data limit, more than 8 bytes in case of CAN or more then 256 bytes in case of FLEXRAY, then a single frame will not be sufficient to carry the entire message and multi frames are needed.

Tester can trigger various actions in the ECU , Request for data, Writing some data, Running tests and getting results back, Flashing a program in the ECU, Clearing the memory, Setting the scheduler And many other actions.

UDS defines what are available services, what is the format of the requested message, what is the format of the response message, what must be timing parameters, and how should services be handled by tester and ECU etc.

UDS Frame Format

At first, remember that the request message is sent by the Tester (client), to the ECU (server), and the UDS is a collection of diagnostic services. Each service has a service identifier assigned to it, this service identifier is called (SID) and is of 1 Byte length and has a range of (0x00 : 0x3E)

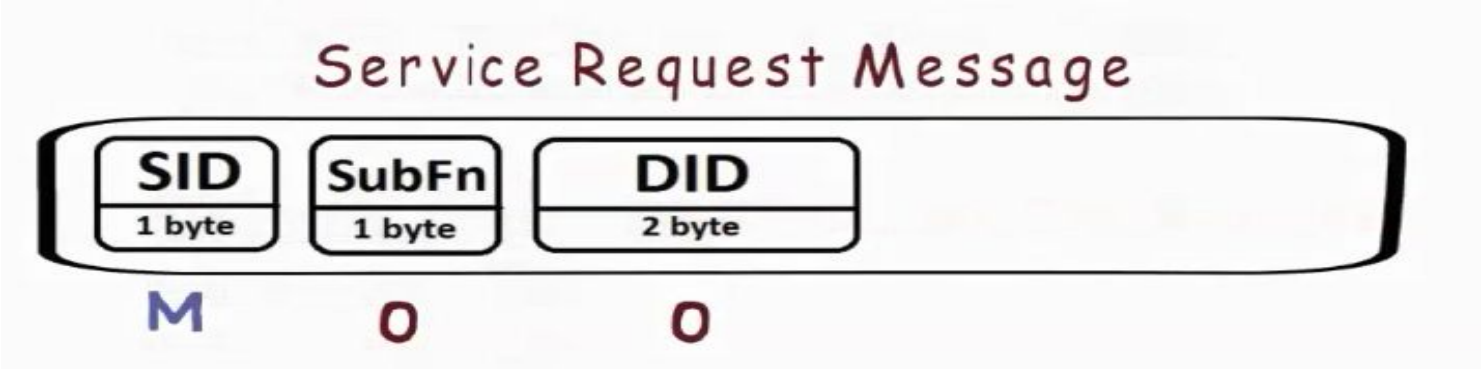
With this SID, the server understands which service is requested by the client. This SID is always the first byte of the service request message, and it is mandatory.

UDS Frame Format

Next comes an optional field called sub-function and it is also of 1 Byte length, value of this byte tells the server which sub functionality is requested.

The sub functions are needed for some selected services only. For example read date by identifier (RDBI) or write data by identifier (WDBI) where no sub functions are needed.

UDS Frame Format

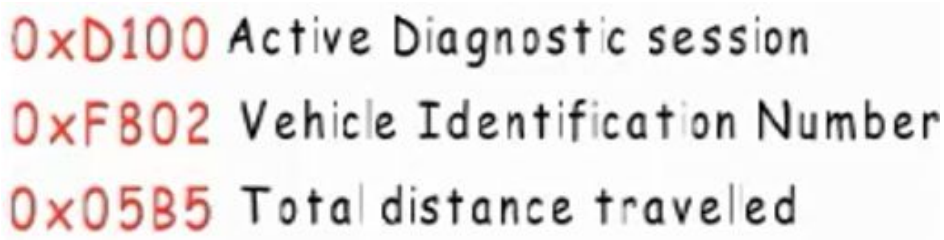


Then there is data identifier field, which is of 2 Bytes length and also it is optional as we don't always need to send data, we may just want to read or even check. In UDS the client and server want to communicate sending numbers, characters, names... Anything

UDS Frame Format

So if you want to read information from the engine speed by a testing tool, Then you can not put the string “Engine speed” in the requested message, For all the data elements which the tester needs, they must also be assigned identifiers. In UDS, this data identifier (DID) is of 2 bytes length, and assigned to all types of data to be sent or received. So let’s say that the motor speed is assigned to 0x1111, the tester and ECU know this information and work upon this information. And different data identifiers are assigned to different services.

Car manufacturers define their data identifier so that only tester tools from OEMs can read these DIDs. Generally, the purpose of this identifier remains same. And also there are services that don’t have DID field, So DID also is optional depending on the service requested.



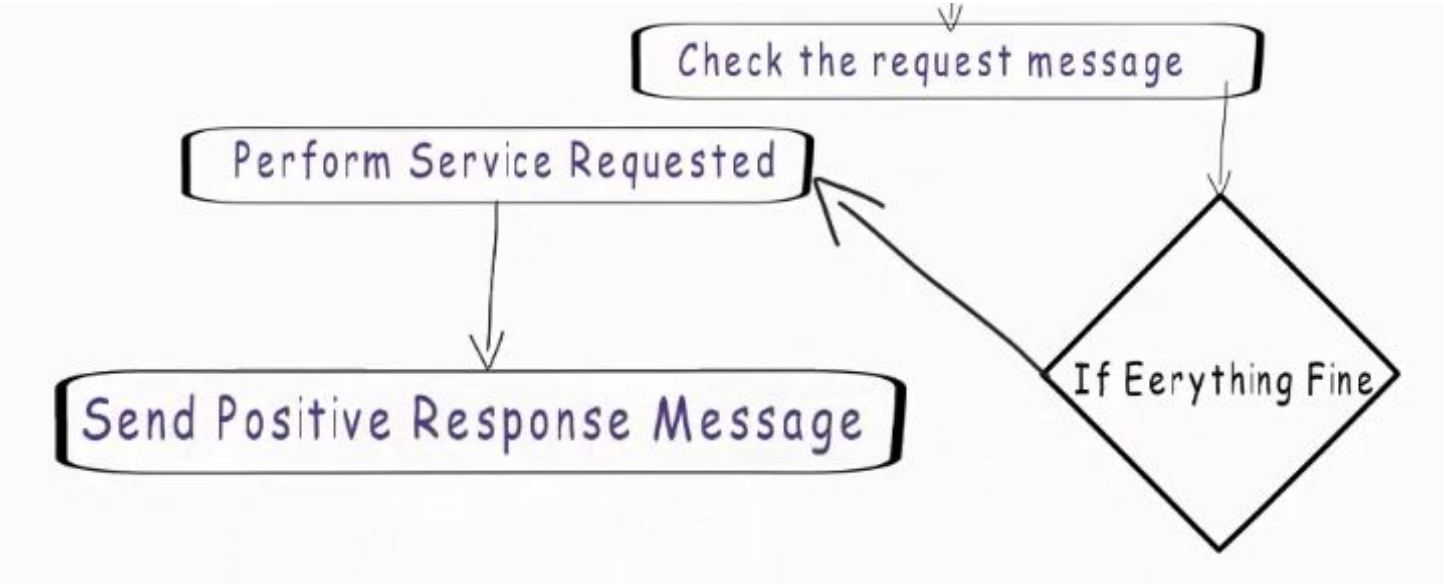
UDS Frame Format

After data identifier, comes the data record field (DataRec), and it is also optional, if there is no DID, then of course there is no DataRec. write data by identifier service(WDBI) needs it, when you say write data by identifier, after mentioning the data identifier, you also need to tell the server which data value has to be written to that data element.

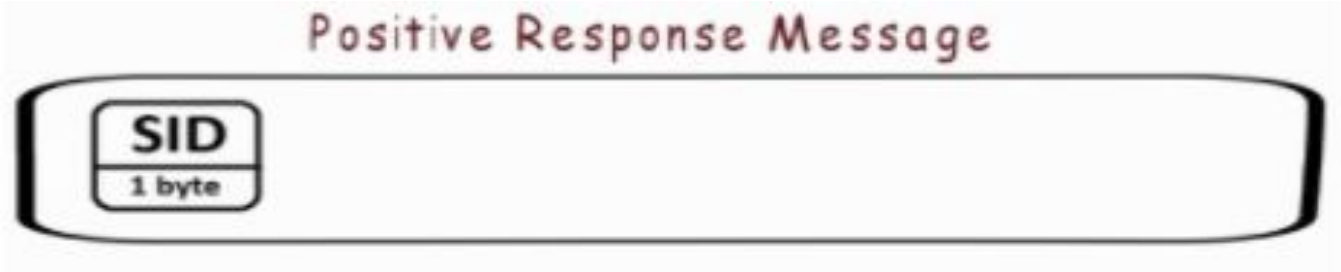
For example, If the (DID 0x9876) represents vehicle's speed limit, and if tester wants to change the current value, then he also needs to specify what new value has to be stored in the data element, So DID tells what data element is referred and the dataRec specifies what new value has to be stored in that data element.

Positive Response

Once the server receives the request message, It checks this request message, if everything is OK “valid ID and the frame is out of errors”, it performs the service request then respond to the client by sending a positive response message.

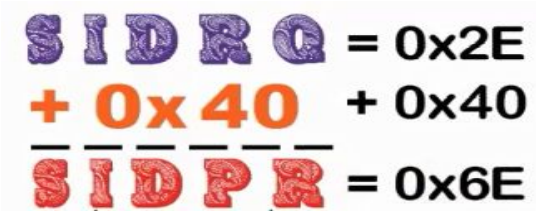


Positive Response Frame



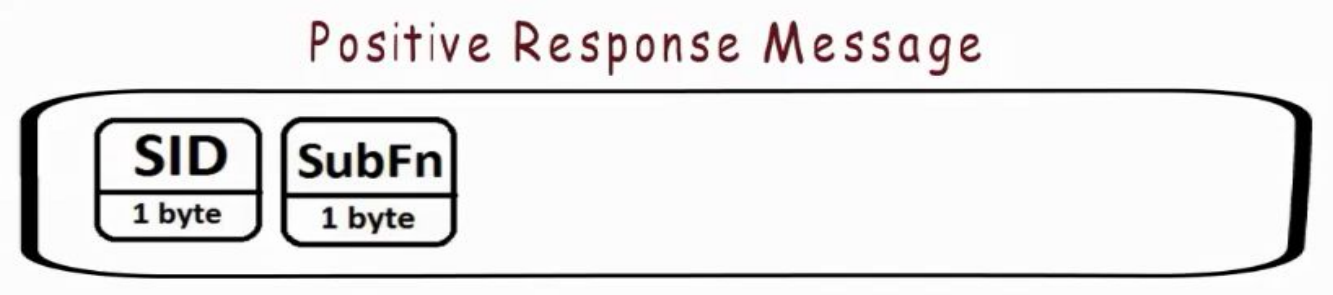
Positive response message starts with positive response service identifier and it is of 1 Byte length. It's value is same as request service identifier with addition of 0x40 to it.

For example, if the requested service identifier is 0x2E. Then the positive response identifier will be 0x6E.



Since the range of the request service identifier is from (0x00:0x3E). Then the range of the positive response identifier is (0x40:0x7E)

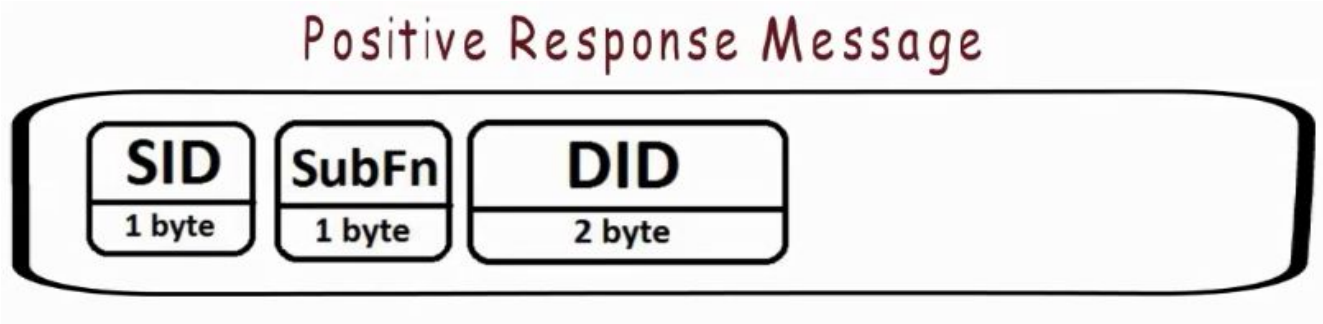
Positive Response Frame



And following the positive response identifier, there is a sub functions which is same as the sub function present in service request message,

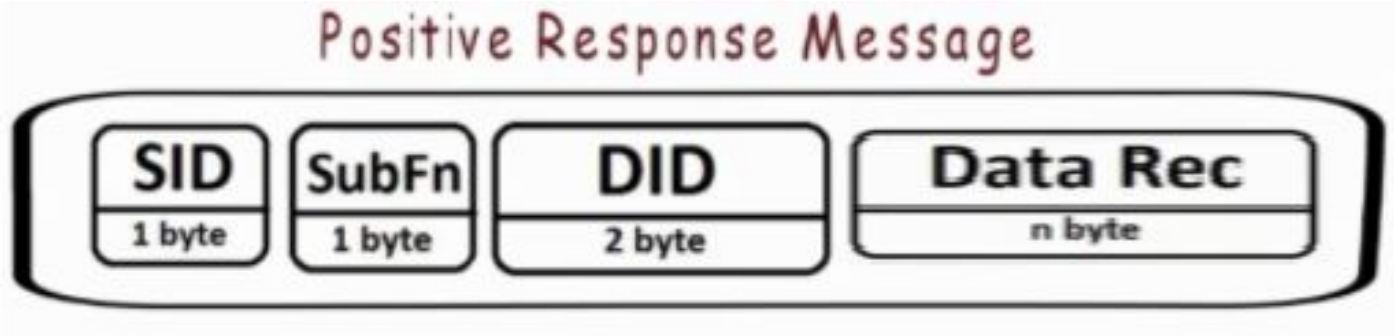
It is also an optional field, if the service request has sub function field, then the positive response also has a sub function field.

Positive Response Frame



And there comes the data identifier with 2 Bytes length. This also exactly as same as the data identifier field present in the service request message and also optional.

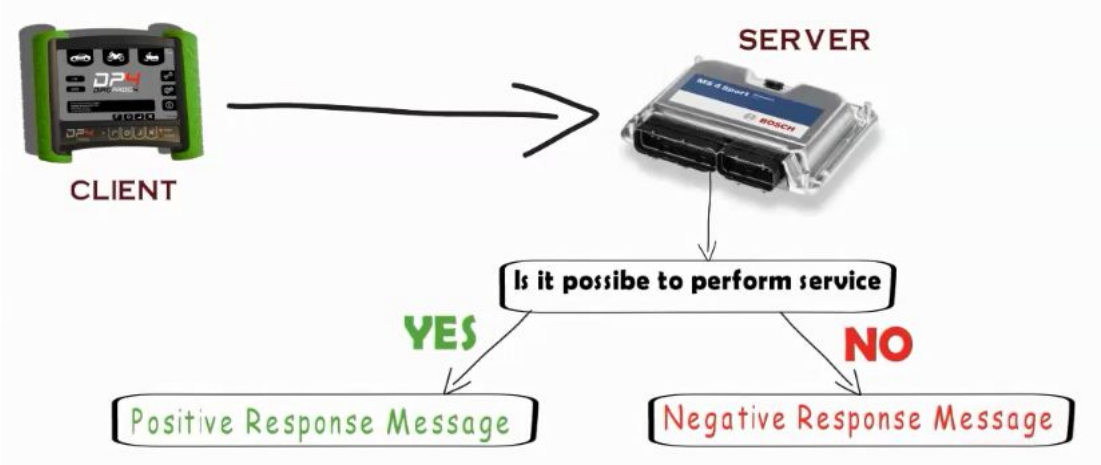
Positive Response Frame



And this DID is followed by the data record field which is optional to service. For example, for service RDBI, Positive Response Message has data record which contains the value of the data element represented by the DID present in the Request Message. So positive response message is almost similar to the request message format except for the addition of 0x40.

Negative Response

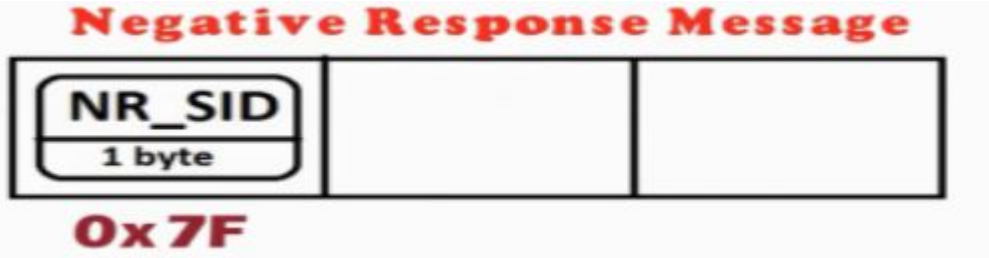
Very simple, if service is not allowed, this response is being sent



Negative Response Frame

Negative response message always contains 3 bytes:

1- Negative response service identifier NR_SID, it is of 1 Byte length and always has the value of 0x7F. Irrespective of what service is requested by client, negative response always starts with 0x7F



Negative Response Frame

Negative response message always contains 3 bytes:

2- Service id SIDRQ: Represents the service that can't be performed,i.e. SIDRQ = SID present in the request message

Negative Response Frame

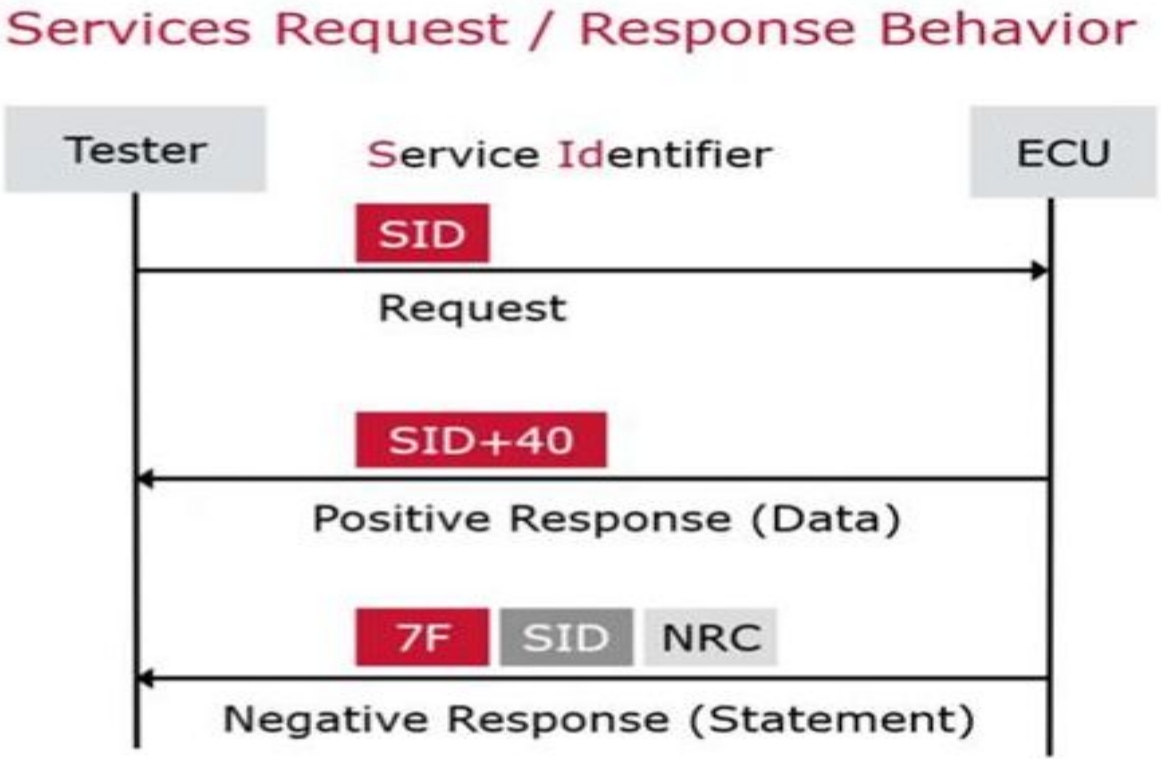
Negative response message always contains 3 bytes:

3- Negative response code NRC
It is being sent due to server not being able to perform the service requested due to some reason, and there are reasons for server to not perform service requested

NRC Example

NRC	Description
10	generalReject
11	serviceNotSupported
12	subFunctionNotSupported
13	incorrectMessageLengthOrInvalidFormat
22	conditionsNotCorrect
31	requestOutOfRange
7E	subFuntionNotSupportedInActiveSession
7F	serviceNotSupportedInActiveSession
...	...

Conclusion


Services Request / Response Behavior
Tester
Service Identifier
ECU
SID
Request
SID+40
Positive Response (Data)
7F
SID
NRC
Negative Response (Statement)

www.imtschool.com

ww.facebook.com/imaketechnologyschool/