# What is Retransmission, ARQ and HARQ?

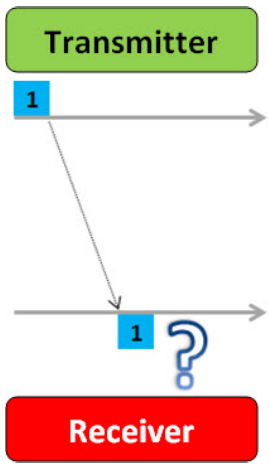
Posted by leopedrini Friday, June 22, 2012 11:12:00 AM Categories: Course

Previous Post << >> Next Post

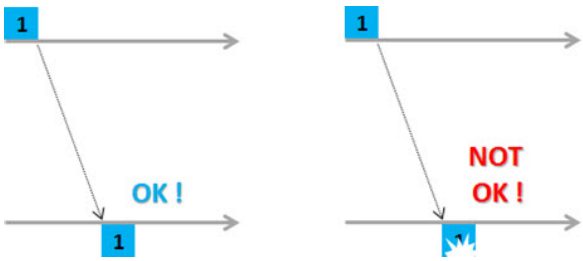
Rate this Content 47 Votes

It's very important to use solutions that improve the efficiency of the adopted model in any data communication system. If the transmission is 'Wireless', this need is even greater.

In this scenario we have techniques that basically checks, or verify if the information sent by the transmitter correctly arrived in the receiver. In the following example, we have a packet being sent from the transmitter to the receiver.




If the information arrived properly (complete), the receiver is ready to receive (and process) new data. If the information arrived with some problem, corrupted, the receiver must request that the transmitter sent the packet again (retransmission).




Let's understand a little more about these concepts increasingly used (and required) in the current systems?



*Note: All telecomHall articles are originally written in Portuguese. Following we translate to English and Spanish. As our time is short, maybe you find some typos (sometimes we just use the automatic translator, with only a final and 'quick' review). We apologize and we have an understanding of our effort. If you want to contribute translating / correcting of these languages, or even creating and publishing your tutorials, please contact us: contact.*

- June 2012 +

| S | M | T | W | T | F | S |
|---|---|---|---|---|---|---|
| 27 | 28 | 29 | 30 | 31 | 1 | 2 |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 17 | 18 | 19 | 20 | 21 | 22 | 23 |
| 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |

RSS FEED

## Statistics

Entries (32)

## Categories

Course (32)
CSFB (1)
IMS (1)
LTE (5)
RF Components (2)
SRVCC (1)
UMTS (1)
VoLTE (1)

## Related Posts

What is Envelope Tracking?
What are Modes, States and Transitions in GSM, UMTS and LTE?
What is CSFB and SRVCC in LTE?
What is CP (Cyclic Prefix) in LTE?
What is LCS (and LBS)?
What does Orthogonal means in Wireless Networks?
What is ISI (Inter Symbol Interference) in LTE?
What is Splitter and Combiner?
Analyzing Coverage with Propagation Delay - PD and Timing Advance - TA (GSM-WCDMA-LTE)
What is RRC and RAB?
IP Packet switching in Telecom - Part 4
IP Packet switching in Telecom - Part 3
IP Packet switching in Telecom - Part 2
IP Packet switching in Telecom - Part 1
Goodbye IPv4... Hello IPv6!

## Archives

April, 2016 (2)
May, 2015 (1)
November, 2014 (2)
October, 2014 (1)
February, 2014 (1)
October, 2013 (1)
June, 2013 (1)

May, 2013 (1)
June, 2012 (1)
March, 2012 (1)
February, 2012 (2)
January, 2012 (1)
November, 2011 (1)
October, 2011 (1)
September, 2011 (1)
June, 2011 (1)
April, 2011 (2)
March, 2011 (3)
February, 2011 (5)
January, 2011 (1)
December, 2010 (2)

## Error Checking and Correction

We start talking about errors. Errors are possible, and mainly due to the transmission link. In fact, we can even 'expect' errors when it comes to Wireless Data Transmission.

If we have errors, we need to take some action. In our case, we can divide it into two steps: error checking and error correction.

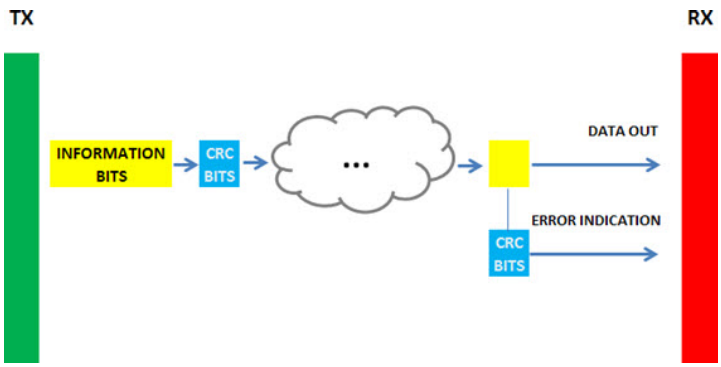


Error checking is required to allow the receiver to verify that the information that arrived is correct or not.

One of the most common methods of error checking is the CRC, or 'Cyclic Redundancy Check', where bits (CRC) are added to a group of information bits. The CRC bits are generated based on the contents of the information bits. If an error happens with the information bits, the CRC bits are used to verify and help recover the degraded information.

The level of protection provided is determined by the ratio: number of CRC bits by the number of information bits. Above a certain error level, the process is eliminated. CRC protection is used practically in all existing Voice and Data applications.

The following diagram shows a simplified demonstration of how the CRC is used.




And the CRC is directly connected to the Error Correction methods. There are various ways of Foward Error Correction (FEC), but the main idea is, given a level of quality in the link, try to get the lowest number of required retransmissions.

Minimizing the number of retransmissions we ended up having a more efficient data flow result, including - mainly - the 'Throughput'.



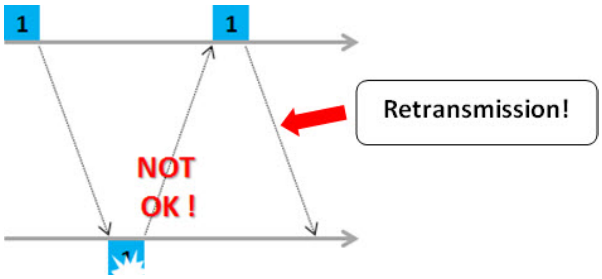
In simplified way: the CRC lets you know if a package arrived 'OK' or 'NOT OK'. Every packet that is sent has a CRC, or a 'Signature'. As an analogy, it's like when we send a letter to someone, and in the end we sign: 'My Full Name'. When the other person receives this letter (information), he checks the signature: 'My Wrong'. In this case, he tells the Messenger: 'I don't know 'My Wrong', this information has some problems. Please ask sender to send it again!'.

I.e. I do CRC checks. If the CRC is 'wrong', the information is 'wrong'. If the CRC is 'correct', probably the information is

'correct'.

## Retransmissions

Retransmissions are then: send information again (repeat) to the receiver, after it make such a request. The receiver requests that the information be retransmitted whenever it cannot decode the packet, or the result of decoding has been an error. That is, after checking that the information reached the receiver is not 'OK', we should request it to be retransmitted.




Of course, when we have a good link (SNR), without interference or problems that may affect data integrity, we have virtually no need for retransmissions.

In practice, in real World, this is very difficult to happen, because the links can face the most different adversities. Thus, an efficient mechanism to enable and manage the retransmission is essential.

We consider such a mechanism as efficient when it allow data communication in a link meet quality requirements that the service demands (QoS).

Voice for example, is a service where retransmission does not apply. If a piece of information is lost, and is retransmitted, the conversation becomes intelligible.

On the other hand, data services practically rely on retransmission, since most have - or allows - a certain tolerance to delays – some more, some less. With the exception only for 'Real Time' services.

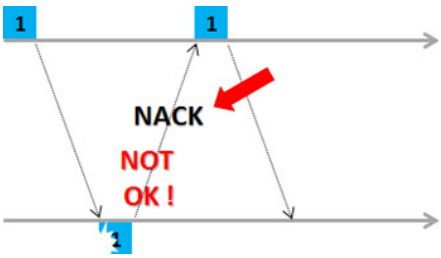
But it is also important to take into account that the greater the number of needed retransmissions, lower the data transmission rate that is effectively reached: If the information have to be retransmitted several times, it will take long for the receiver to obtain the complete - final - information.

## ARQ

Till now we talked in a generic way about data retransmissions, error checking and correction. Let's now see some real and practical schemes.

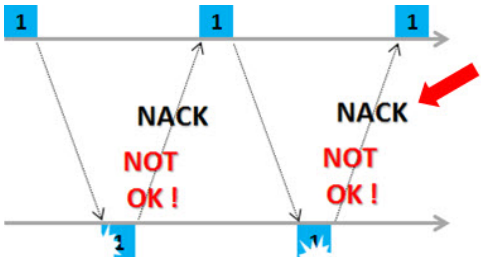
The simplest way (or more common) control using what we described above is known as ARQ, or 'Automatic Repeat Request'.

In ARQ, when we have a 'bad' package, the system simply discards it, and asks for a retransmission (of the same package). And for this, it sends a feedback message to the transmitter.



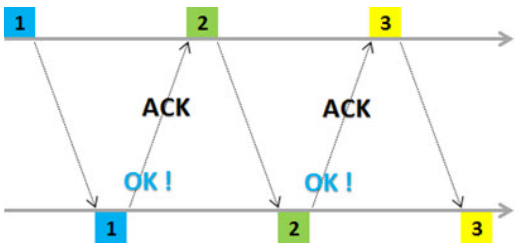

These feedback messages are messages that the receiver uses to inform whether the transmission was successful or not: 'ACKnowledgement' (ACK) and 'Non-ACKnowledgement' (NACK). These messages are transmitted from the receiver to the transmitter, and respectively informs a good (ACK) or bad (NACK) reception of the previous packages.

If in the new retransmission the packet keep arriving with errors, the system requests a new retransmission (still for this same package). That is, sends another 'NACK' message.




The data packets that are not properly decoded are discarded. The data packets or retransmissions are separately decoded. That is, every time a packet that arrives is bad, it is discarded, and it is requested that this same package be retransmitted.

But see that if there were no retransmissions, the performance of the data flow would be much better. In the example below, compared with the previous, we transmit more information - 3 times in the same time interval.




Unfortunately we don't have much to do about the link conditions. Or better, we are able to improve the links performance, for example with configuration parameters optimization, but we'll always be subject to face adverse conditions. In this case, our only way out is to try to minimize retransmissions.

And that's where arise other techniques or more 'enhanced' schemes for retransmission. The main one is HARQ.

### Hybrid ARQ (HARQ)

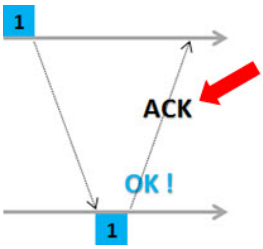
The HARQ is the use of conventional ARQ along with an Error Correction technique called 'Soft Combining', which no longer discards the received bad data (with error).

With the 'Soft Combining' data packets that are not properly decoded are not discarded anymore. The received signal is stored in a 'buffer', and will be combined with next retransmission.

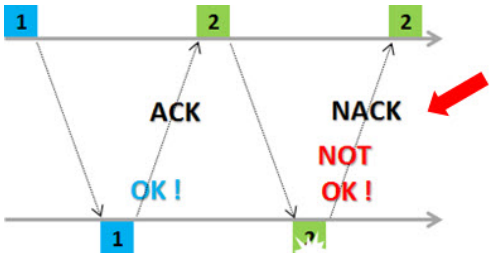


That is, two or more packets received, each one with insufficient SNR to allow individual decoding can be combined in such a way that the total signal can be decoded!

The following image explains this procedure. The transmitter sends a package [1]. The package [1] arrives, and is 'OK'. If the package [1] is 'OK' then the receiver sends an 'ACK'.

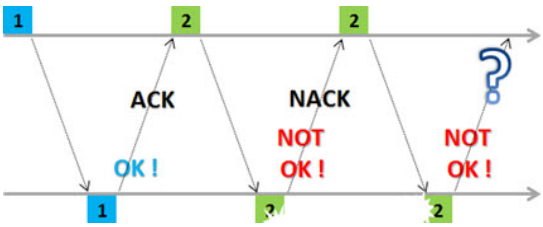



The transmission continues, and is sent a package [2]. The package [2] arrives, but let's consider now that it arrives with errors. If the package [2] arrives with errors, the receiver sends a 'NACK'.




Only now this package [2] (bad) is not thrown away, as it is done in conventional ARQ. Now it is stored in a 'buffer'.

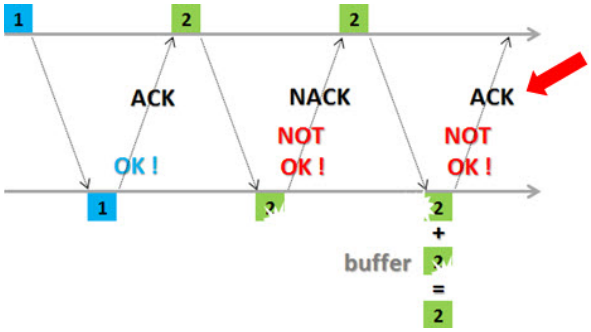

Continuing, the transmitter send another package [2.1] that also (let's consider) arrives with errors.



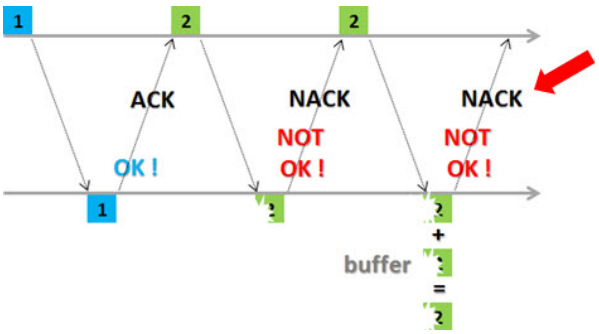

We have then in a buffer: bad package [2], and another package [2.1] which is also bad.

Does by adding (combining) these two packages ([2] + [2.1]) we have the complete information?

Yes. So we send an 'ACK'.

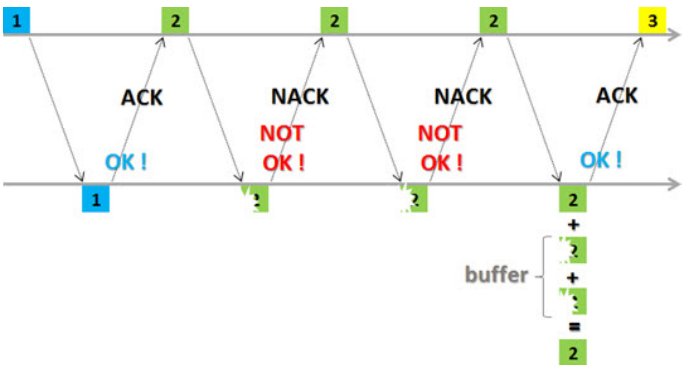



But if the combination of these two packages still does not give us the complete information, the process must continue - and another 'NACK' is sent.




And there we have another retransmission. Now the transmitter sends a third package [2.2].

Let's consider that now it is 'OK', and the receiver sends an 'ACK'.




Here we can see the following: along with the received package [2.2], the receiver also has packages [2] and [2.1], that have not been dropped and are stored in the buffer.

In our example, we see that the package arrived 2 times 'wrong'. And what is the limit of these retransmissions? Up to 4. IE, we can have up to 4 retransmission in each process. This is the maximum number supported by 'buffer'.

### Different HARQ Schemes

Going back a little in the case of Conventional ARQ, whenever we send a package and it arrives with problems, it is discarded.

Taking the above example, when we send the package [2], and it arrives with errors, it is discarded. And this same package [2] is sent again.

What happens is that we no longer have the concept of 'package version' - [2.1], [2.2], etc. We do not have the 'redundancy' version, or the gain we get in HARQ processing.

To understand this, we need to know that information is divided as follows:

*[Information + Redundancy + Redundancy]*

When we transmit the packet [2] we are transmitting this:

*[Information + Redundancy + Redundancy]*

When retransmit the same package [2] we are retransmiting it again:

*[Information + Redundancy + Redundancy]*

But when we use HARQ, and retransmit packet [2.1] or [2.2], we have the possibility of:

- Or retransmit that same information again;
- Or retransmit only the redundancy.

And then, if we retransmit less information (only redundancy), we spend less energy, and that will run much faster. With this we have a gain!

That is, we work with different 'versions of redundancy', that allows us to have a gain in the retransmission. This is called 'Redundancy Version', or what version of redundancy.

The redundancy version, or HARQ scheme with 'Soft Combining' can be 'Chase Combination' or 'Incremental Redundancy'.

### HARQ Chase Combination

'**Chase Combination**': when we combine the same information (the retransmission is an identical copy of the original packet).

We transmit an information, which arrived wrong, and we need to do a retransmission. We retransmit the same information - and there we don't have much gain.

### HARQ Incremental Redundancy

'**Incremental Redundancy**': where we retransmit only the portion that we didn't transmitted before. Thus we retransmit less information. Less information means fewer bits, less energy. And this gives a gain!

Redundancy bits are retransmitted gradually to the receiver, until an ACK is received.

With this, we adapt to changes in the condition of the link. The first retransmission can, for example, contain or not bits of redundancy. If necessary, a small number of these bits is retransmitted. And so on.

### Finishing for today: what are the 2 steps of HARQ? Why it gives me a Gain?

- First because from wrong packets 1 and 2 we can get a correct one, since we do not discard erroneous packets anymore.
- Second because we can - also in retransmission - send less information, and streamline the process.

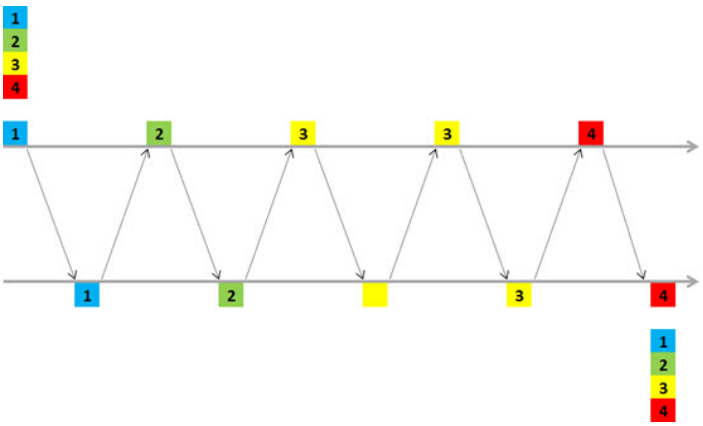
The use of HARQ with 'Soft Combining' increases the received Eb/Io effective value for each retransmission, and therefore also increases the likelihood of correct retransmissions decoding, in comparison to conventional ARQ.

We send a package, and it arrives with errors: we keep this package. Receive the retransmission and then we add or combine both.

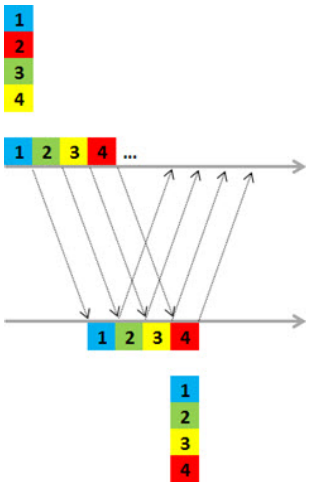
### HARQ Processes (Case Study)

What we have seen so far clarifies the concepts involved. In practice, in retransmission, this type of Protocol is called 'Stop And Wait' (there are other kinds of similar protocols).

What would be: send the information and stop. Wait for the response to send other information. Send, wait for response. Send, wait for response ...




No! Not so in practice. In practice, we work with a number of 'processes', which may vary for example from 4, 6 or 8. The following image illustrates this more clearly.




### Other types of HARQ

New schemes are constantly being developed and used, as the type III HARQ, which uses self-decodable packages.

But enter these variations, terminology and considerations, is not the scope of our tutorial, which was simply to introduce the concept of Retransmission, ARQ and HARQ.



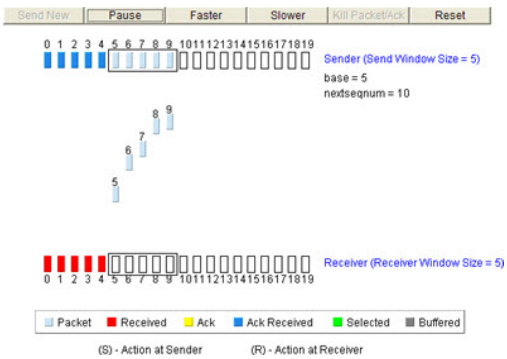
Based on the key concepts illustrated here today, you can extend your studies the way you want, however we believe that the most important thing was achieved – understand how it works and what are all the cited concepts.

### JAVA Applet

Below, you can see how some retransmission schemes work. There are several Applets available, for the many possibilities (ARQ, HARQ, With Sliding Windows, Selective, etc).

The next is a link for a JAVA Applet that simulates a 'Selective Repeat Protocol transmission'.

http://media.pearsoncmg.com/aw/aw_kurose_network_4/applets/SR/index.html




### Conclusion

This was another tutorial on important issues for those who work with IT and Telecom: data Transmission and Retransmission techniques, ARQ and HARQ.

ARQ is used for applications that allow a certain delay, as Web Browsing and Streaming Audio/video. It is used widely in Wimax and WiFi communication systems. However, it cannot be used in Voice transmission, as for example in GSM.

HARQ for example is used in HSPA and LTE, and therefore must be a well-understood concept for those who work or want to work with these technologies.

We hope you enjoyed it. And until our next tutorial.



Previous Post << >> Next Post

**6 Comments** Sort by **Newest**

Add a comment...

**Lữ Tấn Hoà**

Great explanation. Thanks

Like · Reply · 1y

**صالح القماطى**

thank you very much
but i have a question
can we use these Techniques when we want to make a voice call in lte ?

Like · Reply · 2y

**Mahendra Padhiary**

Why there is 8 different harq processes are there in LTE? It is not possible to use one process? Is there something like one process1 handle the transimssion for 1st packet, process2 handle the same for 2nd packet and so on ... for 8 processess?

Like · Reply · 2y

**ਗੁਰਜੰਟ ਸਿੰਘ**

nice explanation

Like · Reply · 3y

**Mahender Singh**

Good One.

Like · Reply · 1 · 5y

**Siddharth Chourey**

suoerb info guys..it just my doubts are clear now

Like · Reply · 1 · 5y

Facebook Comments Plugin