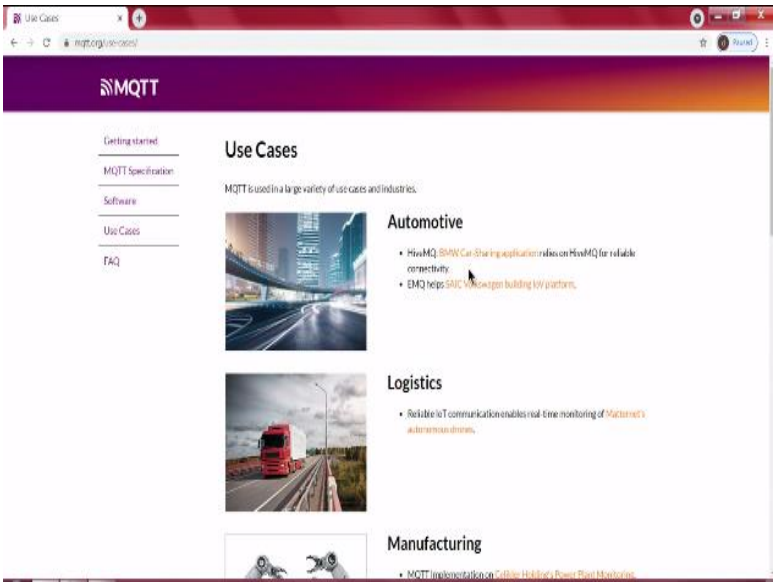
Design for Internet of Things
Prof. T V Prabhakar
Department of Electronic Systems Engineering
Indian Institute of Science-Bengaluru

Lecture - 34
MQTT – 01

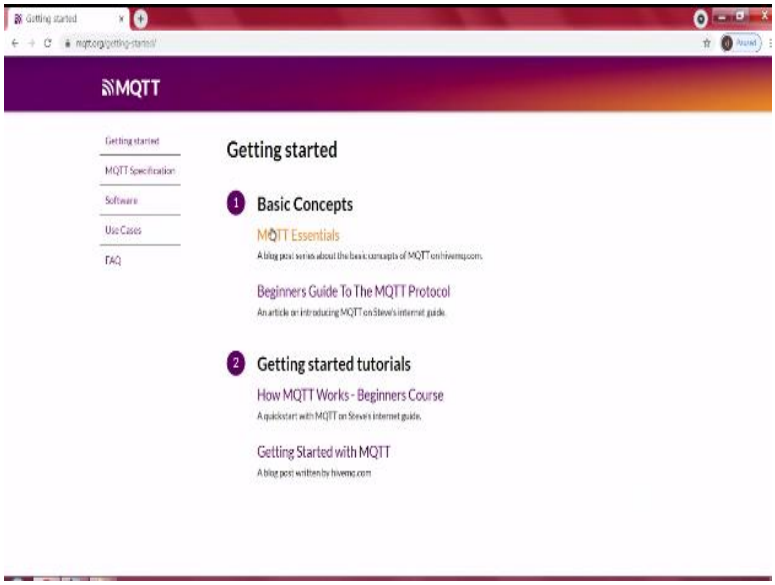
Everything and anything you want to know about this protocol, its use cases, the specification, the deeper understanding of the protocol, everything you should do by going to one website. And that website is what I have shown here.

(Refer Slide Time: 00:45)



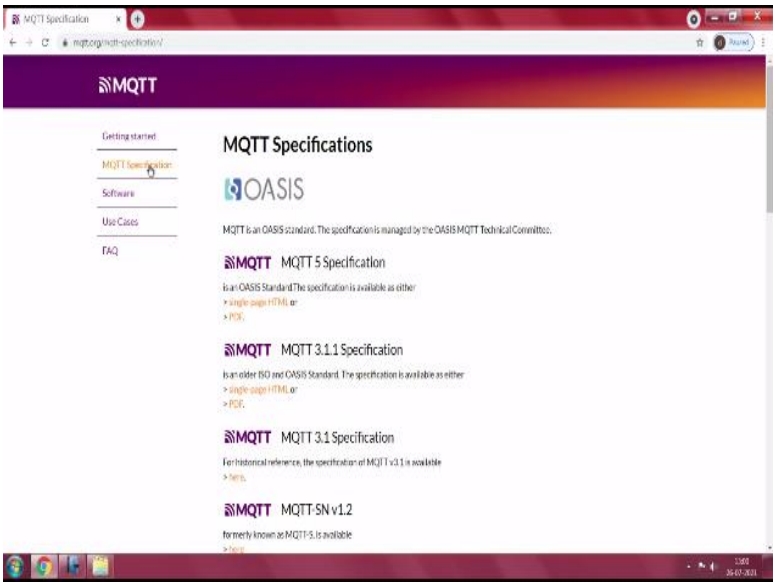
MQTT.org. Go there it is the repository of every piece of information that is available. I would not go into the details. Knowing MQTT is like knowing lot about IoT. In fact, it hinges on the fact that you know the protocol very well, okay. So that is very important. So now I go to this Getting Started.

(Refer Slide Time: 01:11)



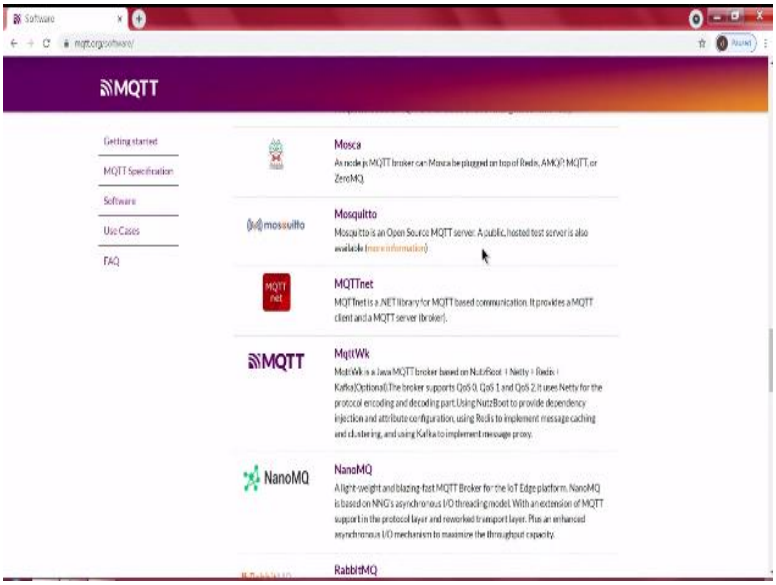
You can see there are some MQTT essentials. There is a beginner's guide to MQTT and how it works and so on, right? You can read them as carefully as possible.

(Refer Slide Time: 01:26)



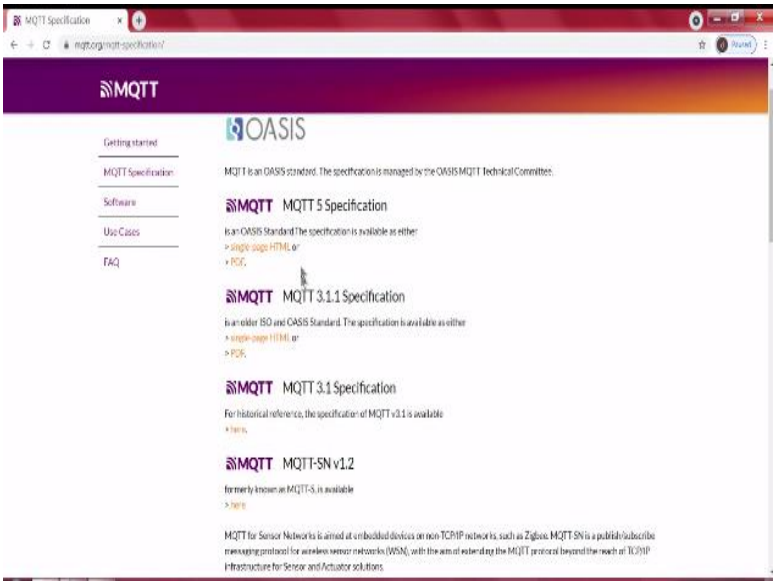
And then you can also look up the MQTT specification.

(Refer Slide Time: 01:36)



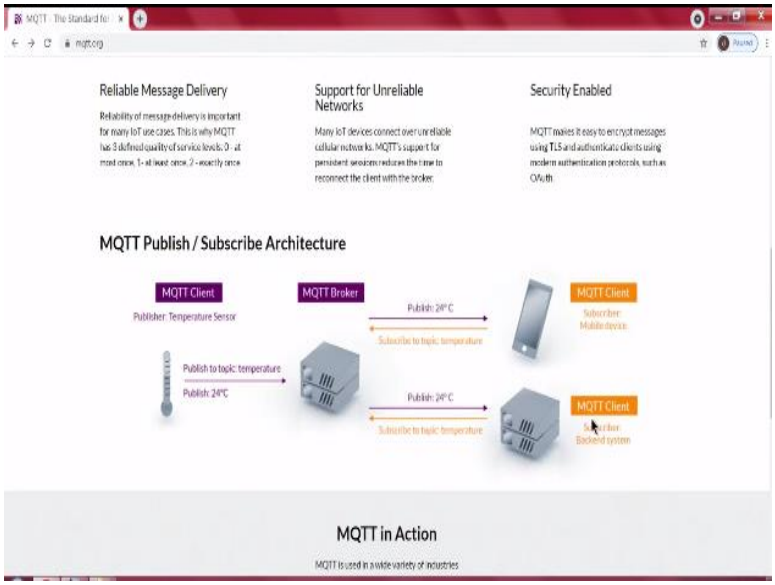
You can also look at the MQTT software. If you go to servers and brokers, you will see that I mentioned about Mosquitto is an open source MQTT server. So, so many of them are out there, okay.

(Refer Slide Time: 01:51)



And so if you go to the specification, MQTT 3.1.1 and then 5, right? are two very popular versions of MQTT. You must start somewhere, perhaps with a lower version of MQTT. So pick up this PDF and read that PDF in great detail, okay. So let me go back. Right, this is how we began.

(Refer Slide Time: 02:13)



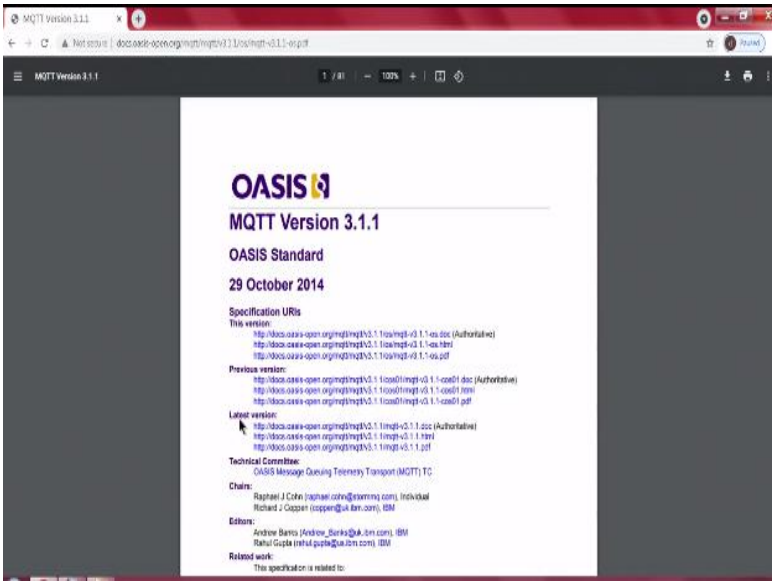
Folks, every single piece of information here is useful. Look at this picture. How different is this picture from what we drew folks about our blinds example? We had in our case light sensor, here he has a temperature sensor. That is the only difference. He is publishing temperature sensor and you publish temperature means your topic is temperature.

So you publish to topic called temperature and what is the value? Value is 24 degrees. In our case it was carrying the amount of light intensity. Here it is just carrying temperature value. So that is it. It goes to the broker. Now this information, since it came to the broker, broker in turn published it to this MQTT client. Why did it publish? Because this client had subscribed to the topic called temperature.

So it is out here already. So you publish 24. He got 24, he published 24. Similarly, this broker also published 24 to this other client, why? Because this client previously had subscribed to the topic called temperature. So it is as simple as that. You have a subscriber which is a mobile device. There is a subscriber which is the back end. So someone is collecting information.

Someone is using this information directly. So both are possible, right? So you can build applications of this nature. Great. Now let us go into the specification. And what is important is to read little bit of the specification in detail.

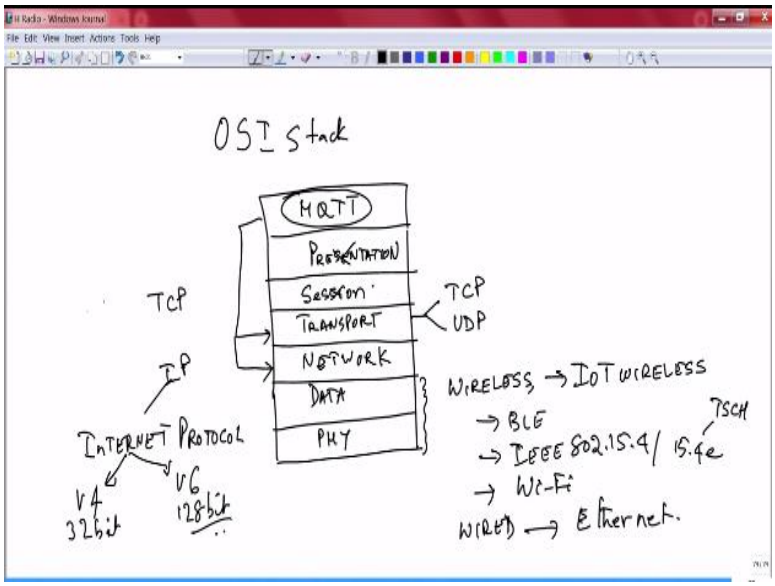
(Refer Slide Time: 04:05)



Folks do not ever pick any other information on MQTT, this is the most authentic one, okay. This is because this is the Oasis standard and this is the standard which tells you all about the protocol, okay. We know this part very well now. It is based on client server. It is based on publish and subscribe. It is used for two machines to talk to each other, which means it is machine to machine communication.

No humans are required. And it is, it has an extremely small code footprint. And it runs over TCP/IP network. So folks here is where you should be clear of one other concept which I will write here.

(Refer Slide Time: 05:03)



Normally, if you go back to the OSI stack okay, if you go to the OSI stack, you will see that you have the seven layer model. The very first layer is the PHY or the

physical layer. You have data link. You have network. You have transport. You have session here. You have presentation. And here you have application, okay. This is your seven layer 1, 2, 3, 4, 5, 6, 7. MQTT runs here, okay?

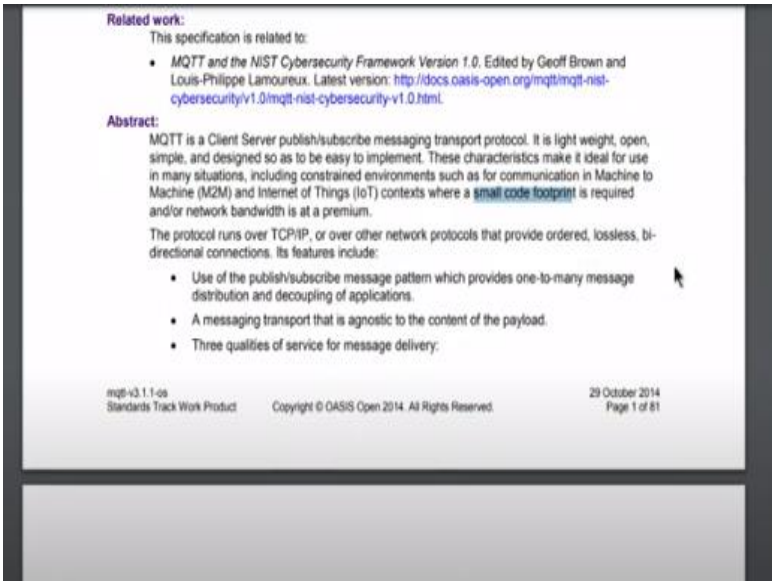
It uses transport layer which is called TCP. Other transport layer is actually called, one is called TCP, the other is called UDP. TCP in turn uses network layers stack called IP. This is nothing but Internet Protocol. This also known as the IP address, which you perhaps know. Here you have V 4 and you have V 6. This is 32 bit and this is 128 bit. Address itself is 128 bit in version 6. In version 4 we have 32 bit address.

And then usually this will be wireless, usually it will be wireless. Some IoT wireless protocol usually, okay. This can be Bluetooth low energy or e if not sometimes 4 or it can also be 15.4e. This is also called time synchronized channel hopping. This is called very popularly you might be knowing about Zigbee, right? It can also be Wi-Fi, nothing stops you. Of course it can be wired also.

It can be wired, I am not saying that it cannot be wired. In which case it can also be Ethernet, nothing stops you. Ethernet MAC and Ethernet PHY. I will just put it like that, okay. Ethernet is mostly a MAC layer protocol. Associated PHY layer is there. Then Bluetooth is the same, which all work in this part of the stack. So we are interested in MQTT, and the fact that usually these layers are used in a merged manner.

So I will not describe these layers too much. It is actually merged. So we will not elaborate that too much. Now let us go back with this background. Let us go back here.

(Refer Slide Time: 08:43)

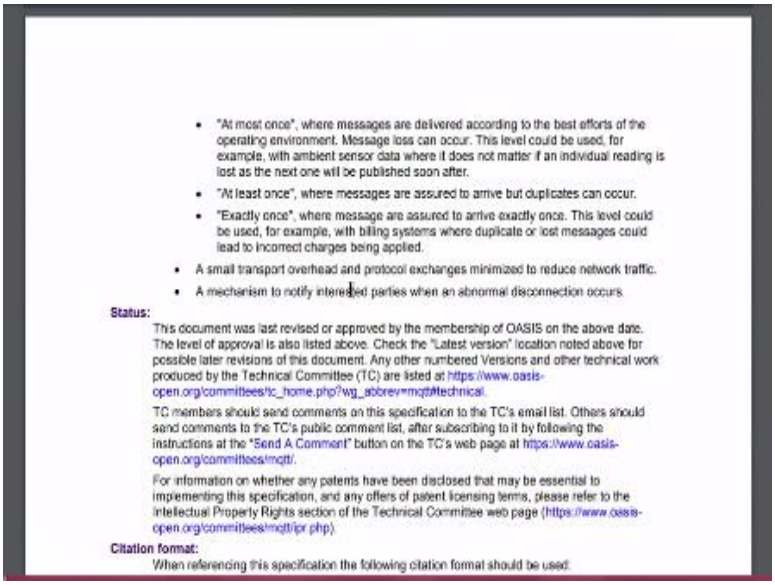


So what it is saying is it provides ordered, lossless, bi-directional connections and it uses super features like publish and subscribe, one-to-many distribution and decoupling of applications. Not very hard for you to make out. Because if you go back to this one, it decouples the applications. What a beautiful answer it gives. Go back to this picture.

You built an IoT node LED lamps just by taking three input sensors. This application is independent of the other application which is motorized blind IoT node. That also took three input sensors and built its own application right, which means you are building applications independently. That is what this statement actually means. You are decoupling of applications.

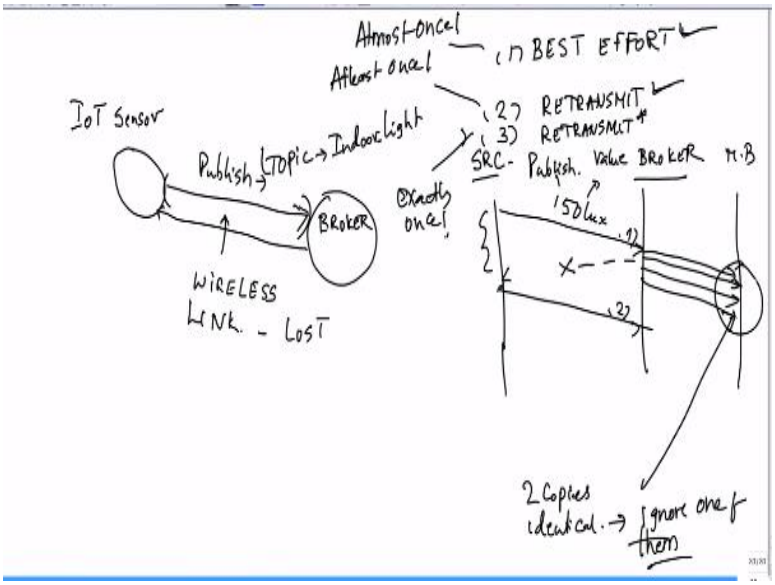
One application has nothing to do with the other. You are only using sensor data to build. Every time the message is sent over a wireless link, you will provide a certain quality of service which are of three types.

(Refer Slide Time: 10:09)



And what are the three? At most once, at least once, and exactly once. What does it mean? It goes back to this picture. Let us redraw this to understand the quality of service.

(Refer Slide Time: 10:27)



You have the broker. For simplicity, I will only draw one IoT node, which is publishing to broker. This is the publish. What is the topic? Topic is indoor light, is it not? That is the name of the topic, published to topic indoor light. I have written like this. This is your sensor, IoT sensor. This is a wireless link. This is a wireless link is it not? And why is it a wireless link?

Because this is an indoor sensor, it has to be connected to a broker like a Raspberry Pi and you are passing that message over a wireless link. Clearly the message can be lost

in which case you have to detect that the broker did not get the message and you have to retransmit that message, okay. You may have to retransmit that message. Now this is one way.

The other way is you transmit the message and say I am not expecting any acknowledgment back. So I really do not know whether it got transmitted and got received successfully or not. But I do not want to know about the fact that it was received successfully by the broker. That is one option. I do not want to know anything about that. So you can tell that let me try best effort, okay.

Second option is you try to retransmit, okay. You try to do a retransmit, okay? You do a retransmit and you have to somehow take care of the fact that there can be multiple copies because you did not get an acknowledgment perhaps from the broker in time and you retransmitted it. Think of the following. You have what is this? This is the broker. This is the source, which is nothing but the indoor light sensor.

It transmits and expects an ACK, okay? This is an ACK. If an ACK comes within this time, no problem source would not do anything. But if the ACK did not come, in fact it got lost till that time, what will the source do? Retransmit it. What will it retransmit? The same value that it had sensed earlier. Let us say the indoor light was 150 lux. 150 lux, pretty dark actually. What is this? This is the value.

What type of message is it? It is a published message, published message and it is sending that value in it. And it expected an acknowledgment in one case, and in the other case, it said I will publish. I do not care whether I get anything or not. That is this case. You send a value, you publish and do not expect an acknowledgment. So this does not appear also. You just send it that is it. The second thing is retransmit.

That is where you wait, you expect the system to give you an acknowledgement. And if the acknowledgement did not arrive in time, because the acknowledgment was lost, then you do a retransmit of the same sensed value back to the broker, okay? Now what actually happened here? Broker got how many copies, two copies. Why he got two? He got one here and he got one more here.

It is the acknowledgement which got lost, not the data packet. So now somehow two copies will be given to the application, okay. Who is the application? The application is from broker to motorized blind. How many he gave, two. So this is called the motorized blind, I will simply call it MB. Now what is the business of motorized blind?

The motorized blind gets two copies, identical and it has to ignore one of them okay, one of them. Think of this folks. How nice it will be irrespective of the number of packets I got retransmitted, if only one was delivered to the application, you may have received many no problem, you may have received many.

But if you can deliver only one to the motorized blind application out of the four or the five or the six that you received, if you are able to deliver only one to the application, then that is the third way okay, that is the third way. This is called, this is called best effort. This is called retransmit. This is also retransmit with a star mark. Why is it star? Because receiver node has a beautiful mechanism of getting rid of all the duplicates and delivering only one to the application.

Now you have to put some names to this, okay. What are the names is the question. So let me go back. See this is how you understand the protocol in great detail. At most once, at least once, exactly once. At most once. What is this? At least once. This is what? Exactly once. There you are. Go back and read this standard. You understand it now beautifully. At most once.

Where messages are delivered according to the best efforts of the operating environment. Messages loss can occur. This level could be used for example, with ambient sensor data where it does not matter if an individual reading is lost as the next one will be published soon after. So it really does not matter if you lose one sample. That is what he says. At least once, where messages are assured to arrive, but duplicates can occur.

And no way of processing the duplicates. All duplicates are delivered, okay? Third one, messages are assured to arrive exactly once. This level could be used for example, with billing systems where duplicate or lost messages could lead to incorrect

charges being applied. Okay, so there you are. Exactly once delivers only one message up there. So this is the overview of QOS setting off MQTT.

Let us move on to understand more about this protocol as we know go along. Thank you very much.