UNIT - V

DESIGN PRINCIPLES FOR WSNS.

WSNs are Networks of small, low-power devices that are used to monitor and collect data from the surrounding environment. There are several design principles that are important to consider when designing a WSN, including:

* Energy Efficiency: WSN nodes are often powered by batteries, so it is important to design the network in a way that minimizes energy consumption.
* Scalability: WSNs can potentially include thousands or even millions of nodes, so it is important to design the network in a way that allows it to scale easily.
* Robustness: WSNs often operate in harsh environments and can be subjected to interference, so it is important to design the network to be Robust and resilient to these challenges.
* Security: WSNs often handle sensitive data, so it is important to design the network to be secure against attacks and data breaches.
* Interoperability: WSNs may need to integrate with other systems and devices, so it is important to design the network to be interoperable with other technologies.
* Adaptability: WSNs may need to adapt to changing environments and Requirements, so it is important to design the network to be flexible and able to adapt to these changes.

5.2 - Gateway concepts:

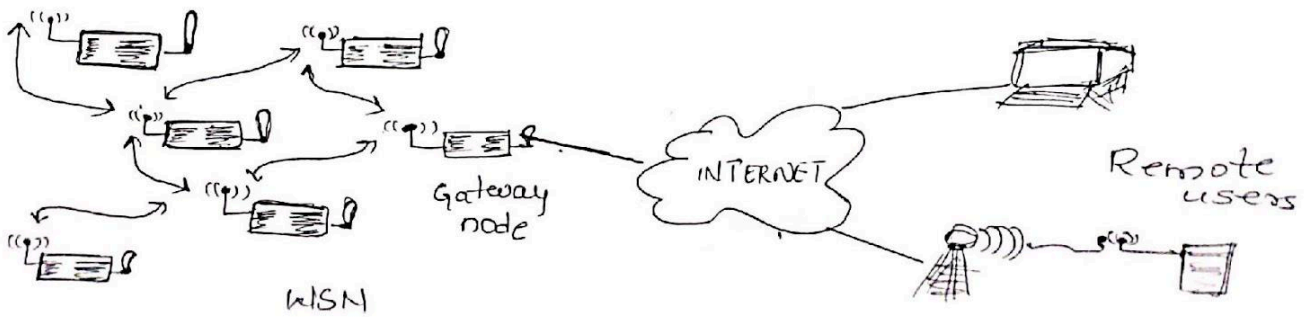
* To this end, the WSN first of all has to be able to exchange data with such a mobile device or with some sort of gateway, which provides the physical connection to the internet.
* This is relatively straightforward on the physical MAC, and link layer- either the mobile device / the gateway is equipped with a radio transceiver as used in the WSN, or some (probably not all) nodes in the WSN support standard wireless communication technologies such as IEEE 802.11. Either option can be advantageous, depending on the application and the typical use case.
* In a WSN, a gateway is a device that connects the sensor nodes to other networks, such as the internet. The gateway serves as the interface between the WSN and the outside world, allowing sensor data to be transmitted to remote locations for analysis and processing.
* There are several key concepts related to gateway design and operation in WSNs:
* communication protocols: The gateway must be able to communicate with the sensor nodes using the appropriate communication protocols. This may involve support for multiple protocols depending on the specific needs of the WSN.
* Data aggregation: The gateway may be responsible for aggregating data from multiple sensor nodes before transmitting

to the outside world. This can help to reduce the amount of data that needs to be transmitted, and can also be used to perform simple data processing tasks.

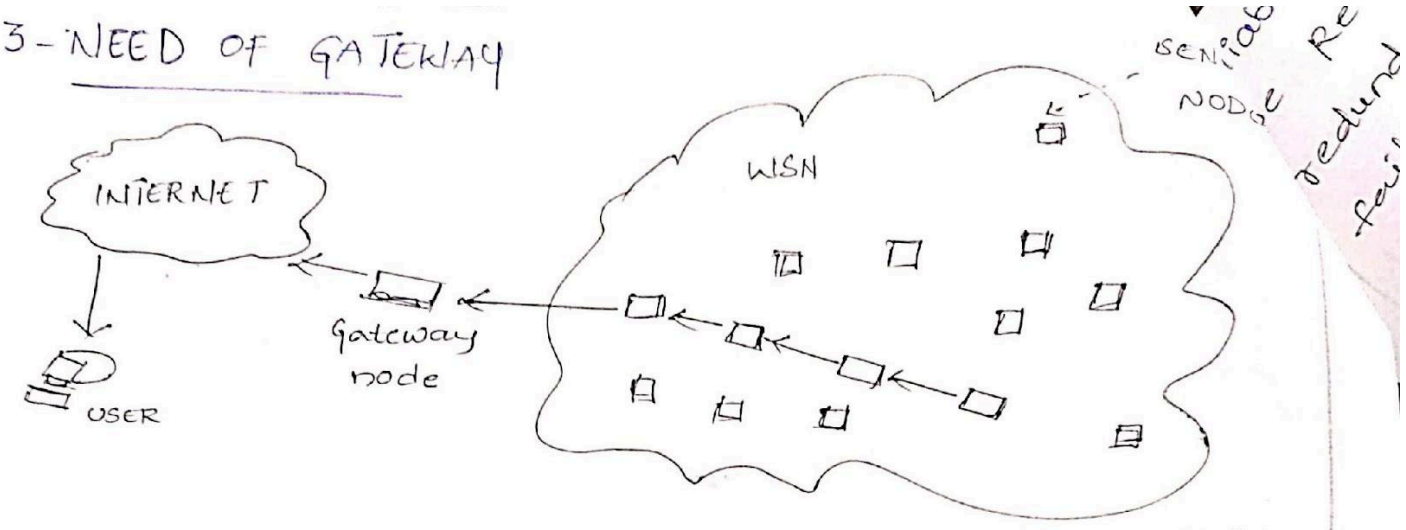
* Power Management: The gateway may need to manage the power consumption of the sensor nodes to ensure that they do not run out of power before their batteries can be replaced or recharged.
* Security: The gateway must be secured against attacks, as it is the primary point of entry for outside access to the WSN. This May involve the use of encryption and other security measures to protect against data breaches.
* Deployment: The gateway must be carefully deployed in the WSN to ensure that it has good coverage and can communicate effectively with the sensor nodes. This may involve the use of multiple gateways to provide redundancy and improve reliability

GATEWAYS IN WSN/MANET

- Allow remote access to/from the WSN
- Bridge the gap between different Interaction Semantics.
  eg: data Vs. address-centric networking.
- Need Support for different radios/protocols



5.3 - NEED OF GATEWAY



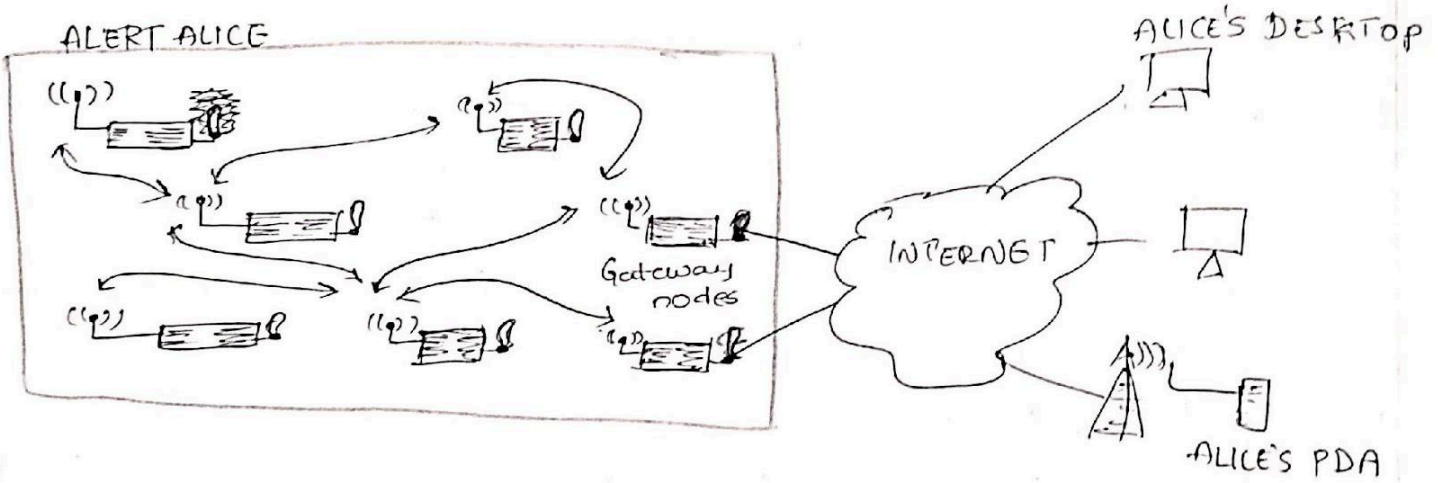
There are several reasons why a gateway may be needed in a WSN:

* Data transmission: The primary function of a gateway in a WSN is to transmit data from the sensor nodes to remote locations for analysis and processing. Without a gateway, the sensor nodes would be isolated and unable to trasmit their data.
* Network Management: The gateway can be used to manage and control the sensor nodes in the WSNs. for example, it can be used to update the firmware on the nodes, or to configure their sensor parameters.
* Security: The gateway serves as the primary point of entry for outside access to the WSN. By securing the gateway, it is possible to protect the entire WSN against attacks and data breaches.
* Interoperability: The gateway can be used to integrate the WSN with other systems and devices. for eg: it can be used to transmit data from the WSN to a cloud-based analytics platform, or to a local server for further processing.

Reliability: The use of a gateway can improve the Reliability of the WSN by providing a redundant communication path. If a sensor node fails or goes offline, the gateway can be used to transmit data from other nodes in the network.

WSN to internet communication:

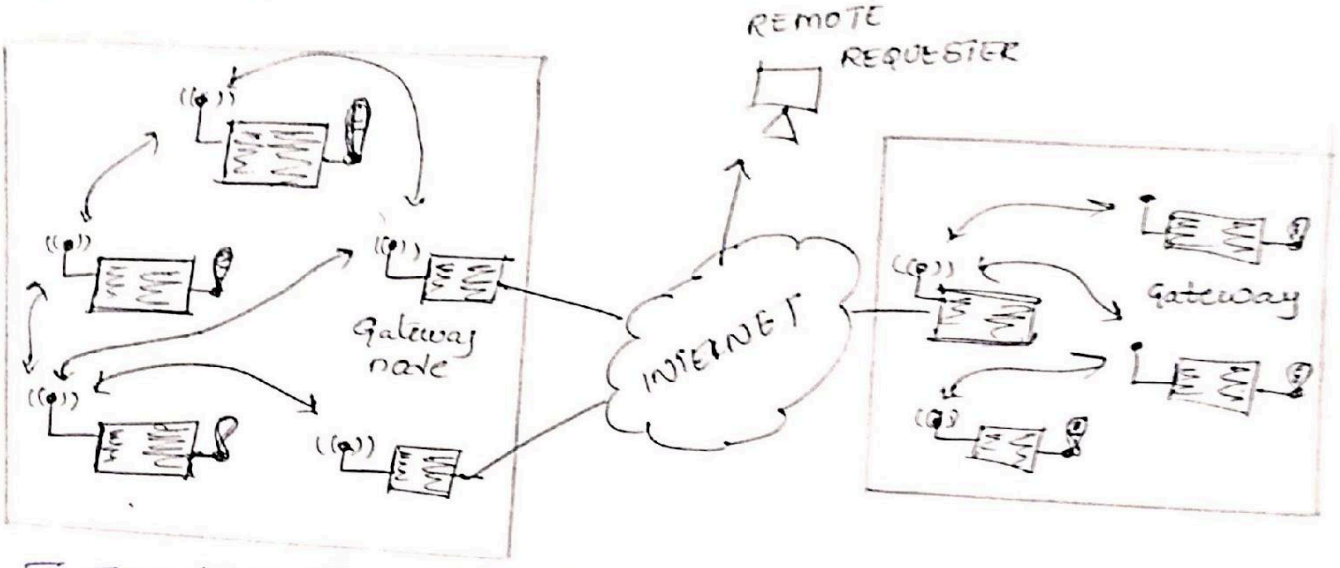
* Eg:- deliver an alarm message to an internet host.
* ISSUES:
  - Need to find a gateway (integrates routing and service discovery)
  - Choose "best" gateway if several are available
  - How to find Alice or Alice's IP?



* Assume that the initiator of a WSN - internet comm. resides in the WSN - for eg:- a sensor node wants to deliver an alarm message to some internet host.
* The first problem to solve is akin to ad hoc networks, namely, how to find the gateway from within the network.
* Basically, a routing problem to a node that offers a specific service has to be solved, integrating routing and service discovery.

* WSNs are networks of small, low-power devices that are used to monitor and collect data from the surrounding environment. In order to transmit this data to the internet or other external networks, a gateway is typically used to connect the WSN to these Networks.
* There are several different ways that a WSN can communicate with the internet, depending on the specific needs of the network and the available infrastructure.
* Cellular:- A WSN can use a cellular Network to transmit data to the internet. This can be done using a cellular modem connected to the gateway or by using specialized sensor nodes that include a built-in cellular modem.
* WiFi: If the WSN is deployed in an area with WIFI coverage, it can use a WIFI network to transmit data to the internet. This can be done using WIFI modem connected to the gateway, or by using specialized sensor nodes with built-in WIFI
* Satellite: In remote or hard-to-reach areas, a satellite network may be used to transmit data from the WSN to the internet. This can be done using a satellite modem connected to the gateway, or by using specialized sensor nodes with built-in satellite modems.
* Long-range Wireless: In some cases, it may be possible to use long range wireless technologies, such as LORa or Sigfox, to transmit data from the WSN to the internet. These technologies can be used to transmit data over long distances, making them well-suited for remote or hard-to-reach areas.

Requesting sensor network information from a remote terminal entails choices about which network to address, which gateway node of a given network, and how and where to adapt application-layer protocol in the internet to WSN-specific protocols.



INTERNET TO WSN COMMUNICATION:

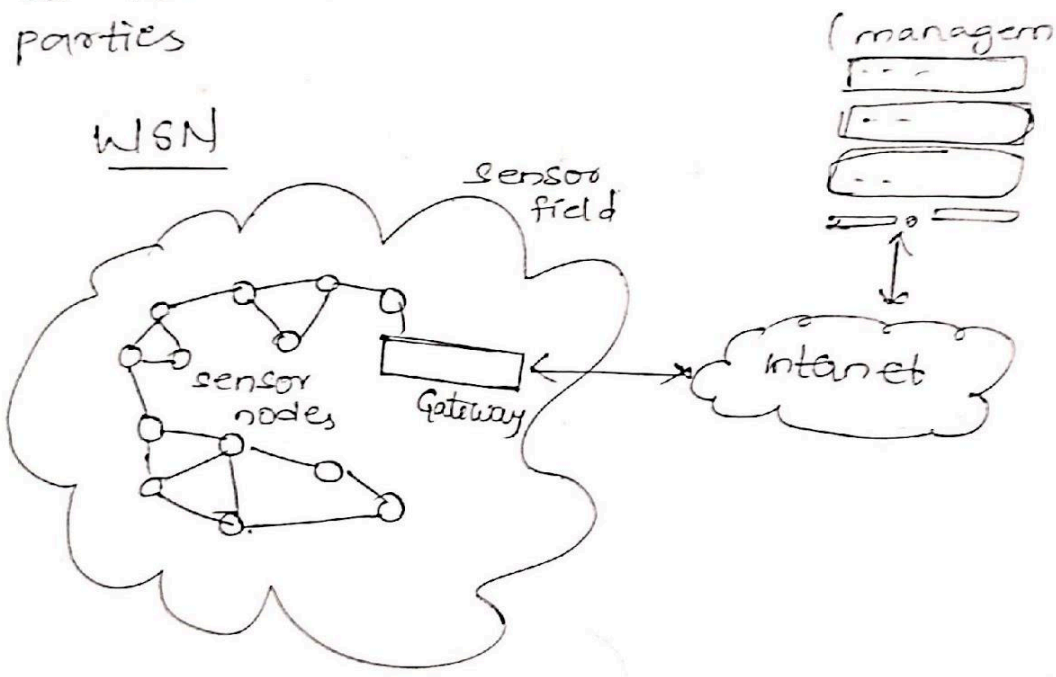
* WSNs are N/Ws of small, low-power devices that are used to monitor and collect data from the surrounding environment. In order to transmit data from the internet to a WSN, a gateway is typically used to connect the WSN to the internet.

* There are several different ways that the internet can communicate with a WSN:

*Remote management: The gateway can be used to remotely manage and control the sensor nodes in the WSN. for eg:- it can be used to update the firmware on the nodes, or to configure the sensor parameters.

* Data collection: The gateway can be used to collect data from the sensor nodes in the WSN and transmit it to the internet for analysis and processing. This can be done using a variety of protocols, such as HTTP, MQTT or CoAP

* Command and control - The gateway can be used to transmit commands from the internet to the WSN, allowing remote users to control the sensor nodes and collect data on demand.
* Alerts and notifications - The gateway can be used to transmit alerts and notifications from the WSN to the internet in real-time. for eg:- if a sensor node detects a problem or anomaly, the gateway can transmit an alert to the internet to notify the appropriate parties



* A single-node architecture in a WSN typically consists of a sensor node that includes the following "hardware components".

CONTROLLER : A microcontroller or a Microprocessor that is responsible for processing data, making decisions, and controlling the other components of the node.

SENSORS/ACTUATORS : Devices that collect data from the environment and/or control physical processes. They can include temperature sensors, humidity sensors, light sensors, accelerometers etc.,

MONICATION DEVICE: A Wireless transceiver, such as a Zigbee or RF module, that enables the node to communicate with other nodes or a central control unit.

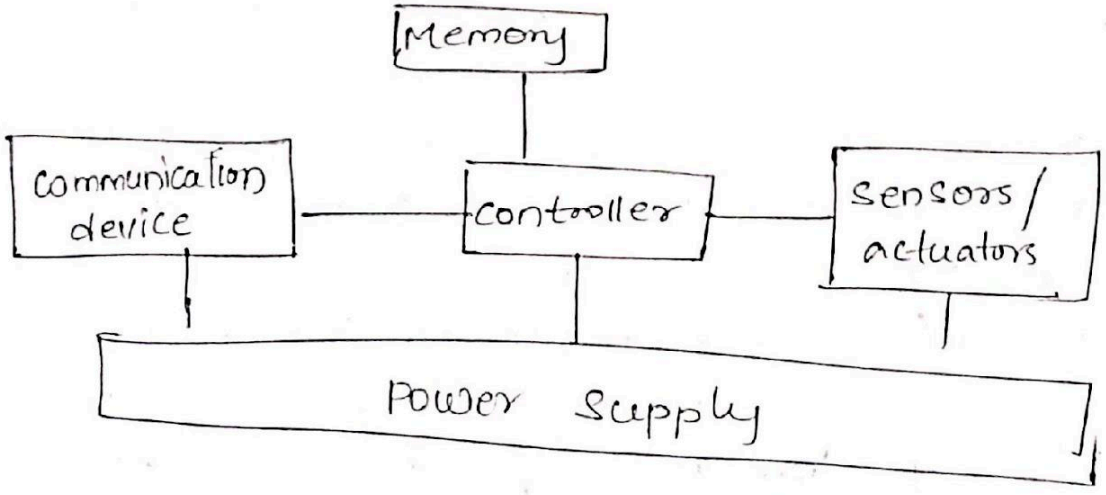
MEMORY: A non-volatile memory device, such as flash memory or an EEPROM, that stores data and programs.

POWER SUPPLY: A battery or other power source that provide power to the node.

DESIGN CONSTRAINTS:

* POWER CONSUMPTION - limited battery life is a major constraint, so power efficiency is critical.
* Cost: sensor nodes must be inexpensive to manufacture and deploy in large numbers.
* SIZE and WEIGHT: Sensor nodes must be small and lightweight for easy deployment and mobility.

SINGLE NODE ARCHITECTURE



Sensor node Hardware Components.

Attacks in the Application layer:

Attacks in this layer have the knowledge of data semantics, and thus can manipulate the data to change the semantics. As the result, false data are presented to applications and lead to abnormal actions. In this section, the following attacks will be discussed.

- clock skewing
- Selective Message forwarding
- Data Aggregation Distortion.

clock skewing - The targets of this attack are those sensors in need of synchronized operations (eg: [59][60][61]). By disseminating false timing information the attacks aim to desynchronize the sensors (i.e. skew their clocks).

For example, in IEEE 802.11 (which can be applied to WSNs), nodes are required to be synchronized with the access point. Beacon packets are broadcasted by the access point periodically. The packets contain timing information to be used by nodes for clock adjustment. Attackers can send false beacon packets with wrong timing information. once nodes adjust their clocks based on the wrong information, they will be out of synchronization with the access point. Although true beacon packets later can bring them back to synchronization, the nodes will oscillate between the two states and be unstable

Slective Message forwarding - for this attack, the adversary has to be on the path between the source and the destination,

.nd is thus responsible for forwarding packet for the source. The attack can be launched by forwarding some or partial messages selectively but not others. Note that the attack is different from the other selective forwarding attack in the network layer. To launch the selective forwarding attack in the application layer, attackers need to understand the semantics of the payload of the application layer packets (i.e treat each packet as a meaningful message instead of a monolithic unit), and select the packets to be forwarded based on the semantics. In comparision, the selective forwarding attack in the network layer only requires attackers to know the network information, such as the source and destination address. Attackers decide whether to forward packets according to those kinds of information only, and therefore operate at coarse granularity.

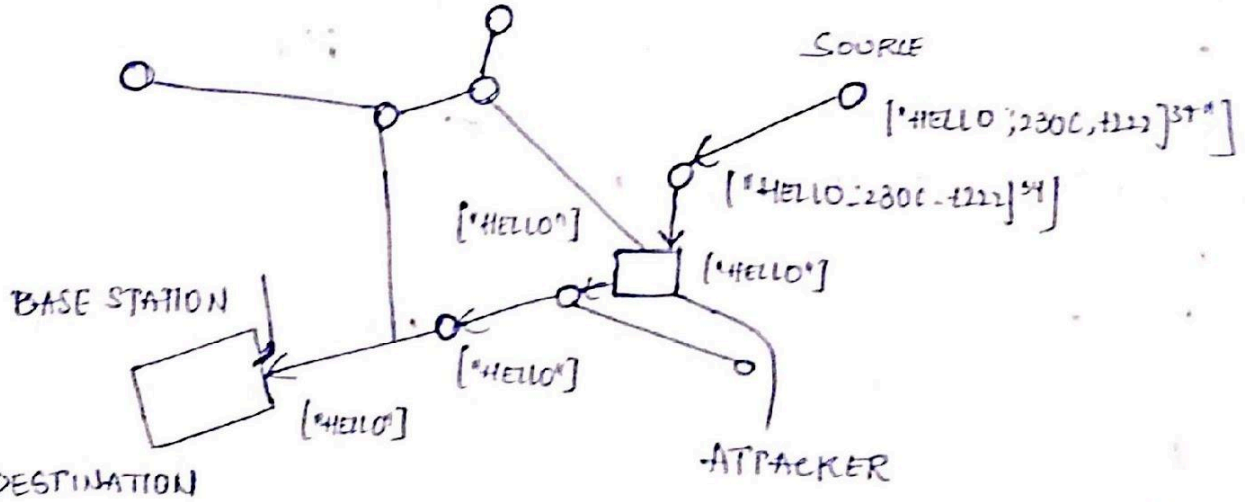


FIGURE: A SELECTIVE MESSAGE FORWARDING EXAMPLE.

Data Aggregation Distortion - once data is collected, sensors usually send it back to base stations for processing. Attackers may maliciously modify the data to be aggregated and make the final aggregation results computed by the base stations distorted. Consequently, the base stations will have an incorrect view of the environment monitored by the sensors, and may take inappropriate actions.

* Tinyos is an open-source operating system designed specifically for WSNs and other embedded systems. It is written in the C programming language and is designed to be lightweight and energy-efficient, making it well-suited for use on small, low-power devices.

SOME KEY FEATURES OF TINYOS INCLUDE:

* MODULAR ARCHITECTURE - TinyOS is designed to be modular, with a wide range of reusable components that can be easily combined to create custom applications.
* EVENT-DRIVEN MODEL - Tinyos uses an event-driven programming model, where tasks are triggered by events such as the arrival of a packet or the expiration of a timer. This can help to minimize energy consumption by allowing the system to sleep between events.
* SUPPORT FOR A WIDE RANGE OF HARDWARE - TinyOS includes support for a wide range of hardware platforms, including microcontrollers, sensors, and radio transceivers.

NETWORKING STACK - TinyOS includes a fully-featured networking stack that supports a wide range of networking protocols, including IPV4, IPV6 and 6LoWPAN.

* DEVELOPMENT TOOLS - TinyOS includes a range of development tools, including a simulator, a debugger, and a code profiler, to help developers create and debug applications.

* overall, TinyOS is a popular choice for developing applications for WSNs and other embedded systems due to its lightweight design, modular architecture, and support for a wide range of hardware platforms.

* WSNs are networks of small, low-power devices that are used to monitor and collect data from the surrounding environment. These devices typically have limited processing power and memory and as a result, they require specialized operating systems and execution environments that are optimized for their unique constraints.

* Some examples of operating systems and execution environments that are commonly used in WSNs include:

- TinyOS - TinyOS is an open-source operating system designed specifically for WSNs. it is written in the nesC programming language and is designed to be lightweight and energy efficient.
- contiki - contiki is another open-source operating system for WSNs. It is written in the C-programming language and includes support for a wide range of networking protocols and applications

- RIOT - RIOT is an open-source operating system for the internet of things (IOT) and WSNs. it is written in the C programming language and is designed to be lightweight and energy efficient.
- JAVA EMBEDDED - java Embedded is a java execution environment that can be used to run java applications on WSN devices. It is designed to be lightweight and energy-efficient, and includes support for a wide range of java libraries and tools for working with sensors and other hardware.