Q1. Attempt *any three* of the Following.

a. Define and explain the Internet of Things and Ubiquitous Computing.

Ans: Internet of Things is defined as the paradigm which is equipped with sensors, actuators and processor that communicate with each other to serve a meaningful purpose. Sensor: It is a module, component whose purpose is to detect event or changes to the environment and sends the information to other electronics, frequently a computer processor. Actuator: It is a component of a machine which is responsible for moving and controlling of mechanism or system.

The name "Internet" Of "Things" come from the fact that the Internet was used to send, receive or communicate information by the object, a Thing. The Things in IoT is present physically in the real world in your home, your work, your car, worn around your body. In IoT, things receive input from the world using Sensor and transform those data which is sent onto the Internet for collection and processing and then output is produced using actuators.

Example: The umbrella has a retractable canopy and the handle to hold it. We can pair a smartphone with a normal "dumb" umbrella, by checking an app to see whether it is likely to rain later before you leave the house. Unlike a calm, subtle light in the umbrella stand, glimpsed from the corner of your eye as an ambient piece of information to process subconsciously when you pass it on the way out of your home, an app requires you to perform several actions.

If you are able to establish and maintain the habit of doing this check, it will be just as effective. Rather than having greater capabilities, the smart umbrella simply moves the same intelligence into your environment so that you don't have to change your routine.

A bus display has to be readable to public transport users, including the elderly and partially sighted and able to survive poor weather conditions and risk of vandalism. In your kitchen, a blinking light reminds you it's a time to take your tablet, If you forget, the medicine bottle goes. online and email the doctor to let her know that you have missed your medicine.

The sports bracelet is easy to wear while running, has a display that is large enough and bright enough to read even when you are moving and will survive heat, cold, sweat and rain.

The Internet of Things suggests that rather than having a small number of the very powerful computing device in our life like a laptop, tablet, phone, music player, you might have a large number of devices which are perhaps less powerful like an umbrella, bracelet, mirror, fridge, shoes. The equation for the Internet of Thing are as follow:

Physical Object

+

Controller, sensor and Actuators

+

Internet

=

Internet Of Things.

1. Ubiquitous computing (or "ubicomp") is a concept in software engineering and computer science where computing is made to appear anytime and everywhere. This paradigm is also described as pervasive computing, ambient intelligence or "everywhere"

2. In contrast to desktop computing, ubiquitous computing can occur using any device, in any location, and in any format.

3. A user interacts with the computer, which can exist in many different forms, including laptop computers, tablets and terminals in everyday objects such as a refrigerator or a pair of glasses.

4. The underlying technologies to support ubiquitous computing include Internet, advanced middleware, operating system, mobile code, sensors, microprocessors, new I/O and user interfaces, networks, mobile protocols, location and positioning, and new materials.

5. Each term emphasizes slightly different aspects. When primarily concerning the objects involved, it is also known as physical computing, the Internet of Things, haptic computing and "things that think".

6. Ubiquitous computing touches on a wide range of research topics, including distributed computing, mobile computing, location computing, mobile networking, context-aware computing, sensor networks, human-computer interaction, and artificial intelligence.

7.Application of ubicomp in IOT: (IoT) infographic includes the smart alarm clock and coffee pot connected to the work scheduler that is alerted to let the user sleep an extra 30 minutes even though there is an accident on the driving route to work, the car knows it needs gas and the train is off its schedule.

b. Any sufficiently advanced technology is indistinguishable from magic, Discuss.

Ans: The objects in folktales and fairy tales are often wish-fulfilment fantasies to fill the deepest desires: if only I had enough to eat; if only my mother was well again; if only I could talk to my friend even though I’m far away; if only I could get home; if only I didn’t have to work every hour of the day to earn enough money for my family to eat.

Literary and anthropological scholars have long studied fairy tales for the lessons that can be learnt about the basic rules of human narrative and meaning and have analyzed the characters, storylines, and objects found within them.

From the point of view of a Silicon Valley entrepreneur and technologist, David Rose has talked about Enchanted Objects and has categorised various objects drawn from fairy tales and fantasy literature in ways that apply as much to technological objects.

For Protection, just as magical swords and helmets protected the protagonists of fairy tales from their enemies, so has much of the development of science and technology throughout history been driven by the need for military superiority, for the purpose of security or conquest.

Human Connection, even when one’s loved ones are far away, is an urgent, aching need: the Finnish hero Lemminkäinen’s family know that he has been hurt when the enchanted comb that he left on the mantelpiece starts to bleed.

Similarly, the postal service, telephones, and social networking help keep us in touch with our family and friends. The need for Creative Expression is fulfilled in stories by the enchanted paintbrushes or magic flutes and harps, while we have always used technology to devise such creative outlets, from charcoal to paint to computer graphics, or from drums to violins and electronic synthesisers.

So, technology has always been associated with magic, and so this will be true almost by default for the Internet of Things.

But there is more to it than that: a key element of many enchanted objects is that above and beyond their practical enchantment they are given a name and a personality- —implying an intelligence greater than strictly necessary to carry out the task for which they are designed.

Just as these enchanted mills, swords, and rings are capable of more than just their functional specification, so our connected devices, or Things, have processing and communicating capabilities well beyond the needs of the average lamp, umbrella, or bubble machine.

c. What is Calm and ambient Technology ? Explain With Example.

Ans: ● Calm Technology means the systems which don’t vie for attention yet are ready to provide utility or useful information when we decide to give them some attention.

● Information from the technology smoothly shifts to the user's attention when needed but otherwise stays calmly in the user's periphery. Mark Weiser describes calm technology as "that which informs but doesn't demand our focus or attention".

● The use of calm technology is paired with ubiquitous computing as a way to minimize the perceptible invasiveness of computers in everyday life.

● With its focus on computing power being embedded everywhere, ubicomp or ubiquitous computing is often also referred to as ambient computing.

● However, the term “ambient” also has connotations of being merely in the background, not something to which we actively pay attention and in some cases as something which we seek to remove (e.g., ambient noise in a sound recording).

● This technology aims to reduce the "excitement" of information overload by letting the user select what information is at the centre of their attention and what information is peripheral.

● Calm and ambient technology allows more information to existing there, ready for selection when needed.

● An example: a video conference maybe a calmer interface than a phone conference because the explicit visual knowledge of details that are peripheral gives participants more confidence in what can be focused on and what can be left at the edge. On the other side, Phone conferences in which participants are never quite sure who has entered or left the room at the other end, this lack of information is not necessarily calming. Knowledge of the periphery gives us "located Ness" without unduly distracting us.

● As devices with embedded programming become an all-pervasive part of our environment the ability to design calming devices and environments is apt to become much more important.

d. "Be conservative in what you do, be liberal in what you accept from others" Discuss.

Ans: In, computing the Robustness principle is a design guideline for software: Jon Postel wrote Be conservative in what you do, be liberal in what you accept from others (often reworded as "Be conservative in what you send, be liberal in what you accept").

The principle is also known as Postel's Law, after who wrote in an early specification of TCP. TCP implementations should follow a general principle of robustness: be conservative in what you do, be liberal in what you accept from others. It is good to bear this in mind when designing or building anything which must interact with other services particularly when you aren't the one building the other components with which your system interacts. In other words, programs that send messages to other machines (or to other programs on the same machine) should conform completely to the specifications, but programs that receive messages should accept known conformant input as long as the meaning is clear. Among programmers, to produce compatible function, the principle is popularized in the form be contravariant in input type and covariant in the output type.

Postel's principle by recommending that programmers "assume that the network is filled with malevolent entities that will send in packets designed to have the worst possible effect". Protocols should allow for the addition of new codes for existing fields in future versions of protocols by accepting messages with unknown codes (possibly logging them).

Programmers should avoid sending messages with "legal but obscure protocol features" that might expose deficiencies in receivers, and design their code "not just to survive other misbehaving hosts, but also to cooperate to limit the amount of disruption such hosts can cause to the shared communication facility" In 2001, Marshall Rose characterized several deployment problems when applying Postel's principle in the design of a new application protocol.

For example, a defective implementation that sends non-conforming messages might be used only with implementations that tolerate those deviations from the specification until, possibly several years later, it is connected with a less tolerant application that rejects its messages.

In such a situation, identifying the problem is often difficult, and deploying a solution can be costly. Rose, therefore, recommended, "explicit consistency checks in a protocol ... even if they impose implementation overhead". In an Internet-Draft of 2017, Martin Thomson argues that Postel's robustness principle actually leads to a lack of robustness, including security.

In a published paper to the annual Privacy Enhancing Technologies Symposium (PETS), Florentin Rochet and Olivier Pereira show how to exploit Postel's robustness principle inside the Tor routing protocol to compromise the anonymity of onion services and Tor clients.

e. Define protocol. Explain the following application layer protocols:

a) HTTPS b)SMTP c)FTP d)POP3 e)IMAP

Ans: ● A protocol is a set of rules for communication between computers. It includes rules about how to initiate the conversation and what format the messages should be in.

● It determines what inputs are understood and what output is transmitted. It also specifies how the messages are sent and authenticated and how to handle errors caused by transmission.

a) HTTPS: If someone eavesdrop your connection with tools like WireShark then that person can easily read the conversation. In fact , it isn’t the format of the protocol that is the problem even if the conversation happened in binary, an attacker can write a tool to translate the format into something readable.

b) SMTP: It’s set of communication guidelines that allow the software to transmit any electronic mail over the internet is called Simple Mail Transfer Protocol. It’s the program use for sending messages to other computer users based on email addresses.

c) FTP: The File Transfer Protocol is standard network protocol use for transfer of computers file between client and server on computer network.

d) POP3: Post Office Protocol 3 is the most recent version of standard protocol for receiving e-mail. POP3 is client/server protocol in which email received and held by internet server. POP3 is designed to delete mail on the server as soon as user has downloaded it.

e) IMAP: Internet Message Access Protocol is an internet standard protocol used by email clients to retrieve email messages from mail server over a TCP/IP connection. IMAP is defined by RFC 3501.

f. Discuss the following IOT device use at Does Liverpool.

a) Central Heating System. b) Doorbot

Ans:

LIVERPOOL:

● DOES Liverpool is a community of people with a diverse range of skills and interests.

● DOES workshop is well equipped with a range of electronics equipment, 3D printers, vacuum former, CNC mill and laser cutters.

● Some IoT devices which are used at DOES Liverpool are Central heating system and Doorbot.

CENTRAL HEATING SYSTEM:

● DOES Liverpool, the central heating system has been hooked up to the internet.

● YAHMS, as the system name, consist of a collection of sensors to measure temperature in the office and outside, an actuator to turn the heating on or off and some server software to manage timer control and provide a webbased interface to the system.

● Like many non-Interconnected heating systems, there is a timer-based program which ensures a basic level of comfort automatically.

● However, users can log to the YAHMS Website to find out what the temperature is and decide to turn the heating on or off to override the program.

● As it is web-based, it works equally well if you are sitting in the office at the time or if you are at home getting ready to head into work on a cold winter Saturday.

DOORBOT:

● It originally consisted of a networked Pc with a flat-screen monitor facing out towards the corridor through a conveniently located window.

● The Doorbot works as a Kiosk device, showing webcam views of the office, a list of coming events and a welcome message to any expected guests.

● Currently, its only input device is an RFID reader.

● Our members can register their RFID cards.

● Finally, this device is also connected to speakers, so it can play a personalized tune or message when members check-in or out.

● The computer that fits neatly with an integrated screen might work, such as iMac, a laptop or a tablet.

● These devices are much more expensive than original commodity Pc.

- A small embedded computer, such as a Raspberry Pi, might be ideal because it costs relatively little, runs Linux and has HDMI output.

Q2. Attempt *any three* of the Following.

a. How can we decide between the Cost and Ease of Prototyping.

Ans: The cost of prototyping a connected product is directly proportional to the complexity of the solution and the duration of the discovery phase. The Internet of Things is a multi-level system where smart devices communicate with each other and with the server and are managed via mobile and web apps.

Top 3 Factors Affecting IoT Prototyping Costs:

1. Proof of Concept & Product: An IOT development, Proof of Concept (POC) is the evidence that there are no technical limitations to turning a novel concept into a fully-fledged product which, among other things, has strong commercial appeal.

It should be noted that 70% of IoT projects stall at the Proof of Concept stage. The relatively high failure rate can be largely attributed to insufficient research into the Internet of Things impact on business, technical expertise and project management issues.

2. Hardware Prototyping: The entire prototype development phase Printed Circuit Board (PCB design and hardware production cost included) could be estimated at $25-40 thousand. At the early stages of IOT product development, you can also use off-the-shelf microcontrollers, sensors and enclosure. The benefits of such an approach include flexibility, efficiency.

3. Software Development: The Internet of Things is a multi-level system, but it is the software tier where the real magic happens. According to firmware development will cost you anything between $10 thousand and $20 thousand. IOT

b. What are the merits and demerits of mixing open source and close source?

Ans: Open source and close source software development used to be competing strategies. Now software firms are experimenting with strategies that mix the two models.

Merits:

- Software companies are taking a "best of both worlds" approach by creating products that use a combination of OS and proprietary software code.
- The researchers wanted to get a clearer sense of when a profit-maximizing firm should adopt a mixed-source business model and what that model might look like under different circumstances.
- Results indicate recurring patterns and strategies that managers can take into consideration when setting strategy.

● Open-sourcing many of libraries and keeping core business closed.

● The benefits of mixing open source and close source are flexibility and scalability.

● It is easier to work with beginners or those who don’t know how to code.

● Less material waste.

● Increased strength and longer part life.

● Valuable cost savings.

Demerits:

● Customers of closed source software companies are more or less at the whim of where their software supplier wants to take them. To change vendors once their software has embedded within your enterprise is likely to be prohibitively expensive.

● Closed source software is closed to viewing. Users of this software unable to modify the code effectively.

● So many developments are going on at the same time it is hard to keep track of which version is the most up to date.

● Although the cost is low there may be still some indirect costs involved, such as paying for external support.

Example:

Adrian’s project Bubblino has a mix of licences:

● Arduino code is open source.

● Schematics are available but not well advertised.

● Server code is closed source. The server code was partly kept closed source because some details on the configuration of the internet of things device were possibly part of the commercial advantage.

c. Explain the transition from prototype to production.
Ans:

TRANSITION FROM PROTOTYPES TO PRODUCTION

•Prototyping is a major factor but also the biggest obstacle to start a project

• Scaling to build many projects, but building them on such a large amount brings many difficulties and challenges.

CHANGING EMBEDDED PLATFORM

• When we scale up we might think of changing platforms to cut down the cost or opting for different solutions.

• If we start with a free and powerful programming language we may reduce the size to bring compactness to make it cheaper but reducing size might be bought more challenges.
• This issue is something that we should be aware of, If the first prototype built on a PC, iPhone, or whatever has helped you get investment or collaborators, you may be well placed to go about replicating that compelling functionality on
your final target.

PHYSICAL PROTOTYPES AND MASS PERSONALISATION

Chances are that the production techniques that you use for the physical side of your device won't translate directly to mass production. However, while
the technique might change—injection moulding in place of 3D printing, for
example—in most cases, it won't change what is possible.
An aspect that may be of interest is in the way that digital fabrication tools
can allow each item to be slightly different, letting you personalise each
the device in some way. There are challenges in scaling this to production, as
you will need to keep producing the changeable parts in quantities of one,
but mass personalization, as the approach is called, means you can offer
something unique with the accompanying potential to charge a premium.

d. With a help of an example explain the process of Scaling up the electronics

Ans: ● Scaling up electronics is the process where a circuit is built from initial testing through prototype to a finished PCB.

● The starting point for prototyping is usually a Breadboard. This lets u push-fit components and wires to make up circuits without requiring any soldering and therefore makes experimentation easy.

● When we are happy with how things are wired up its easy to solder the components, which may be sufficient to make the circuit more permanent and prevent wires from getting away.

For example:

● Let us consider an evolution of the part of Bubblino circuit, from initial testing through prototype, to finished PCB.

1. The first step in creating a circuit is generally to build it up on a breadboard, this way we can easily reconfigure things as we decide exactly how it should be laid out

. 2. When we are happy with how the circuit works, soldering it onto a stripboard will make the layout permanent. This means you can stop worrying about any wire getting loose, and if we are going to make only one copy of the circuit, then it might be as far as we need to take things.

3. If we need to make copies of the circuit, or if we want a professional finish, we can turn our circuit into a PCB, this makes it easier to build up the circuit because the position of each component will be labelled, there will be holes only where the components go, and there will be

less chance of short circuits because the track between components will be protected by the solder resist.

● When you want to scale things even further moving to combined board allows us to remove any unnecessary components from the microcontroller board.

e. Explain the following with respect to prototyping device : Processor speed , RAM, Networking , USB , Power Consumption and physical size and form factor.

Ans:

<u>Processor Speed: -</u>

● The processor speed, or clock speed of processor tell us how fast it can process the individual instructions in the machine code for the program it's running.

● A faster processor speed means that it can execute instructions more quickly.

● The clock speed is the simplest proxy for raw computing power.

● Comparison is made based upon millions of instructions per second, depending on what numbers are being reported in the datasheet or specification for the platforms being compared.

● Generally, one will use the processor speed as one of a number of factors when weighing up similar systems.

● Microcontroller tend to be clocked at speeds in the tens of MHz, but SoC's run at hundreds of MHz or possibly low GHz.

<u>RAM:</u> -

● RAM provides the working memory for the system.

● Pc with more RAM can do more things or can have flexibility over choice of coding algorithm.

● Space requirements are based upon the size of datasets needs to be handled.

● The RAM required for a device is not exact as it varies from project to project.

● Microcontroller with less than 1KB of RAM are of no interest.

● If one has to run standard encryption protocol , you will need at least 4 KB and more .

● For linux operating system, we recommend at least 256MB.

<u>Networking</u>: -

● How a device connects to the rest of the world is a key consideration for Internet of Things products.

● Wired ethernet is the simplest and cheapest for user , but it requires a physical cable.

● Wireless solutions avoid physical cable requirements but introduce a more complicated configuration.

● WiFi is most widely deployed to provide an existing infrastructure for connections, but can be more expensive and less optimized for power consumption.

● Other short-range wireless can offer better power consumption or cost than WiFi but lower bandwidth.

● For low bandwidth, high latency communication one can use SMS ; for higher data rates use same data connection like 3G as a. Smartphone.

USB: -

● If a device can rely on a more powerful computer being nearby , tethering to it via USB can be an easy way to provide both power and networking.

● Some microcontroller in version include support for USB, which reduce the need for extra chip in circuit.

● Instead of microcontroller presenting itself as a device, some can also act as USB "host".

● USB host lets you connect items that would normally except to be connected to a computer like phones.

Power Consumption: -

● Faster processor often consumes more power than slower ones.

● Device which might be portable or rely on an unconventional power supply, power consumption may be an issue.

● Processors have a minimal power consumption sleep mode which allows you to use a faster processor to quickly perform operations and then return to low power sleep.

● Hence, a more powerful processor may not be a disadvantage even in a low power embedded device.

Physical size and form factor: -

● Nowadays , the size of chip is governed by the number of connections it needs to make to the surrounding components on the PCB.

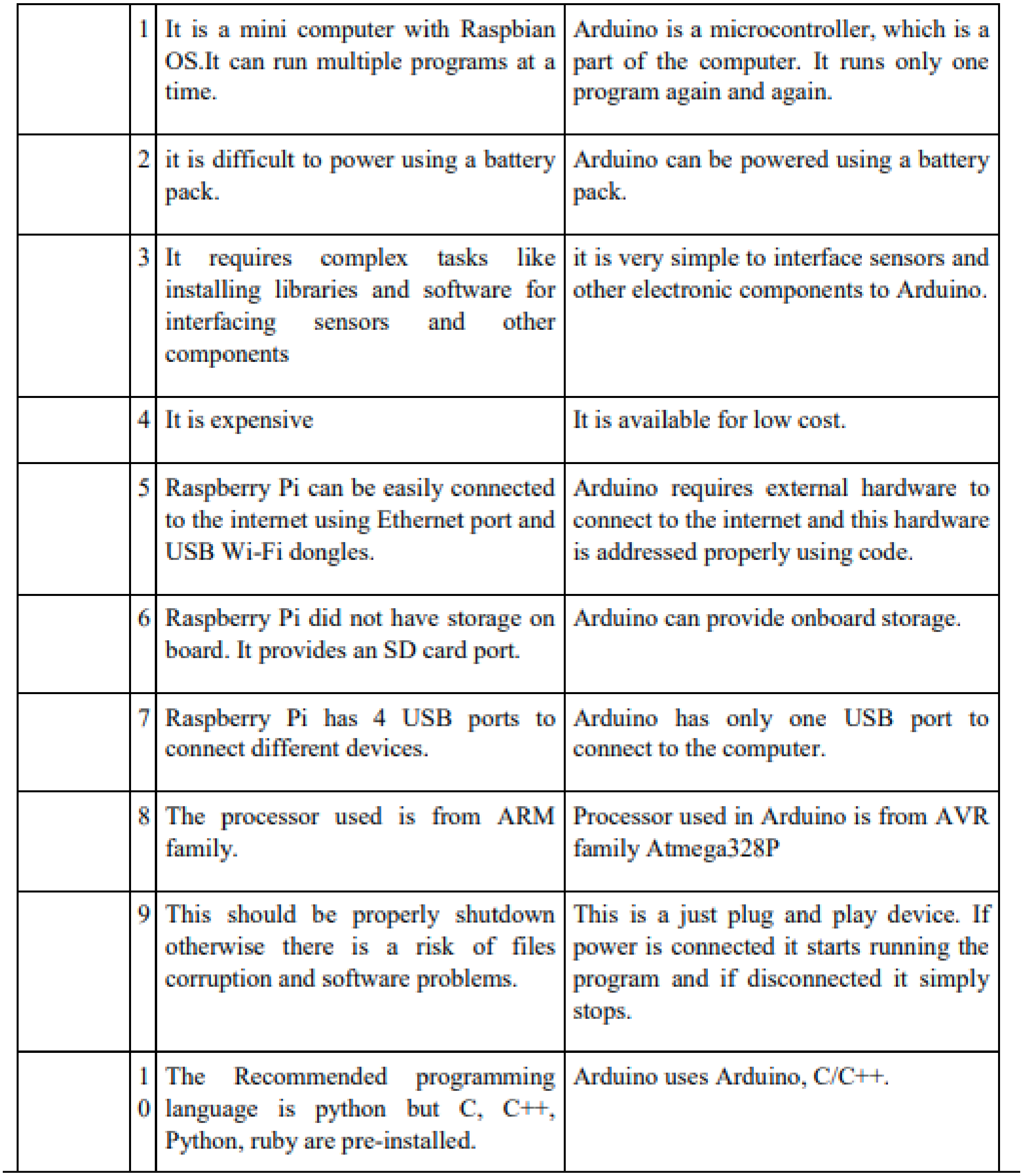
● With the traditional through- hole design , most commonly used for home made circuits , the legs of chip are usually spaced at 0.1" intervals.

● More complex chips can easily run to over a hundred connections.

● Due to these trade-off in size versus manufacturing complexity, many chip design are available in a number of different form factors, know as packages.

f. Compare Raspberry Pi and Aurdino.

Ans: Raspberry PI Aurdino



	1	It is a mini computer with Raspbian OS.It can run multiple programs at a time.	Arduino is a microcontroller, which is a part of the computer. It runs only one program again and again.
	2	it is difficult to power using a battery pack.	Arduino can be powered using a battery pack.
	3	It requires complex tasks like installing libraries and software for interfacing sensors and other components	it is very simple to interface sensors and other electronic components to Arduino.
	4	It is expensive	It is available for low cost.
	5	Raspberry Pi can be easily connected to the internet using Ethernet port and USB Wi-Fi dongles.	Arduino requires external hardware to connect to the internet and this hardware is addressed properly using code.
	6	Raspberry Pi did not have storage on board. It provides an SD card port.	Arduino can provide onboard storage.
	7	Raspberry Pi has 4 USB ports to connect different devices.	Arduino has only one USB port to connect to the computer.
	8	The processor used is from ARM family.	Processor used in Arduino is from AVR family Atmega328P
	9	This should be properly shutdown otherwise there is a risk of files corruption and software problems.	This is a just plug and play device. If power is connected it starts running the program and if disconnected it simply stops.
	10	The Recommended programming language is python but C, C++, Python, ruby are pre-installed.	Arduino uses Arduino, C/C++.

Q3. Attempt *any three* of the Following.

a. Define business model. Explain different factors in the definition.

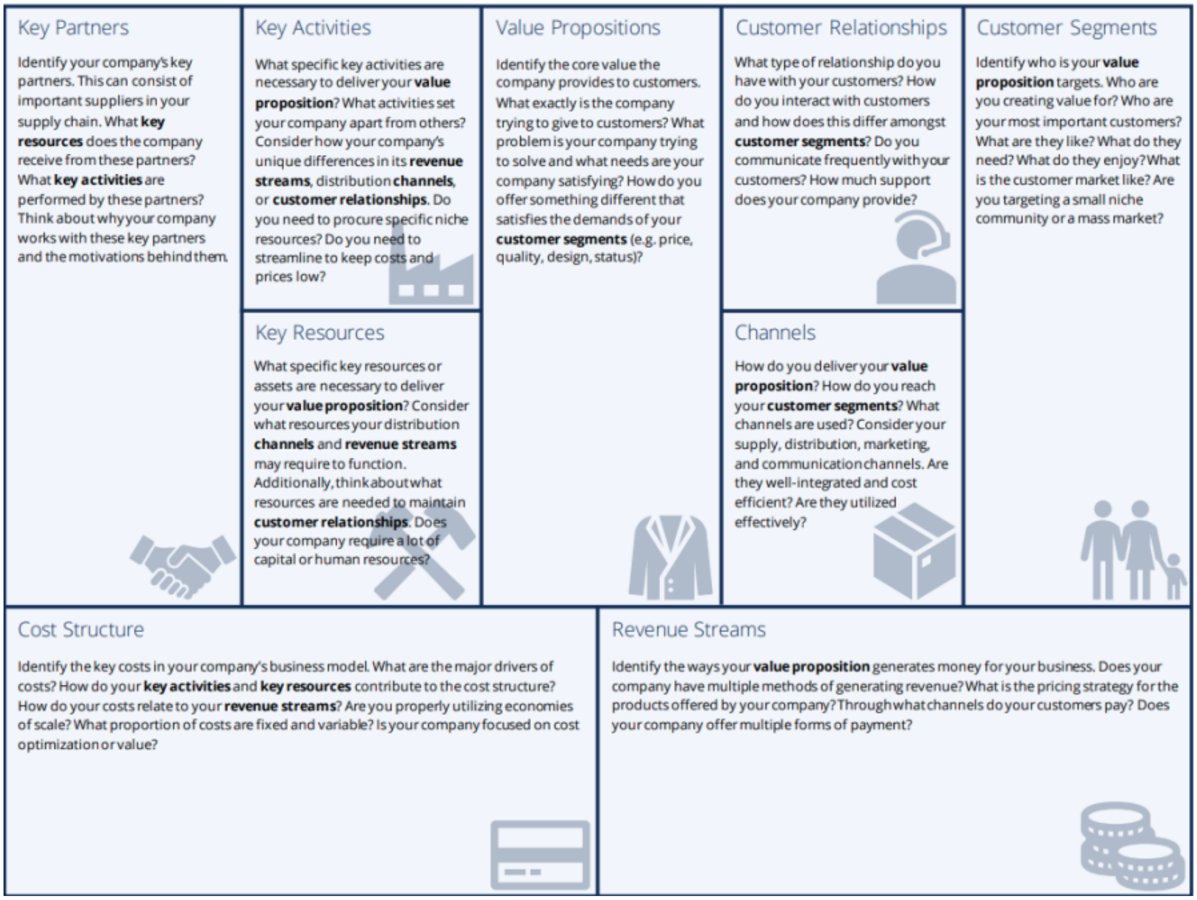
Business Model describes the rationale of how organization creates, delivers , capture value in economic, social , cultural or other contexts And the process of business model construct and modify is also called business model innovation. The business model definition brings together a number of factors:

- A group of people(customers).
- The need of those customer.
- A thing that your business can do to meet those needs.
- A success criterion such as making a profit
- Organisational practices that help to achieve this goal:
- And to be able to carry on doing so, sustainably.

(Incomplete Answer)

b. With the help of the diagram explain business model canvas.

Ans: One of the most popular templates for working on a business model is the Business Model Canvas by Alexander Osterwalder and his startup, the Business Model Foundry. The canvas is a Creative Commons–licensed single-page planner.



Key Partners

Identify your company's key partners. This can consist of important suppliers in your supply chain. What **key resources** does the company receive from these partners? What **key activities** are performed by these partners? Think about why your company works with these key partners and the motivations behind them.

Key Activities

What specific key activities are necessary to deliver your **value proposition**? What activities set your company apart from others? Consider how your company's unique differences in its **revenue streams**, distribution **channels**, or **customer relationships**. Do you need to procure specific niche resources? Do you need to streamline to keep costs and prices low?

Key Resources

What specific key resources or assets are necessary to deliver your **value proposition**? Consider what resources your distribution **channels** and **revenue streams** may require to function. Additionally, think about what resources are needed to maintain **customer relationships**. Does your company require a lot of capital or human resources?

Value Propositions

Identify the core value the company provides to customers. What exactly is the company trying to give to customers? What problem is your company trying to solve and what needs are your company satisfying? How do you offer something different that satisfies the demands of your **customer segments** (e.g. price, quality, design, status)?

Customer Relationships

What type of relationship do you have with your customers? How do you interact with customers and how does this differ amongst **customer segments**? Do you communicate frequently with your customers? How much support does your company provide?

Channels

How do you deliver your **value proposition**? How do you reach your **customer segments**? What channels are used? Consider your supply, distribution, marketing, and communication channels. Are they well-integrated and cost efficient? Are they utilized effectively?

Customer Segments

Identify who is your **value proposition** targets. Who are you creating value for? Who are your most important customers? What are they like? What do they need? What do they enjoy? What is the customer market like? Are you targeting a small niche community or a mass market?

Cost Structure

Identify the key costs in your company's business model. What are the major drivers of costs? How do your **key activities** and **key resources** contribute to the cost structure? How do your costs relate to your **revenue streams**? Are you properly utilizing economies of scale? What proportion of costs are fixed and variable? Is your company focused on cost optimization or value?

Revenue Streams

Identify the ways your **value proposition** generates money for your business. Does your company have multiple methods of generating revenue? What is the pricing strategy for the products offered by your company? Through what channels do your customers pay? Does your company offer multiple forms of payment?

At first sight, it looks as though each box is simply an element in a form and the whole thing could be replaced by a nine-point checklist. However, the boxes are designed to be a good size for sticky notes, emphasizing that you can play with the ideas you have and move them around. Also the layout gives a meaning and context to each item. Let's look at the model, starting with the most obvious elements and then drilling down into the grittier details that we might neglect without this kind of template. At the bottom right, we have Revenue Streams, which is more or less the question of "how are you going to make money?" we used to start this chapter. Although its position suggests that it is indeed one of the important desired outputs of the business, it is by no means the only consideration. The central box, Value Propositions, is, in plainer terms, what you will be Producing that is, your Internet of Things product, service, or platform. The Customer Segments are the people you plan to deliver the product to. That might be other makers and geeks. The Customer Relationships might involve a lasting communication between the company and its most passionate customers via social media. This position could convey an advantage but may be costly to maintain. Maintaining a "community" of your customers may be beneficial, but which relationships will you prioritise to keep communicating with your most valuable customer segments? Channels are ways of reaching the customer segments. From advertising and distributing your product, to delivery and after-sales, the channels you choose have to be relevant to your customers. On the left side, we have the things without which we have no product to sell. The Key Activities are the things that need to be done. The Thing needs to be manufactured; the code needs to be written. Perhaps you need a platform for it to run on and a design for the website and the physical product. Key Resources include the raw materials that you need to create the product but also the people who will help build it. The intellectual resources you have are also valuable, as are the finances required to pay for all this. Of course, few companies can afford the investment in time and money to do all the Key Activities themselves or even marshal all the Key Resources. We will need Key Partners, businesses that are better placed to supply specific skills or resources, because that is their business model, and they are geared up to do it more cheaply or better than you could do yourself. Perhaps you will get an agency to do your web design and use a global logistics firm to do your deliveries. Will you manufacture everything yourself or get a supplier to create components or even assemble the whole product? The Cost Structure requires you to put a price on the resources and activities you just defined. Which of them are most expensive? Given the costs you will have, this analysis also helps you determine whether you will be more cost driven (sell cheaply, and in great volume via automation and efficiency) or more value driven (sell a premium product at higher margins, but in smaller quantities).

c. Explain government funding for IoT projects.

Ans: It is important to plan our future costs and revenue for a well-planned business model, there would be a point where there would be only cost present and no profit. The main look out is on how to get rid initial funding and the ways in which you can get it. If you have enough money with you to invest in your Internet of Things startups as full time without working extra then we can fund our business ourselves. Keeping aside the criteria of investing in the project that has no success that has no success in forthcoming time we are the luckier one to have a good startup project and luckiest when we still have money to invest in material and staff. If we don't have enough funding that does not mean we cannot have a startup build but we need to try some other ways. Many people club their startup project with consulting business where they take up small project to have experience about the same ad also gain profit out of it. Government is always active in promoting new business and startups and they actively find for the same. Having all the funding schema known across the world is a project in itself. Government can set their own venture capital or have collaboration with interested companies funds are managed differently. Funding are been provided for new researches and innovation that are been done. The money that is been provided has many conditions attached which would be handled differently. ● Outputs: The deliverable that are been seen are the metrics based on which the government decides whether to fund or not. The owner has to write daily reports of achieve certain defined milestones in given schedule. If funding is been provided in stages you are expected to be submit parts that were predecided to be delivered. ● Spending constraints: You need to think twice before spending on the resources. The means just because you are been funded by the government you blindly can't spend on the resources you are expected to analyze the cost and then spend accordingly. It is perfectly normal for companies to work through multiple sources of funding.

d. How can we make optimum use of RAM while writing code for embedded devices?

Ans: ● Random-access memory (RAM) is a form of computer memory that can be read and changed in any order, typically used to store working data and machine code

● RAM contains multiplexing and demultiplexing circuitry, to connect the data lines to the addressed storage for reading or writing the entry.

● Usually more than one bit of storage is accessed by the same address, and RAM devices often have multiple data lines and are said to be "8-bit" or "16-bit", etc. devices.

● We may want to use as much of the memory as possible to provide more features.

● The way to achieve this result is to not allocate any memory dynamically, that is, while the program is running.

● To people coming from programming on larger systems, this concept is exotic. ● For example when we're downloading some information from the Internet how could we possibly know beforehand exactly how large it is going to be?

● The standard mechanism on desktop or server systems would be to allocate just enough memory at the time we're downloading things, when we know how much we'll need.

● In a deterministic model, we need to take a different tack.

● Rather than allocate space for the entire page, we set aside space to store the important information that we're going to extract and also a buffer of memory that we can use as a working area while we download and process the page.

● Rather than download the entire page into memory at once, we download it in chunks filling the buffer each time and then working through that chunk of data before moving on to the next one.

● An upside of this approach is that we are able to process pages which are much larger than we could otherwise process; that is, we can handle datasets which are bigger than the entire available memory for the system!

● The downside of this sort of single-pass parsing, however, is that we have no way to go back through the stream of data.

● After we discard the chunk we were working on, it's gone.

● We might cache the download to an SD card or other area of flash memory, where it could be processed in multiple passes without needing to read it all into RAM at once.

● If the format of the data we're consuming means that we know if something is needed only by the time we reach a later part of the file, we have to set aside space to save the potentially useful segment when we encounter it.

● Then when we reach the decision point, we still have it available to use and can discard it then if it's not required.

e. With the help of examples compare stack and heap?

Ans:

1. Stack:

i) New items which are added to the stack go on the top, and items can be removed only in strict reverse order, so the first thing to be removed is the last item that was placed onto the stack.

ii) This arrangement makes it easy for the processor to keep track of where things are and how much space is being used because it has to track only the top of the stack.

iii) Consequently, the stack is really only useful for (1) Items that aren't going to survive for long periods of time. (2) Items that remain in constant use, from the beginning to the end of the program.

iv) The downside to this approach is that if you're finished with a particular variable, you can release the memory used for it only when you can remove it from the stack, and you can do that only when everything added since it was allocated is removed from the stack, too.

v) Global variables, which are always available, are allocated first on the stack. After that, whenever the path of execution enters a function, variables declared within it are added. The parameters to the function get pushed onto the stack immediately, while the other variables are pushed as they are encountered. Because all the variables within a function are available only to code inside it, when you reach the end of that function, all those parameters and variables are ready to be discarded. So, the stack gets unwound back to the same size it was just before control passed to the function.

vi) Example,

// global variables

function A { variable A1; variable A2; call B(); } function B { variable B1; variable B2; variable B3; call C(); call D(); } function C { variable C1; } function D { variable D1; variable D2;} call A();

Before function A is called, the stack looks like state (i).

a) As execution moves into function A, its variables are added to the stack (ii).

b) Function A then calls function B, resulting in its variables being added to the stack (iii).

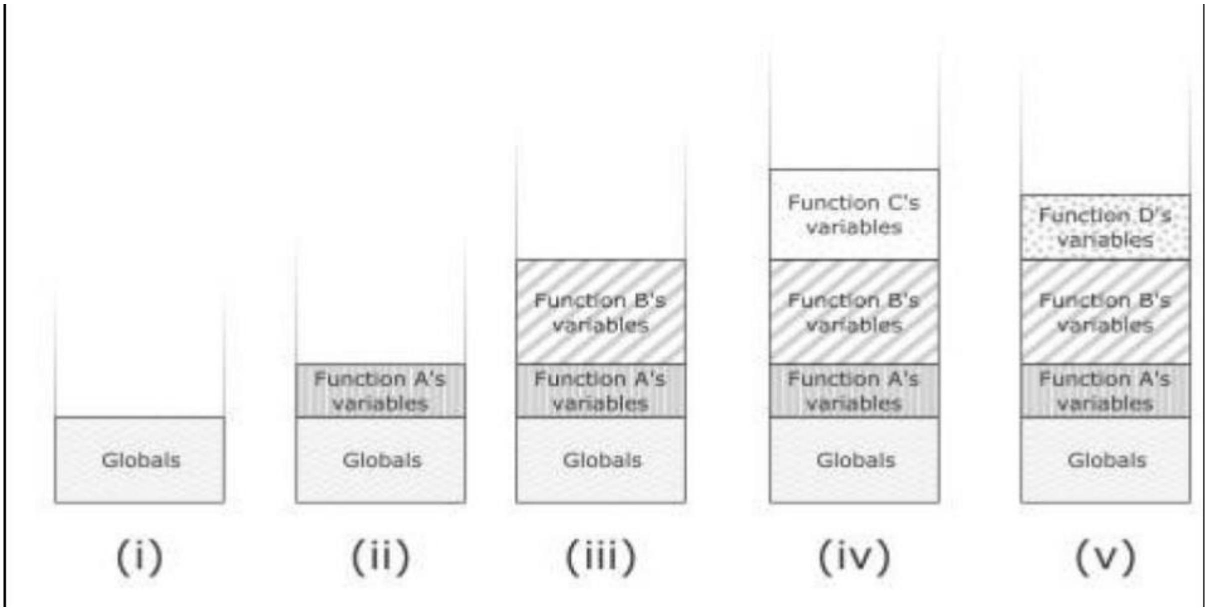
c) Inside function B, first function C is called, resulting in its variables being added to the stack (iv).

d) When execution returns from function C, its variables are removed from the stack, taking you back to stack (iii).

e) Then function D is called, so its variables are pushed onto the stack instead (v). f) Then execution returns to function B, with D's variables removed (iii).

g) And back to A, removing B's variables (ii).

h) And, finally, you leave function A, dropping back to just the global variables being defined (i).



2. Heap:

i) Heap allows the allocation of chunks of memory at any time.

ii) The heap is a bit like the seating area of a train where you fill up the seats strictly from the front and have to keep everyone who is travelling as a group in consecutive seats.

iii) To begin, all the seats are empty. As groups of people arrive, you direct them to the next available block of seats.

iv) But two possible problems exist,

> (1) First, you might simply have more people to fit on the train than there are seats (this is the same problem as running out of memory).
>
> (2) The second problem is more subtle: though you theoretically have enough free seats for the next group of passengers, those free seats are spread across the train and aren't available in a continuous block of free seats. This last situation is known as memory fragmentation.

v) Example, create object A (size 20 bytes) create object B (size 35 bytes) create object C (size 50 bytes) // do some work that needs object C delete object C create object D (size 18 bytes) // do more work with objects B and D delete object B create object E (size 22 bytes)

(1) At the start of execution, the heap will be empty (i).

(2) Object A is added to the heap (ii), taking up 20 bytes of space.

(3) Object B is added to the heap (iii), consuming a further 35 bytes straight after the space for object A.

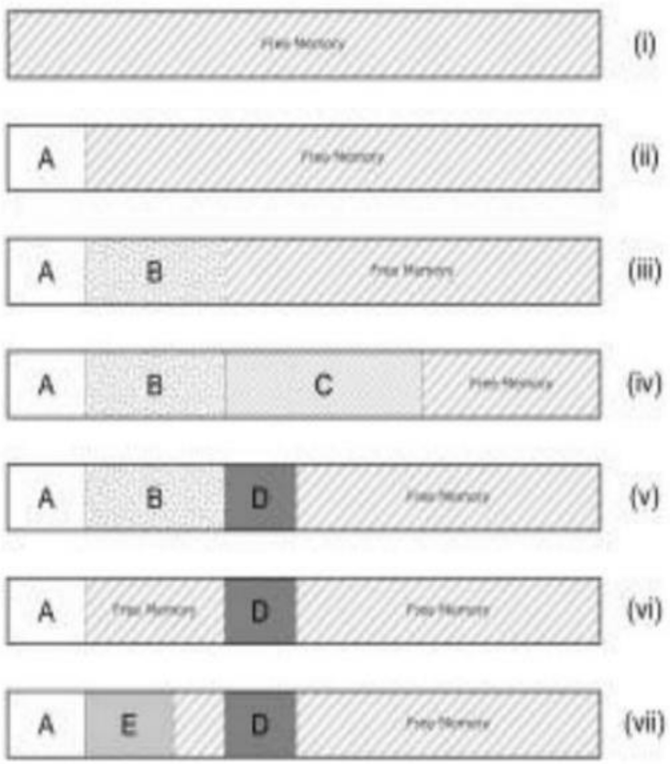
(4) Object C is added to the heap (iv), adding 50 bytes to the heap right after object B.

(5) Object C is no longer needed and is deleted, releasing the space it consumed on the heap and taking us back to heap (iii).

(6) Object D is created and takes up 18 bytes of the space just vacated by object C (v).

(7) Now object B is finished with and deleted. As other code might be relying on the position of object D, we can't move it, so there's now a free space between objects A and D (vi).

(8) Object E is created. It requires 22 bytes of space, which means it will fit in the hole left by object B (vii).



f. Explain in detail the process of debugging the code for embedded devices?

Ans: One of the most frustrating parts of writing software is knowing your code has a bug, but it's not at all obvious where that bug is. In embedded systems, this situation can be doubly frustrating because there tend to be fewer ways to inspect what is going on so that you can track down the issue.

Building devices for the Internet of Things complicates matters further by introducing both custom electronic circuits (which could be misbehaving or incorrectly designed) and communication with servers across a network.

Modern desktop integrated development environments (often shortened to IDEs) have excellent support for digging into what is going on while your code is running. You can set breakpoints which stop execution when a predefined set of conditions is met, at which point you can poke around in memory to see what it contains, evaluate expressions to see whether your assumptions are correct, and then step through the code line by line to watch what happens. You can even modify the contents of memory or variables on the fly to influence the rest of the code execution and in the more advanced systems rewrite the code while the program is stopped.

The debugging environment for embedded systems is usually more primitive. If your embedded platform is running a more fully featured operating system such as Linux, you are better placed than if you’re developing on a tiny microcontroller.

Systems such as embedded Linux usually have support for remote debugging with utilities such as gdb, the GNU debugger (www.gnu.org/software/gdb/). This utility allows you to attach the debugger from your desktop system to the embedded board, usually over a serial connection but sometimes also over an Ethernet or similar network link. Once it is attached, you then have access to a range of capabilities similar to desktop debugging— the ability to set breakpoints, single-step through code, and inspect variables and memory contents.

Another way to get access to desktop-grade debugging tools is to emulate your target platform on the desktop. Because you are then running the code on your desktop machine, you have access to the same capabilities as you would with a desktop application. The downside of this approach is that you aren’t running it on the exact hardware that it will operate on in the wild.

Q4. Attempt *any three* of the Following.

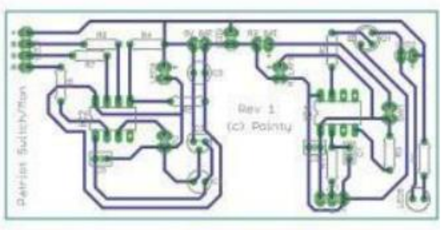
a. How are printed circuit board are designed? Explain.

Ans:

The designing of PCB is split between two main views: the schematic and the board.

● Step 1: Design the PCB circuit with a Software Draw the schematic circuit diagram with the PCB layout software such as CAD software, Eagle and Multisim software. This type of PCB design software contains a library of components that can be used to build the circuit. It is also possible to change the circuit design’s position and then to modify according to your convenience and requirement.

● Step 2:Film generation The film is generated from the finalized circuit board diagram of the PCB layout software which is send to the manufacturing unit where the negative image or mask is printed out on a plastic sheet.



● Step3: Select Raw Material The bulk of the printed circuit board is made with an unbreakable glass or fiberglass having copper foil bonded unto one or both the sides of the board. Thus, the PCBs made from unbreakable paper phenolic with a bonded copper foil are less expensive and are often used in household electrical devices.

● Step4: Preparing Drill Holes Machines and carbide drills are used to put holes on the printed circuit board. There are two types of machines available to drill the PCBs; they include hand machines and CNC machines. The hand machines require human intervention or effort to drill the holes, whereas CNC machines are computer-based machines that work-based on the machine timetables or programs that run both automatic as well as manually. The drilled pattern is stored in the computer like drill bit sizes, number of holes per panel, drilled stack, drilled time per load, etc. The PCB boards are placed into the CNC machine and the holes are drilled according to the determined pattern to place printed circuit board components.

● Step5: Apply Image The printed circuit layout can be printed in different ways on PCBs like manual pen, dry transfers, pen plotters and printers. The laser printers are a better way to print the layouts on printed circuit boards.

● Step6: Stripping and Etching This process involves removing the unwired copper on the PCBs by using different types of chemicals like ferric chloride, ammonium per-sulphate, etc. Make the solvent by mixing 1% of sodium hydroxide and 10 grams of sodium hydroxide pellets to one liter of water and mix it until everything is dissolved. Next, the PCB is put on a chemical bowl and cleaned up with a brush. during this process, if the PCB is still greasy, due to applied sunflower or seed oil, the developing process may take about 1 minute.

● Step7: Testing After finishing the manufacturing process of the Printed Circuit Board, the Board undergoes a testing process to check whether the PCB is working properly. Nowadays many automatic testing equipments are available for the high volume testing of the PCBs. The two different types of testing equipment available today that test your circuit boards include ATG test machines that are flying probe, fixtureless testers and in addition to a universal grid testing capability as well. This is all about the PCB designing process

b.Write a short note on mass-producing the case and other fixtures?

Ans:

● A good rule is of keep down cost production is to minimize amount of time a person has work on each item. Machines will be tend to cheaper than people.

● The most common method of mass production : injection moulding of plastic.

● The process involve injecting molten plastic will convert into desired shape.

● After the plastic has cooled sufficiently, the mould is separated out successfully and other parts of injection will collect in collection bin.

● The whole cycle is takes less time than 3-D printing, which means lot of parts can easily churned at lower cost per part.

● Case study : BERG's little printer

● The little printer, made by London design firm BERG.

● Initially it was called social letterbox. With the idea that it will be socially able to share printed thing among when family is not close to each other.

● Then printer is connecting to computer, the printer has its own network connection and intelligence. ● It is a two-stage process for part of the case. It is injection-moulded without the holes, which gives the correct finish, and then drilled to provide the holes.This process is expensive but it results in the higher quality finish.

● BERG cloud bridge, a box with both ZigBee wireless and Ethernet which you plug into an Ethernet port on your network to talk to Internet.

● Printer allows to communicate with BERG cloud server software, which provides for both the delivery of 'publication' to printer and the user-facing website.

c.What is the important of certification? Why it is required?

Ans: 1. One of the less obvious sides of creating an Internet of Things product is the issue of certification.

2. If you forget to make the PCB or write only half of the software for your device, it will be pretty obvious that things aren't finished when it doesn't work as intended.

3. Fail to meet the relevant certification or regulations, and your product will be similarly incomplete—but you might not realise that until you send it to a distributor, or worse still, after it is already on sale.

4. For the main part, these regulations are there for good reason.

5. They make the products you use day in, day out, safer for you to use; make sure that they work properly with complementary products from other suppliers and ensure that one product doesn't emit lots of unwanted electromagnetic radiation and interfere with the correct operation of other devices nearby.

6. You may not have noticed before, but if you take a closer look at any gadget that's near to hand, you will find a cluster of logos on it somewhere…CE, FCC, UL.… Each of these marks signifies a particular set of regulations and tests that the item has passed: the CE mark for meeting European standards; FCC for US Federal Communications Commission regulations; and UL for independent testing laboratory UL's tests.

7. The regulations that your device needs to pass vary depending on its exact functionality, target market (consumer, industrial, and so on), and the countries in which you expect to sell it.

8. Of particular interest is the electromagnetic compatibility, or EMC, testing.

9. This tests both how susceptible your device is to interference from other electronic devices, power surges on the main's electricity supply, and so on, and how much electromagnetic interference your product itself emits.

10. The problem arises when a circuit emits a sufficiently strong signal unintentionally which disrupts the desired radio frequencies. This is sometimes noticeable in the "dit, dit-dit-dit" picked up by a poorly insulated stereo just before your mobile phone starts ringing.

d.Explain privacy with respect to IOT devices in detail.

Ans: ● The Internet, as a massive open publishing platform, has been a disruptive force as regards the concept of privacy.

● As the Internet of Things is about Things, which are rooted in different contexts than computers, it makes uploading data more ubiquitous.

● Let's consider the mobile phone, in particular an Internet-connected phone with on-board camera.

● Although we do not typically consider phones as Internet of Things devices, the taking of a photo with a camera phone is a quintessential task for a Thing: whereas in the past we would have had to take a photo, develop it, take the printed photo to computer, scan it, and then upload it, now we can upload that compromising photo, in a single click, while still drunk.

● The ability to do something is present in a defined context rather than locked in a set of multiple processes, culminating in a general-purpose computer.

● Even innocuous photos can leak data.

● With GPS coordinates embedded into the picture's EXIF metadata, an analysis of our Flickr/Twitpic/Instagram feed can easily let an attacker infer where our house, our work, or even our children's school is.

● Even if we stripped out the data, photo-processing technology enables searching of similar photos, which may include these coordinates or other clues.

● Similar issues exist with sports-tracking data, whether produced by an actual Thing, such as Nike+ or a GPS watch, or a pseudo-Thing, like the Run Keeper app on our smartphone.

● This data is incredibly useful to keep track of our progress, and sharing your running maps, speed, heartbeat, and the like with friends may be motivating.

● But again, it may be trivial for an attacker to infer where our house is (probably near where you start and finish your run) and get information about the times of day that we are likely to be out of the house.

● When we tell family and friends about the Good Night Lamp or the Where Dial, they often bristle and start muttering about “Big Brother”.

● The idea of people knowing where we are can evoke strong emotions.

● Yet the idea of knowing that our loved ones are safe is a similarly deep-seated human emotion.

● To the extent that we allow our location to be shared with people we have chosen to share it with, there is no infringement of privacy.

● But the decision to give our mother a Good Night Lamp might seem less sensible months later when we arrive home late at night.

● Or we might regret giving your partner Where Dial if later she becomes jealous and suspicious of our innocent (or otherwise) movements.

● Even if these devices are themselves respectful of our privacy, their security or lack thereof might allow an attacker to get information.

e.Discuss the environmental issues associated with IOT devices.

Ans:

Some classical environmental concerns about the production and running of the Thing:

● Physical thing:-

> o Creating the object has a carbon cost, which may come from the raw materials used, the processes used to shape them into the shell, the packing materials, and the energy required to ship them from manufacturing plant to the customer.
>
> o It is easier than ever to add up the cost of these emissions: for example, using the ameeConnect API, you can find emissions data and carbon costs for the life-cycle use of different plastics you might use for 3D printing or injection molding. Calculating manufacturing cost for manufacture is harder.
>
> o You may need to consider other environmental factors, such as emissions produced during normal operation or during disposal of the object. For example, thermal printer paper may contain Bisphenol-A, which has health and environmental concerns.

● Electronics:-

> o The electronics contained in a Thing have their own environmental cost. Buying PCBs locally or from a foreign manufacturer affects the carbon cost of shipping the completed units.
>
> o Considering the potential cost savings, even a responsible manufacturer may find it reasonable to offset the extra carbon emissions.

o Many electronic components rely on “rare earth minerals” which have been extracted from China or other locations worldwide.

o If mining process is not managed properly then the slurries formed of mildly radioactive waste minerals will be left behind long after the mines cease production. Refining them involves the use of toxic acids. Shipping raw material from mine to refinery to manufacturer has its own carbon cost too.

o Every electronically enhanced Thing that you produce will incur these costs and will also need to be powered to run. o Speaking to internet is a major part of the power cost of running an IOT device. Anything that can be done to reduce this cost will make the device more efficient. o Choosing suppliers of Wi-Fi chips wisely and following the low-power IPv6 developments closely will be helpful here.

● Internet service:-

o As Nicholas Negroponte preaches, ”Move bits, not atoms”. In the digital world, moving data rather than physical objects is faster, is safer, and has a lower environmental cost. Running the internet has a cost: electricity to run the routers and the DNS lookups, plus establishing the infrastructure-laying cabling across the sea, setting up microwave or satellite links, and so on.

o As well as the cost of transferring the data over internet, running your own web server uses power.

o Many server hosting specialists now offer carbon-neutral hosting, where you pay extra to offset your emissions. Running inefficient code or services may cause higher power usage.

f.What is cautious optimism? Explain.

Ans:

● Between the person who dislikes technology and the one who has unquestioning positive attitude we prefer cautious optimism approach.

● The technology did change the world for worse but without that changes we wouldn’t have the world where magical objects can speak to us , to each other, and to powerful machine intelligences over the internet.

● Technology can be misused by corporations, repressive government or criminals. ● By being more aware of ethical issues and facing them responsibly , it will make it more sustainable and more human .

● Practitioners of IOT may have an opportunity to contribute to providing moral leadership in many ethical challenges.

● While designing anything we have to remember two contrasting points :-

1. Everyone is not you :- Though you might not personally care about privacy or flood levels caused by global warming. They may be critical concerns for other people in different situations.

2. You are not special : - If something matters to you, then perhaps it matters to other people too.

● We should remember an important lesson – Don't assume you know it all . The internet of things is interdisciplinary and it stretches most of the individual displinces too . You will need help from others . Be ready to partner with their organization's, collaborate with people from very different backgrounds to you .

Q5. Attempt *any three* of the Following.

a. Explain the sketch, iterate and explore process in prototyping.

Ans:

1) Sketching enables you to brainstorm, explore multiple ideas, define flows, communicate with team members all why being quick and cheap. Iterative design is a design methodology based on a cyclic process of prototyping, testing, analysing, and refining a product or process.

2) A prototype is a draft version of a product that allows you to explore your ideas and show the intention behind a feature or the overall design concept to users before investing time and money into development.

3) Doing a large amount of introductory jobs doesn't mean that you won't do extra research and gathering of possible beginning points when you do eventually sit down with a particular estimate in mind.

4) To start with, you're looking for a extensive search across the problem area. The motive is to get to clutch with as many aspects of the design as possible, rather than drilling down into one specific possible result. Pushing beyond the obvious solutions forces you to look at things differently and increases the likelihood that you'll have a good design among your options.

5) Use whatever tools make most sense to help with the idea triggering and investigation. You might use a mood board—a whiteboard where you jot down thoughts and sketches over a few days—or a notebook that you doodle sketches.

6) They only need to capture and convey your ideas. The more sketching you do, the better they will get.

7) Then give it or show it to some of the people who might use the finished item to find out how they interact with it.

8) Arguably, in the early stages of a design, you can never do too much iterating through ideas and trying out different approaches to solving the problem.

9) If the idea warrants it (and maybe even if it doesn't), don't be afraid to take your sketching into three dimensions. Mock up different designs with modelling clay or LEGO or some of the other methods we cover in this chapter. Try out different sizes and see how the changes in dimensions affect the feel or the look of the design.

10) Maybe even combine the approaches here with the things you learnt in for prototyping the software and electronics and lash up a rough-and-ready prototype that you can try out properly. Then give it or show it to some of the people who might use the finished item to find out how they interact with it.

11) For example, the evolution of the design for the Good Night Lamp (which we featured in a case study in the last chapter). The original design was a more traditional lamp shape, but in a design workshop, the team batted around a range of ideas with the help of a purely functional prototype. They realised that a design echoing the shape of a house better conveyed the core concept of connecting loved ones in their homes.

12) The key lesson is to use these techniques to experiment with different possibilities and learn which features of which designs are best. This approach allows you to synthesize the results into a coherent final design.

b. What are the features that need to be considered while choosing a laser cutter?

Ans:

● When choosing a laser cutter, you should consider two main features:

1. The size of the bed: This is the place where the sheet of material sits while it's being cut, so a larger bed can cut larger items. You don't need to think just about the biggest item you might create; a larger bed allows you to buy material in bigger sheets (which is more cost effective), and if you move to small-scale production, it would let you cut multiple units in one pass.

2. The power of the laser: More powerful lasers can cut through thicker material. For example, the laser cutter at our workplace has a 40W laser, which can cut up to 10mm-thick acrylic. Moving a few models in the same range, to one with a 60W laser, would allow us to cut 25mmthick acrylic. ● For Internet of Things devices you will probably be looking at something more rigid. Card and, particularly, corrugated cardboard are good for quick tests and prototyping, but MDF, plywood, and acrylic (also commonly known by the brand name Perspex) are the most common choices. ● Specialised materials are also available for specific purposes. For example, laserable rubber can be used to create ink stamps, and laminate acrylic provides a thin surface in one colour, laminated with a thicker layer in a contrasting colour so that you can etch through the thin layer for crisp, highcontrast detailing and text.

● The file formats or software which you need to use to provide your design vary across machines and providers. Although some laser-cutting software will let you define an engraving pattern with a bitmap, typically you use some type of vector graphics format.

● Vector formats capture the drawing as a series of lines and curves, which translate much better into instructions for moving the laser cutter than the grid-like representation of a bitmap. There's also no loss in fidelity as you resize the image.

● CorelDRAW is a common choice for driving the laser cutters themselves, and you can use it to generate the designs too. Other popular options are Adobe Illustrator, as many designers already have a copy installed and are familiar with driving it, and Inkscape, largely because it's an open source alternative and therefore freely available.

c. Explain the use of repurposing /recycling in prototyping IoT devices.

Ans:

Owning the designs of and knowing how to create all of the components of your device put you in a great position, but they aren't necessarily the overriding concerns in all prototyping scenarios.

As with the other elements of building your connected device, a complete continuum exists from buying-in the item or design through to doing-it-yourself. So, just as you wouldn't think about making your own nuts and bolts from some iron ore, sometimes you should consider reusing more complex mechanisms or components.

One reason to reuse mechanisms or components would be to piggyback onto someone else's economies of scale. If sections or entire subassemblies that you need are available in an existing product, buying those items can often be cheaper than making them in-house. That's definitely the case for your prototypes but may extend to production runs, too, depending on the volumes you'll be manufacturing. For example, the bubble machine usedin Bubblino is an off-the-shelf unit from a children's game. In the batch production volumes that Bubblino is currently being built, it's cheaper to buy them, even at retail price, than it would be to manufacture the assorted gears, fans, bubble ring, and casing in-house. Or perhaps you're making just a couple of units or maybe only one. In that scenario the labour involved in working out how to integrate the electronics, graft in newly fabricated parts, or work out how to disassemble the reused item for the bits you need might not matter, as you aren't going to be repeating it many, many times.

That's often the case with one-off items, when they are deliberately incorporated into existing, mass-produced products. When Russell Davies commissioned Adrian to build him a few minimal-interface WiFi sound boxes. He asked for two separate devices. One was made from scratch with all-new electronics and a laser-cut case, whereas the other was a reworked 1974 transistor radio. The radio circuits were removed to make space for a small ARM Linux board, but the original amplifier was retained and the wave band selector switches were reconfigured to act as the interface to the program. When everything was boxed up, it became a very familiar object but with very new capabilities.

We’ve drifted away from the idea of prototyping as a way to explore and develop your idea, but that is probably the most common case where reuse and repurposing of existing items are useful.

Given that the prototyping phase is all about rapid iteration through ideas, anything that helps speed up the construction period and gets you to where you can test your theories is useful. When you are thinking through the user interaction of a connected bedside table, for instance, gaffer taping an Arduino to your alarm clock would provide a good-enough approximation to let you try out different scenarios.

If the final design requires processes with massive up-front costs (such as tooling up for injection moulding the plastics) or the skills of a designer that you don’t have the funds to hire right now, maybe a product already exists that is near enough to work as a proxy. That lets you get on with taking the project forwards, ending up at a point, one hopes, where making the bigger investment makes sense.

d. What is an API? What do you mean by mashing up API?

Ans:

API: ● API is a short form of Application Programming Interface.

● It is a software intermediate which allows 2 or more than 2 Applications to work or communicate with each other.

● Whenever you use an application on your phone, the application connects to Internet, the data is sent to the server.

● The server then analyses the data entirely and performs necessary actions and sends it back to your device.

● The Application than work on that data and presents it to the user in humanreadable format. This what an API is, everything happens via API.

● To explain this in better way, suppose you are sitting in a restaurant, you have many choices of orders. The kitchen is the main part of the System. Now the link between your order and the kitchen is missing.

● Here is where waiter (API) comes in, the waiter is the messenger who takes the order to kitchen and again serve you the food from the kitchen.

● There are 3 types of APIs as followed:

> ➢ Public APIs : These APIs are available publicly for everyone with no restrictions.
>
> ➢ Partner APIs : These APIs are available for business partner, not available for everyone but needs entitlement to access them.
>
> ➢ Private APIs : These APIs are usually less known to the users and are only accessible by internal systems.

<u>API Mashup:</u> ● A Mashup API is the API that integrates countless API.

● Sometimes there is data on the internet which is in the form which doesn’t work for you.

● So, we merge different APIs together to get our work done.

● Mostly Mashups are visual or interactive in nature.

● To the user/developer, Mashup provides a interactive experience.

● It is also useful for developers because it requires less coding and it developed quickly.

● Here the idea of mashing up of multiple APIs is use for powerful effect.

●

  - ◆ Using mapping API to allocate properties to sell, rent or buy.
  - ◆ Displaying Twitter trends on a global map or a charting API.

e.What are the legalities associated with scrapping?

Ans:

•most of the companies have sample of relevant data but they don’t have knowledge to make it available as API's.

•From facebook example you understand that the computer is only suppose to run the browser and navigate it through UI elements which can be delicate but making computer do more work is not impossible.

•this is known as "screen - scrapping"

•screen-scrapping is not useful for website as it may break the terms and conditions that a website can follow.

•For example the Google won't allow to screen scrap its search bar but will provide a API for the same. Even if you don't follow the legal rules the basic terms and conditions that are put by the company like Google should be followed else the company will exclude you from all other services that is provides that me lead to in convenience.

•Rest of the data is protected with various security techniques like copyright, authenticating users to database rights and many more.

•There are always alternatives for all the information that is available.for example you can use microsoft Outlook rather than using Gmail or you can use OpenStreetMap rather than using GoogleMaps.

f.Explain HTML5 web socket?

Ans: Web Socket is a web technology that provides full-duplex communication channels over a single TCP connection. A full-duplex communication is the communication system that allows simultaneous bidirectional communication. A telephone conversation is a good example of full-duplex communication where in both parties can speak and hear at the same time.

- The API used to talk directly to the TCP layer is known as the sockets API. When the web community was looking to provide similar capabilities at the HTTP layer, they called the solution Web Sockets.

- Web Sockets are currently a working draft in the HTML5 spec, they are embedded in modern browsers, servers, and other clients.

- Web Sockets have the benefit of being bidirectional. You can consider them like a full Unix socket handle that the client can write requests to and read responses from.

- This might well be the ideal technology for the task timer. After a socket is established, the timer can simply send information down it about tasks being started, modified, or cancelled, and can read information about changes made in software, too.

- Because Web Sockets are new and push the HTTP protocol in a slightly unorthodox direction, they are known to have some issues with proxy servers.

- This situation should change as the proxies currently broken in this respect are fixed to be aware of Web Sockets.

- One of the more unique features WebSockets provide is its ability to traverse firewalls and proxies, a problem area for many applications. Comet-style applications typically employ long-polling as a rudimentary line of defense against firewalls and proxies.

- The technique is effective, but is not well suited for applications that have sub-500 millisecond latency or high throughput requirements.

- A WebSocket detects the presence of a proxy server and automatically sets up a tunnel to pass through the proxy.

- WebSockets—like other pieces of the HTML5 effort such as Local Storage and Geolocation—was originally part of the HTML5 specifications.