SYLLABUS

Unit	Details
V	**Moving to Manufacture:** What Are You Producing? Designing Kits, Designing Printed circuit boards, Software Choices, The Design Process, Manufacturing Printed Circuit Boards, Etching Boards, Milling Boards. Assembly, Testing, Mass-Producing the Case and Other Fixtures, Certification, Costs, Scaling Up Software, Deployment, Correctness and Maintainability, Security, Performance, User Community. **Ethics:** Characterizing the Internet of Things, Privacy, Control, Disrupting Control, Crowdsourcing, Environment, Physical Thing, Electronics, Internet Service, Solutions, The Internet of Things as Part of the Solution, Cautious Optimism, The Open Internet of Things Definition.

Books and References

Sr. No.	Title	Author/s	Publisher	Edition	Year
1.	Designing the Internet of Things	Adrian McEwen, Hakim Cassimally	WILEY	First	2014
2.	Internet of Things Architecture and Design	Raj Kamal	McGraw Hill	First	2017
3.	Getting Started with the Internet of Things	Cuno Pfister	O'Reilly	Sixth	2018
4.	Getting Started with Raspberry Pi	Matt Richardson and Shawn Wallace	SPD	Third	2016

Chapter 10

Moving to Manufacture

After creating and showcasing your prototype to friends or on the Internet, one to most frequent questions they ask is “Can I get one?” and you start to wonder about creating the product and selling it.

How many devices are you looking to sell? Even if you’re just going to make a few in your spare time, what would happen if your idea turns out to be a huge hit and you have half the Internet beating a path to your order page? It’s a nice problem to have, but if your costings expect your labour to be free because you enjoy soldering up a few boards, you might need to rework those costs more realistically. Either that or you will soon find out just how much you enjoy soldering boards when you’re doing it all day every day to meet demand.

If you don’t want to make each item by hand, you will need to find someone who can do the job for you. Hiring out work is an extra cost, naturally, but at the same time you’ll usually be buying the components in greater bulk, which will be cheaper, so things often balance out. If you are making enough, the economies of scale of automation kick in, which means you can either lower your prices or increase your profits—or, ideally, both.

Then there are the less obvious things. Will you need to design some packaging so that the products aren’t damaged during shipping or to make them appealing when sat on a shelf alongside a host of other gadgets? And what about certification? If you’re selling the product in the United States, the Federal Communications Commission (FCC) will want to make sure it is safe and won’t be throwing out a lot of unwanted electromagnetic interference. In Europe, you’ll need to meet similar requirements to use the CE mark, and there are additional regulations such as the Restriction of Hazardous Substances (RoHS), a directive which bans the sale of products containing certain substances (such as lead and cadmium) above given levels.

In this chapter we dig into both the obvious and the less obvious steps in turning your prototype into a finished product.

10.1 What are you producing?

Before we get into the specifics of *how* to scale up production of your device, it’s useful to pause for a moment and consider *what* exactly you’ll be producing. Depending on your motives and desires for the product, not to mention the amount of time and money that you have available to devote to it, a full spectrum of possibilities is open to you. If your ambition is just to enable others to build a version of it for themselves, you only need to document the build process and share it, along with any source code and design files you used, on the Internet—either on your own blog or website, or on a site like Instructables. Aside from maybe a bit of promotion on sites such as Make or Hack-a-Day to help people find it, you are basically done.

A huge maker community on- and offline share their projects and experience this way. Joining in is a great way to meet fellow makers and to give something back to the community if they

helped you get started. You might find that you become the go-to person for that sort of project, and people might commission you to build related projects for them or want to hire you to work in their company. When you decide that you want to get your invention into the hands of many more people, you can generally split your choices into three categories:

- A kit that your customers can assemble themselves.
- A ready-made electronics board which users can use as a sub-assembly in their own projects.
- A fully fledged product, complete with its housing, instructions, and even packaging, just waiting to be put on the shelf at your local department store.

Figure 10-1: Raspberry Pi 3 starter kit

Each of these options builds on a lot of the work you would already have done for the previous one. A complete device will contain an assembled electronics board, and the electronics board would need a printed circuit board (PCB), much like the one users would need to solder components to in a kit.

10.2 Designing Kits

The first step towards selling your idea as a product is to provide it as a kit. Although you might think it is simple to order the relevant parts and then lay them out to follow your schematic, you are likely to be underestimating how much you have learned in working out how to make

it. For every person like yourself who gets to the end of building something, there are many more who would have given up halfway through or found it too daunting to even start.

Most kits tend to provide only the electronics and software for a particular application rather than any physical housing. The reason partly comes down to the difficulty, for a cottage kit-building industry, of sourcing custom plastic components. Also, because the main market for such kits is others in the maker community, these makers will perhaps prefer to combine the kit into a project of their own. However, with the growing accessibility of 3D printers and laser-cutters, it is becoming vastly more feasible to provide housing and other physical components even for kits.

Kits tend to piggyback on existing microcontrollers and often take the form of a standard format plug-on board—for example, a *shield* in the Arduino ecosystem or a *cape* on the BeagleBone. This makes sense because it reduces the support overhead for the kit provider; either users will already be familiar with the platform, or, if not, plenty of assistance will be available elsewhere to cover the basics of getting up and running. The kit's documentation can then focus on just what is specific to building the project.

Given that you've built your prototype, you've already done most of the work needed to create a kit. You've worked out which components you need, where to source them, and how to wire them up to create a functioning circuit. And you've written the necessary software to interface to and control the electronics. The parts you might not have done include designing a PCB, documenting the build process, and working out the costs. Documenting the build process isn't too tricky. When you have everything together for one of your kits, run through assembling one of them yourself. As you go, take photos or video of each step and write down the order of each step. The Instructables website has a number of guides to help out, and another good approach is to copy the procedures used by other kit makers that you've found useful.

Working out what price to charge is harder and more of an art form than a science. Some rules of thumb can help, though. The most obvious is to understand what your costs are. You should create a spreadsheet or list of all the items that make up your product, along with their cost to you to buy. You should list every single electronic component, connector, cable, PCB, case, and so on, in addition to the packing box you'll ship it in and the time taken to put things together. This list is called the *bill of materials* (BOM), and it forms the starting point for all your costings and prices. The BOM gives you the marginal cost for your product and doesn't include any of the fixed costs in setting up for production or developing the software, as they'll be spread out across all the items you sell.

A good starting point for working out your price is to take the total cost of the BOM and multiply it by 4 or 5. That calculation gives you a margin to cover that item's portion of the fixed costs and also some profit. It also provides enough margin for resellers to cover *their* fixed costs and make some profit. This means that if you end up with a hit on your hands, you can enlist some intermediaries to help scale up distribution without losing money on each kit you sell.

The most important cost to drive down is that of the BOM. That will give you the most flexibility in setting your price so that you can maximise the profits that you make. By looking at the prices of similar products, you can get a flavour of the market and use that to guide your pricing. You don't need to necessarily find products which solve the same problem; you can

also look at ones that are delivered in a similar form. For example, if your kit is a shield for Arduino, you can compare prices of other shield kits to get a feel for the range of prices that people expect to pay.

Because the customer is responsible for assembling and soldering the kit together, usually the only thing you'll need to get custom made just for your application is the bare PCB. That tends to keep your costs down, as you're offloading the labour required to build the circuit to the kit's user. For the target demographic for a kit, the need for assembly can be seen as a benefit because it adds to the sense of achievement and ownership in getting the kit working. However, it can also add to the support overhead, because you will need to deal with remotely debugging your customers' issues, which may be down to their poor soldering rather than defects in your work. These problems can all be resolved by moving on a step towards a consumer product and selling fully assembled PCBs, populated with all the components. If you aren't selling too many and your design doesn't include any especially fine-pitched or complicated surface-mount components, you might decide to assemble them yourself; otherwise, you'll need to find an assembly house to do the work on your behalf. Either option will also benefit from your working out a way for you to test the finished board, checking that the components are soldered in properly and, ideally, that they function correctly. Obviously, the final step from kit to consumer product is to manufacture and assemble the housings, linkages, and whatever else is part of the finished device. This incorporates the assembled PCBs and adds more components or processes.

Video URL: http://youtube.com/watch?v=mv7Y0A9YeUc

Video 10-1: How to make a PCB at home

10.3 Designing Printed Circuit Boards

Soldering things up is a good step towards making your prototype more robust, because the connections should, if your soldering skills are up to par, mean that you have a solid electrical connection that won't shake loose. Moving beyond stripboard, designing and etching your own custom PCBs gives you more options on how to lay out the circuit and makes it easier to solder up as the only holes in the board will be those for components. It also lets you use components that don't easily fit into the stripboard grid pattern of holes, including some of the simpler surface-mount components. Moving to professionally manufactured boards further simplifies the assembly process because the solder mask will make the soldering a bit easier, and, more importantly, the silkscreen provides outlines of where each component should be placed. The range of options for building a custom PCB runs from etching (or milling) boards yourself, through using one of the many mail-order batch PCB services, to having them made and populated for you.

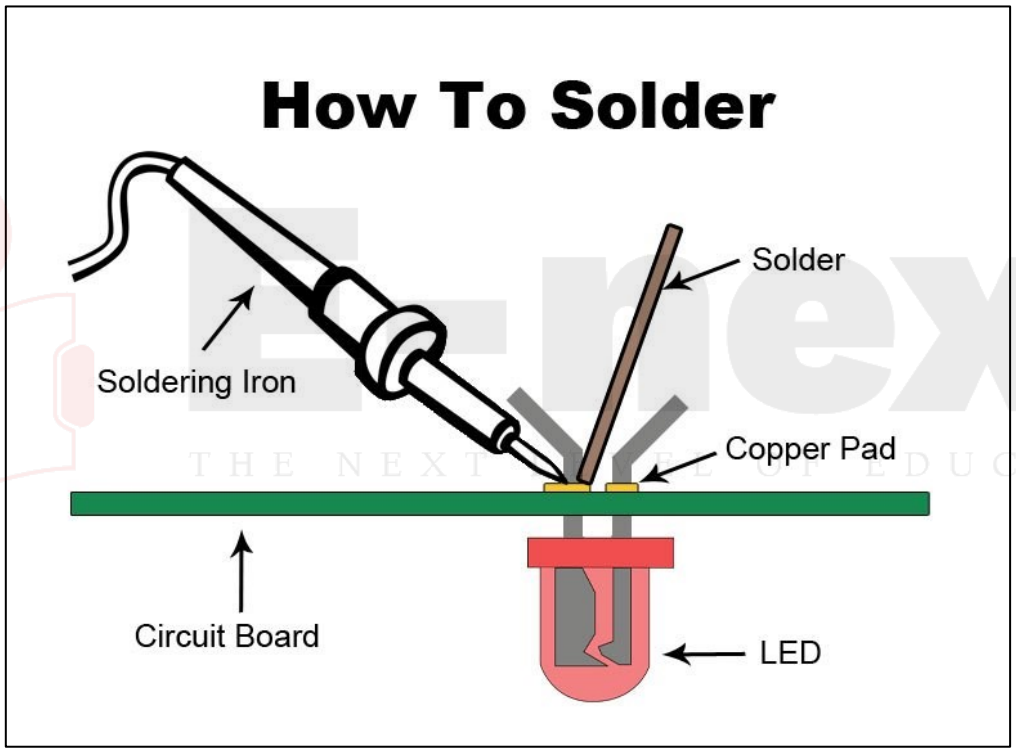
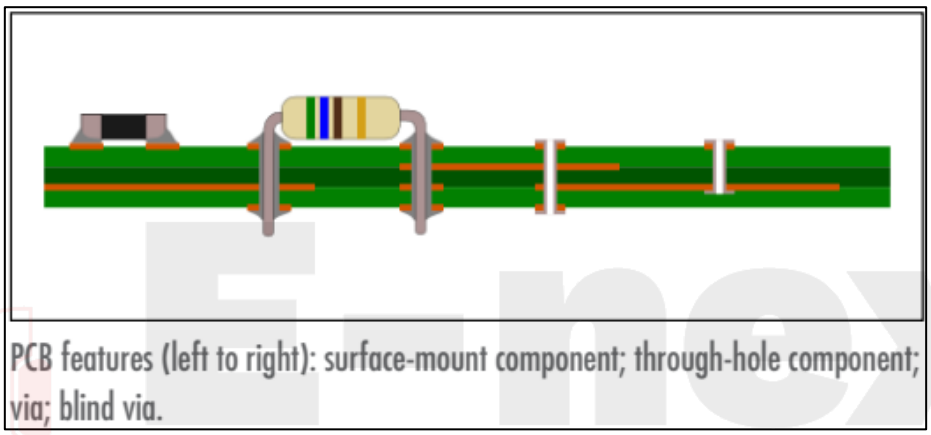


Figure 10-2: How to solder

Whichever of those options you choose, the first step in creating your PCB is going to involve designing it. Before we investigate the available software for PCB design, we should look at what makes up a PCB and some of the terms you are likely to encounter. The PCB is made up of a number of layers of fibreglass and copper, sandwiched together into the board. The fibreglass provides the main basis for the board but also acts as an insulator between the different layers of copper, and the copper forms the "wires" to connect the components in the circuit together. Given that you won't want to connect all the components to each other at the same time, sections of the copper are etched away—usually chemically, but it is possible to use a CNC mill for simple boards. These remaining copper routes are called *tracks* or *traces*

and make the required connections between the components. The points on the tracks where they join the leg of a component are known as *pads*. For surface-mount components, they are just an area of copper on the top or bottom of the board, which provides the place for the component to be soldered. For through-hole connections, they also have a hole drilled through the board for the leg to poke through.

Single-sided boards have only one layer of copper, usually on the bottom of the board; because they're often for home-made circuits with through-hole components, the components go on the top with their legs poking through the board and soldered on the underside. Double-sided boards, predictably, have two layers of copper: one on the top and one on the bottom. More complicated circuits, particularly as you start to use more advanced processors which come in smaller packages, may need even more layers to allow room to route all the traces round to where they are needed. Three- or five-layer boards aren't uncommon, and even some seven-layer boards are used for really complicated circuits.



PCB features (left to right): surface-mount component; through-hole component; via; blind via.

Figure 10-3: Different way to mount component on PCB

When you need to connect traces on two layers together at a point where there isn't a hole for the leg of a component, you use a *via*. This is a similar, though generally smaller, hole through the board purely to connect different layers of copper once plated. You also can use blind vias, which don't pierce all the way through the board, when you don't want to connect every layer together; however, because of this, they complicate the PCB manufacturing process and are best avoided unless absolutely necessary.

In places where you have many connections to a common point, rather than run lots of tracks across the circuit, you can more easily devote most of a layer of the board to that connection and just leave it as a filled area of copper. This is known as a *plane* rather than a trace and is frequently used to provide a route to ground. An added advantage of this approach is that the ground plane provides a way to "catch" some of the stray electromagnetic signals that can result, particularly from high-frequency signal lines. This reduces the amount of electromagnetic interference (EMI) given out by the circuit, which helps prevent problems with other parts of the circuit or with other nearby electronic devices.

The surfaces of professionally manufactured PCBs undergo processes to apply two other finishes which make them easier to use. First, all the parts of the board and bare copper which aren't the places where component legs need to be soldered are covered in solder mask. Solder mask is most commonly green, giving the traditional colour of most circuit boards, though

other colours are also available. The mask provides a solder-resistant area, encouraging the solder to flow away from those areas and to adhere instead to the places where it is needed to connect the components to the tracks. This reduces the likelihood of a solder joint accidentally bridging across two points in the circuit where it shouldn't. Then, on top of the solder mask is the silkscreen. This is a surface finish of paint applied, as the name suggests, via silkscreen printing. It is used to mark out where the components go and label the positions for easy identification of parts. It generally also includes details such as the company or person who designed the board, a name or label to describe what it is for, and the date of manufacture or revision number of the design. This last piece of information is vital; it is more likely than not that you'll end up with a few iterations of the circuit design as you flush out bugs. Being able to tell one version from the other among the boards on your workbench, or, more importantly, knowing exactly which design is in a product with a user reported fault, is essential.

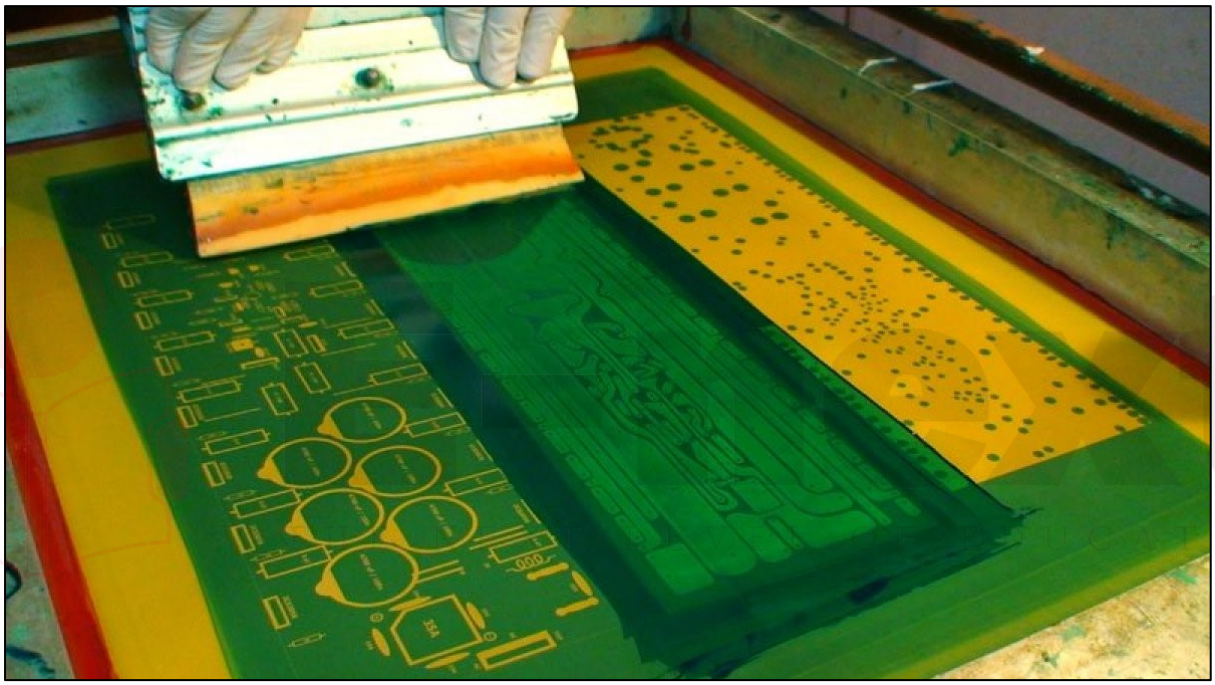


Figure 10-4: Silkscreen PCB printing

10.3.1 Software Choices

As you might expect, you have many different choices when looking for some software to help you design your PCB. If you are working with a contract electronics design house, the staff may well use something like Altium Designer, but you're more likely to use one of the following lower-end (and cheaper) options.

Fritzing is a free, open source design package aimed particularly at beginners in PCB design. It deliberately starts with a design screen resembling a breadboard and lets you map out your circuit by copying whatever you have prototyped in real life. It then converts that design to a schematic circuit diagram and lets you arrange components and route the traces on the PCB view. Fritzing even offers a fabrication service to make it simple to make your design a reality. You also can export the breadboard design view as an image or a PDF.

KiCad is another open source offering but with a more traditional workflow. It has a more comprehensive library of predefined parts and can be used to design boards with up to 16 layers of copper, compared to the double-sided boards that Fritzing produces.

Probably the most popular PCB layout software for the hobbyist and semi-professional market is EAGLE from CadSoft. The reason for its popularity most likely comes down to its long having a free version for non-commercial use, allowing beginners to get started. That led to a wealth of how-to guides and other helpful resources for EAGLE being developed and shared by the user community.

A more recent rival to EAGLE in the commercial pro-sumer market is DesignSpark PCB. It is provided by electronics distributor RS Components as part of the company's DesignSpark community and, as a result, is free of charge. Unlike EAGLE, DesignSpark PCB does not restrict the size of boards you can design (up to 1m2) or the number of layers (it supports up to 14 layers). However, it is the only program described here which isn't available for Linux or Mac. Compatibility is an issue which RS Components acknowledges and does endeavour to ensure that it works under the Windows compatibility layers Wine on Linux and CrossOver for Mac; however, in practice it's a little clunky.

10.3.2 The Design Process

Regardless of the exact program you decide to use to lay out your PCB, your task in creating the design is split between two main views: the schematic and the board.

10.3.2.1 The Schematic

You usually start the design in the schematic view. It lets you lay out the components logically and make the necessary connections without having to worry about exactly where they'll sit in physical space or whether any of the tracks cross. The schematic provides a conceptual view of how the circuit is laid out. Components are represented by their standardised symbols, and you usually group them into areas of common functionality instead, which won't necessarily match the groupings they'll have in the physical layout. Your software package includes most of the common items you need in its library: common resistors, diodes, capacitors, transistors, integrated circuits (ICs), and more. Adding one of those is merely a case of finding the relevant part and sometimes adding the exact part number.

You should also make sure you choose the correct *package* for the component; this is the physical format for it. Many parts are available in a range of different formats, depending on the target application. The manufacturer's datasheet lists the available packages for that part and how they differ. You usually have a choice of packages depending on the soldering style (through-hole or surface-mount) and several other criteria. The choice of package doesn't actually affect the schematic because at this point you are interested only in the functionality of the component rather than its physical attributes. However, making the right choices when initially choosing the component will save you having to rework that down the line when you come to the board view.

It’s not uncommon to find that you want to use a part which isn’t included in your PCB software’s component library. If you’re lucky, you might find that someone else has provided it in a third-party library. Otherwise, you’ll have to design your own. Creating the design is not difficult because all the information you need is in the datasheet, but attention to detail here will pay dividends with how well the part sits on the finished PCB.

When you are happy with the schematic design of your board, you can proceed to laying out the physical board. Although it makes sense to approach the PCB design in this manner, it is not a rigid procedure. You can flip back and forth between the two views of the design, but finishing the schematic as best you can before moving to the physical aspect will minimise the amount of rework you’ll have to perform if the design changes.

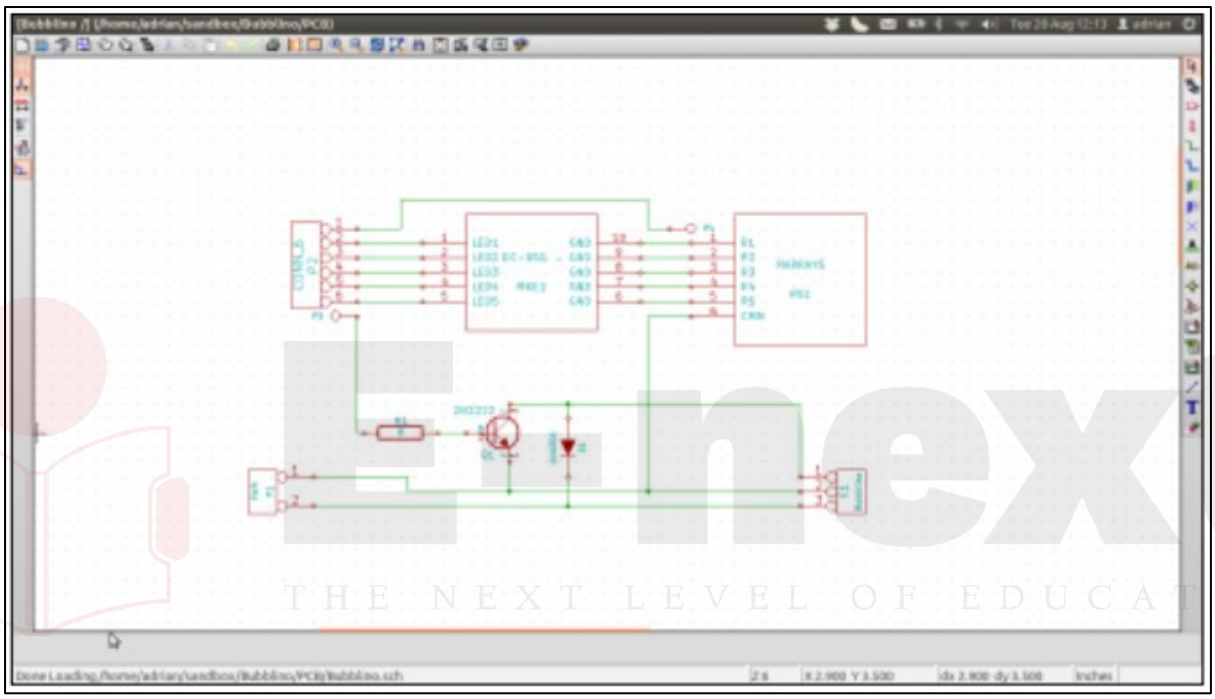


Figure 10-5: The schematic view of Bubblino PCB, as designed in KiCad.

10.3.2.2 The Board

There are no hard-and-fast rules on how to arrange the board. It makes sense to keep together groups of components that constitute an area of functionality, such as the power supply, for a more logical layout. And you might find the design has certain fixed aspects—things such as connectors which all need to be along one side for access once in a case, for example, or arrangements of headers to mate with an Arduino board. After that, it’s just a case of arranging components in a way that makes the routing of wires easiest.

Start by placing the components which have the most connections first and then fitting the remainder around those constraints. As you move the components around, you may notice a criss-crossing of fine lines joining some of them together, which seem as messy as cheese strings trailing from a slice of pizza. These connections join the various pads on the components to each other. As you position the components, you should try to reduce how tangled they are,

but don’t worry about this issue excessively; you’re just aiming for a good starting point for the work of routing the connections properly.

Your PCB software has an auto-route function, which you can use to route all the tracks for you. However, such functions are far from perfect, and in practice, you will find it best to at least lay out the more important tracks first by hand, if not to do all the routing manually. After all the tracks have been routed, your PCB design is almost finished. You should add any labels to the silkscreen layer to explain things such as nonobvious connectors and to identify the board itself. Also include a version number so that you can differentiate any future revisions, and also add maybe your name or the company name and a website for people to find out more. Then give your PCB a final once-over.

Run the design rules check to catch any missed connections or problems where tracks are too close to each other or too thin for your PCB manufacturer to reliably manufacture. The design rules effectively contain the manufacturing tolerances for your PCB manufacturer. They define things like the minimum track width or the minimum distance between pads. Then print out a copy on paper; this way, you can compare the components to their location on the PCB in real life and spot any errors in their footprints. After you fix any issues thrown up by those last checks, you’re ready to make the PCBs.

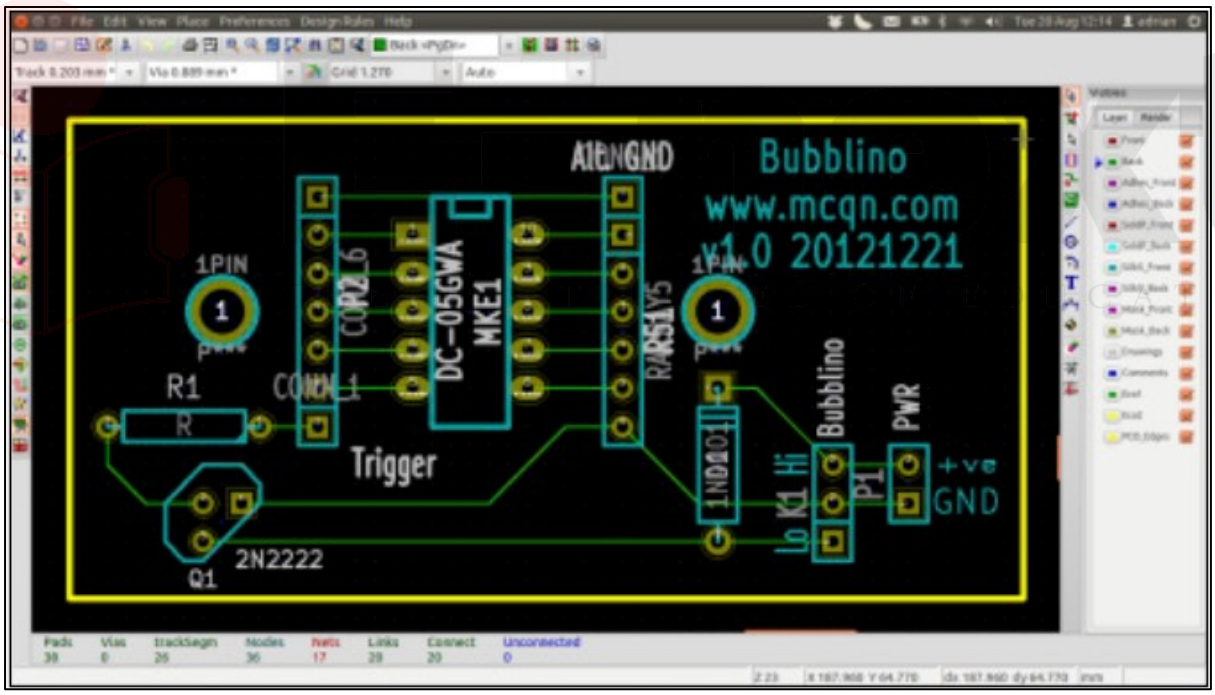


Figure 10-6: The finished View of Bubblino PCB board, ready for export.

10.4 Manufacturing Printed Circuit Boards

Now that you’ve designed your PCB, the next step is to make one or lots of them. If you want only a couple of boards, or you would like to test a couple of boards (a very wise move) before ordering a few hundred or a few thousand, you may decide to make them in-house.

10.4.1 Etching Boards

The most common PCB-making technique for home use is to etch the board. The first step is to get the PCB design onto the board to be etched. This process generally involves printing out the design from your PCB design software onto a stencil.

If you're using photo-resist board, it will be onto a stencil which masks off the relevant areas when you expose it to UV light; or if you're using the toner-transfer method, it will be for your laser printer to print onto glossy paper ready to transfer. Your stencil then needs to be transferred to the board. For photo-resist board, you will expose it under a bright lamp for a few minutes; and for the toner-transfer method, you'll use a super-hot iron. With the board suitably prepared, you can immerse it into the etching solution, where its acidic make-up eats away the exposed copper, leaving the tracks behind.

Figure 10-7: Etched Board immersed in etching solution

After all the unnecessary copper has been etched away, and you've removed the board from the etching bath and cleaned off any remaining etchant, your board is almost ready for use. The last step is to drill the holes for any mounting points or through-hole components. You can do this by hand, or, if you have access to a CNC mill, you can export the drill file from your PCB design package to provide the drill locations for your mill.

10.4.2 Milling Boards

In addition to using a CNC mill to drill the holes in your PCB, you can also use it to route out the copper from around the tracks themselves. To do this, you need to export the copper layers

from your PCB software as Gerber files. These were first defined by Gerber Systems Corp., hence the name, and are now the industry standard format used to describe PCBs in manufacture.

To translate your Gerber file into the G-code that your mill needs requires another piece of software. Some CNC mills come with that software already provided, or you can use a third-party program such as Line Grinder. The mill effectively cuts a path round the perimeter of each track to isolate it from the rest of the copper. As a result, PCBs which have been milled look a bit different from those which are etched because any large areas of copper that aren't connected to anything are left on the board (to save time milling it away).

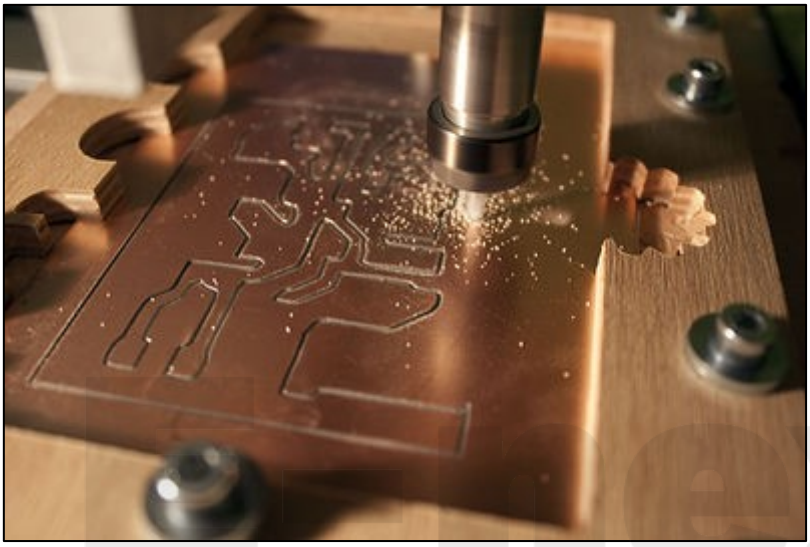


Figure 10-8: CNC mill drilling holes for PCB

10.4.3 Third Party Manufacturing

If your design has more than two layers, if you want a more professional finish, or if you just don't want to go to the trouble of making the PCBs yourself, many companies can manufacture the boards for you. The price for getting the boards made varies based on the complexity and the size of the design but also varies quite a bit from company to company, so it's worth getting a few quotes before deciding which one to use. If you need the boards quickly, a local firm is your best bet and generally has a lead time measured in days. If you have the luxury of more time, you can cast your net further, including to China, which might reduce your costs but could mean a few weeks' wait before you receive your order. Either way, the Gerber files are what you need to provide to the manufacturer. Make sure you export all the relevant layers from your design, meaning each of the copper layers you're using, plus the solder mask (which gives the PCBs their colour and stops any solder adhering to areas where it shouldn't), silkscreen (for the labels, versioning info, and so on) and drill files.

10.4.4 Assembly

After your PCBs have been manufactured, you still need to get the components soldered onto them. If you're selling them as kits, the customers will solder things up, so you just need to pack everything into bags and let them get on with it. Otherwise, you have to take responsibility for making that happen. For small runs, you can solder them by hand. For through-hole boards, break out your soldering iron. Surface-mount assembly is a little more involved but quite achievable if you don't have any components with particularly complicated package types.

For assembling surface-mount boards, you need one more item from your PCB design Gerber collection: the solder paste layer. You use it to generate a stencil that allows you to apply the solder. You can laser-cut one from a thin sheet of Mylar plastic or have one made for you out of thin steel. The solder for surface-mount work comes as a paste, supplied in tubs or tubes. Using a squeegee and the solder paste stencil, you need to put down an even layer of solder over all the component locations and then carefully lift the stencil off the board.

Now comes the tricky bit. Using tweezers and ideally a loupe or magnifying glass, place each component onto the relevant spot on the PCB. The paste holds the parts in place to a degree, but take care not to knock the board at this point in case some of the parts get displaced. When you have all the components on the board, you need to melt the solder to fix everything in place. You can do this with a soldering iron, but doing it by hand is easier if you use a hot-air rework station. You can solder all the connections at once if you use a reflow oven. As the name suggests, this oven heats up the PCB and components evenly until the solder melts. The maker-community retailer Sparkfun Electronics has a good tutorial covering some of the techniques that it has used for surface-mount soldering.

After you outgrow hand assembly, you will need some help from robots. In this case, you will need robots that can pick up components using a tiny vacuum nozzle, rotate and place them in the right location on the PCB, and then repeat that process at a rate of tens of thousands of components per hour. These robots are known as pick-and-place assembly machines. The components to feed into the machine are supplied in a form known as tape and reel. Considering all this, if you are running your own pick-and-place machine. This complexity can be reduced if you offload some of the work to an assembly house, also known as a contract manufacturer. Contract manufacturers are firms geared up to helping people produce finished, populated PCBs. Often they offer a range of services from PCB design, through dealing with PCB manufacturers, to soldering up the components and even testing the completed boards.

Using an assembly house saves you from buying the expensive machinery yourself. But offloading the work to someone who specialises in it has other benefits, too. The extra throughput that they deal with means they will naturally employ dedicated staff to run the production lines. Paradoxically, those staff will most likely be both more skilled than you are at running the pick-and-place machines and doing any hand soldering and cost less per hour than you do. This then frees you up for the many other tasks involved in bringing a product to market or to working on your next idea. If you do decide to use a contract manufacturer, having a conversation about components is worthwhile. For common parts, if you don't have specific requirements for tolerances, and the like, you might be able to specify that the assembler should use the parts it already holds in stock. If the manufacturer has dealt with the supplier before, its reputation might let you negotiate a better deal.

10.4.5 Testing

Now your boards are all ready and assembled, but how do you know that they all work as they're meant to? This is where testing comes in. Actually, through the automated assembly process, you might have had some testing steps included already. Assembly lines can include automatic optical inspection (AOI). In this process, a high-resolution camera inspects some aspect of the board and its components; for example, it could check that the solder paste is laid properly before the board goes into the pick-and-place machine and compare it to a known good version. Any boards which vary from the "golden" reference version by too high a margin are flagged for further checks from a skilled human operator.

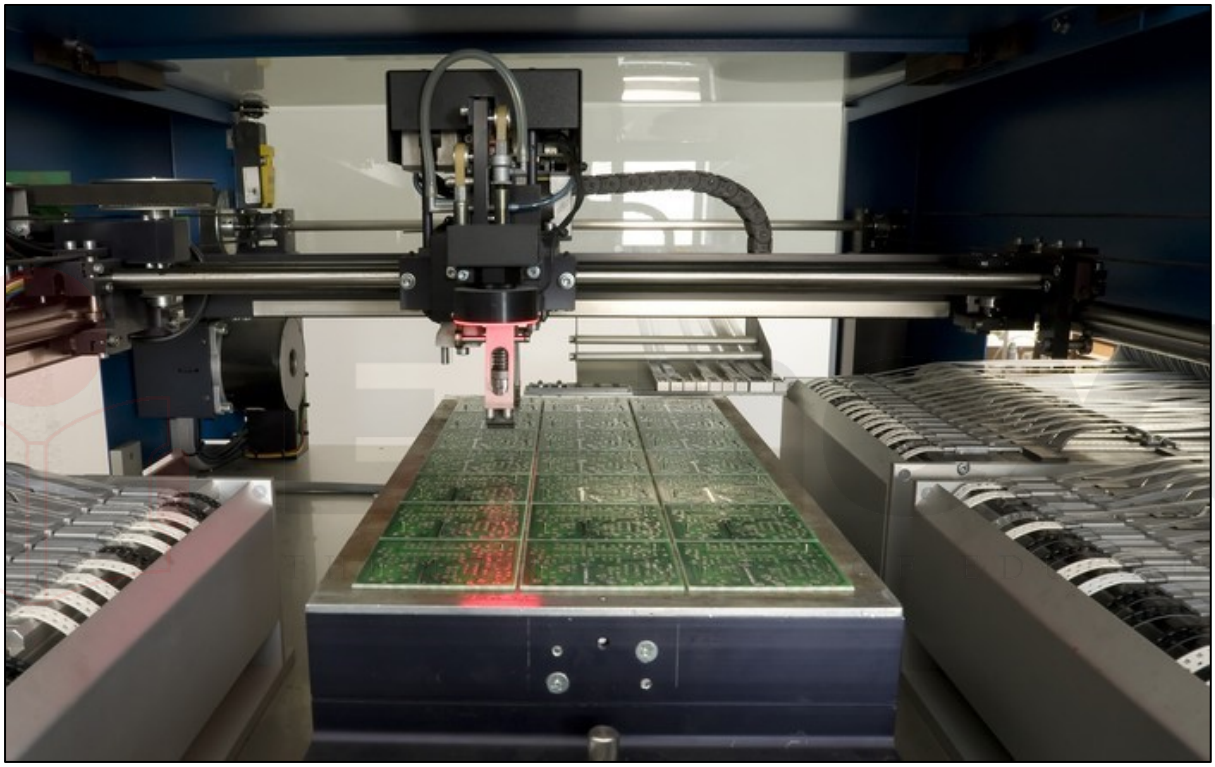


Figure 10-9: Automated Optical Testing

After the boards pass the AOI, the next step is to run them through a functional test. This step is something that you can, and should, be doing even with boards that you've soldered by hand. The functional test just involves powering up the board as it will be used in the finished product and ensuring that it does what it is supposed to. However, that might take a nontrivial amount of time. The focus here is not on ensuring it will run through all normal operations, but just that the PCB and its components are soldered correctly, that none of the components are faulty, and that there aren't any manufacturing defects in the PCB itself.

A better approach is to build a specific test rig to exercise the different parts of the circuit and measure the voltages at set points on the board. These measurements can then be compared against known-good values and a decision made automatically as to whether or not the device under test (DUT) has passed. Because you don't want to spend time making individual

connections for each test, the normal practice for the test rig is to use the mounting holes for the PCB for alignment and then have it held by some clips against a number of carefully prepositioned, spring-loaded pins. These pins are known as pogo pins, and the spring means they can make a good connection to the board without any extra work, such as soldering, when the board is placed into the test rig.

The test program can then run through its tests and measure voltages at different pogo pins at the relevant time in the test to see how the board being tested performs. If the DUT includes a microcontroller chip, the test rig could flash it with a test program to help with the testing, and even at the end of the test, assuming it has passed, flash it with the final firmware image ready for deployment.

10.5 Mass Producing the Case and Other Fixtures

A good rule of thumb for keeping down the costs of production is to minimise the amount of time a person has to work on each item. Machines tend to be cheaper than people, and the smaller the proportion of labour is in your costs, the more you'll be able to afford to pay a decent wage to the people who are involved in assembling your devices. However, whilst minimising labour costs is a good target, it's not the only factor you need to consider in your production recipe; production rates are also important.

To give you a flavour of the sorts of issues involved, we look at what must be the most common method of mass production: injection moulding of plastic. As the name suggests, the process involves injecting molten plastic into a mould to form the desired shape. After the plastic has cooled sufficiently, the mould is separated and the part is popped out by a number of ejection pins and falls into a collection bin. The whole cycle is automated and takes much less time than a 3D print, which means that thousands of parts can be easily churned out at a low cost per part.

The expensive part of injection moulding is producing the mould in the first place; this is known as tooling up. The moulds are machined out of steel or aluminium and must be carefully designed and polished so that the moulding process works well and the parts come out with the desired surface finish. Any blemishes in the surface of the mould will be transferred to every part produced using them. Often for a super-smooth surface, the moulds are finished with a process called electrical discharge machining (EDM), which uses high-voltage sparks to vaporise the surface of the metal and gives a highly polished result. The mould also needs to include space for the ejection pins to remove the part after it's made and a route for the plastic to flow into the mould. In assembled products, the parts are naturally removed from the sprue during production.

Like any production technique, injection moulding has its own design considerations. Because the parts need to be extracted from the mould after they're formed, very sharp corners and completely vertical faces are best avoided. A slight angle, called the draft, from vertical allows for clean parting of the part and its mould, and consistent wall thicknesses avoid warping of the part as it cools. If you need the thicker walls for strength, an alternative is to use ribs to add rigidity without lots of additional plastic.

Some of the common techniques for achieving maximum strength with a minimum amount of material and also ways to mould mounting points for PCBs or screw holes for holding the assemblies together. The simplest moulds are called straight-pull and consist of the mould split into two halves. If your design needs to include vertical faces or complex overhangs, more complicated moulds which bring in additional pieces from the side are possible but add to the tooling-up cost. One way to reduce the tooling-up costs and also increase the production rate is to mould more than one part at a time. If your parts are small enough, you can replicate many of them on one mould or collect lots of different parts together.

In a process known as multishot moulding, you can even share parts of different colours on the same mould. With carefully measured volumes for each part, one of the colours of plastic is injected first to fill the parts which need to be that colour. Then the other colour is injected to fill the remainder of the mould. Obviously, there is a section of the mould cavity where the different colours mix, but with careful design, that is just part of the sprue and so discarded.

10.6 Certification

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

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.

The regulations that your device needs to pass vary depending on its exact functionality, target market, and the countries in which you expect to sell it. Negotiating through all this isn't for the faint of heart, and the best approach is to work with a local testing facility. They not only are able to perform the tests for you but also are able to advise on which sets of regulations your device falls under and how they vary from country to country.

Such a testing facility subjects your device to a barrage of tests (hopefully) far beyond anything it will encounter in general use. Testers check over the materials specifications to ensure you're not using paint containing lead; zap it with an 8KV static shock of electricity to see how it copes; subject it to probing with a hot wire—heated to 500 degrees Celsius—to check that it doesn't go up in flames; and much more.

Of particular interest is the electromagnetic compatibility, or EMC, testing. 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. Electromagnetic interference is the "electrical noise" generated by the changing electrical currents in circuitry. When generated intentionally, it can be very useful: radio and television broadcasts use the phenomenon to transmit a signal across great distances, as do

mobile phone networks and any other radio communication systems such as WiFi and ZigBee. The problem arises when a circuit emits a sufficiently strong signal *unintentionally* which disrupts the desired radio frequencies. All the tests are performed on a DUT, which needs to be the final specification for the entire product. As a result, the testing will most likely be a critical point in your delivery schedule, and any problems discovered will delay shipment while you iterate through a new revision to fix the issues.

Shenzhen BCTC Technology Co.,Ltd.
A,Floor 3, 44 Building, Tanglang Industrial Park B,
Taoyuan Street, Nanshan District, Shenzhen, China

Certificate of Compliance

Certificate Number: BCTC-150505076

Applicant	:	**Shenzhen Wisva Optoelectronics CO.,Ltd** Bld B, Shengxuan Ind. Park, Shiyan Town Bao'a District, Shenzhen, China 518108
Manufacturer	:	**Shenzhen Wisva Optoelectronics CO.,Ltd** Bld B, Shengxuan Ind. Park, Shiyan Town Bao'a District, Shenzhen, China 518108
Product	:	**LED strip light**
M/N	:	**HH-SAAABBBFXCCDEE-FFFF** HH: represent company name: Wisva; S: represent flexible strip;AAA:represent LED color R,G,B,Y,W,WW,RGB;BBB: represent led quantity per meter:30,48,60,72,78,96,120,204 240;X:0 represent non-waterproof,1 represent waterproof;CC: represent width of PCB board 8,10,13.5,15,20;D: represent color of PCB board W: white,Y: yellow,B:black;EE:represent working voltage:DC12V,DC24V;FFFF:LED type:3528,5050,3014,5630,5730,2835,335.
Test Standard	:	**IEC62321-1:2013**

The EUT described above has been consolidated by us and found in compliance with the council RoHS directive—2011/65/EU. It is possible to use CE marking to demonstrate the compliance with this RoHS Directive.

CE

Approved
May. 12, 2015

This certificate of conformity is based on a single evaluation of the submitted sample(s) of the above mentioned product. It does not imply an assessment of the whole product and relevant. Directives have to be observed.

Tel: 400-788-9558 0755-33019988 Http://www.bctc-lab.com Http://www.btc-lab.com

Figure 10-10: Sample Certificate of Compliance

For the EMC tests, the device is isolated in an anechoic radio frequency (RF) chamber to minimise the chance of external electromagnetic interference confusing the tests. The resultant test report is added to your technical file, which is referenced by your declaration of conformity. Assembling this is a requirement for certification and documents the key information about your device and the testing it has undergone. In addition to the test report, you need to gather together PCB layouts, assembly certificates, the certificates for any precertified modules that you have used, and datasheets for critical components. This information is all held in a safe place by the manufacturer in case the authorities need to inspect it.

For certain regulations you must also notify a specific, named body; for example, circuits that perform radio communication and so intentionally emit electromagnetic interference must be registered with the FCC when sold in the US. Such registration is in addition to issuing the declaration of conformity for self-certification. Because of the added complexity and

overhead—both administrative and financial—of some of the more involved directives, it is often wise to use pre-approved modules. You therefore can include a WiFi module (chips, antenna, and associated circuitry), for example, or a mains power adaptor, without having to resubmit to all the relevant testing. As long as you don't modify the module in any way, the certification done by its manufacturer is deemed sufficient, and you just need to include that in your technical file.

10.7 Costs

You have many things to consider as you move to higher volume manufacturing. Unfortunately, lots of them involve sizeable up-front costs. In fact, the further you get into the process, the less you will need your hardcore coding or critical design skills, and the more time you'll spend balancing cash flow and fund-raising. However, the upside is that you'll be able to get your *awesome* connected device into the hands of many more people and that's much more important. If you've raised enough money upfront, the task of managing the "manufacturing project" will be easier, but as with all projects, there will still be time scales and dependencies to look after.

You don't need to be able to wield a Gantt chart or have a PRINCE2 project management qualification, but *someone* on the team needs to keep an eye on how things are progressing; which things must be done before other tasks can proceed; and, peering further into the project timeline, spot and deal with problems before they become a roadblock for the rest of the team.

The time between ordering an item and its being ready to use can come as a bit of a shock. You are likely to encounter this situation first with PCB manufacture, where it can often take a couple of weeks from sending off the Gerber files to having something to solder up. Faster turnaround times are possible, but you tend to pay a premium.

The setup and tooling for processes such as injection moulding can be worse. Getting the tool machined and tested and then fine-tuned is likely to take a month or two. Sometimes during that process you're involved, but there also will be big chunks of time while you sit waiting for the work to be done. With careful planning, you'll be able to work on other parts of the project while you are waiting—maybe finishing the server coding or preparing the marketing materials—but you will still need to block out the time for it in your project plan.

Working out the likely cost for your project is a bit of a black art and obviously depends greatly on the complexity of what you're trying to achieve. Many of the online PCB services include a quoting tool, so even before the design is finished, you can get a feel for the likely price. You should be able to make a reasonable guess on the size of PCB you need and the number of layers it's likely to require; those are the main factors that the quoting tools use to work out the cost, plus extras if you want something out of the ordinary such as a different coloured solder mask. For assembling the PCBs, you should budget something around £150–£800 for setup costs—getting the solder paste stencil made, programming the pick-and-place machine, and so on—and then between 3 and 14 pence for placing each component. This doesn't include the cost of the components themselves. When buying components in small quantities, you will often notice that the price list also shows the cost if buying in greater bulk. And to get through certification, simpler devices being certified in fewer territories might get through certification

for around £2,000; more complicated designs, particularly those involving uncertified RF modules, could see costs 10 times that amount (£20,000) to get all the certifications in place.

Naturally, as the project progresses, you will get more accurate quotes as you talk to suppliers and get ready to place orders with them. This will let you update your BOM and cost spreadsheets and have the new information ripple through your plans.

10.8 Scaling up Software

Producing a physical *thing* as a prototype or as a manufactured product turn out to be two entirely different propositions. The initial prototype may well be of different size, shape, colour, materials, finish, and quality to what ends up on the shelf.

Yet software is intangible and malleable. There are no parts to order, no Bill of Materials to pay for. The bits of information that make up the programs which run in the device or on the Internet are invisible. The software you wrote during project development and what ends up in production will be indistinguishable to the naked eye. Yet, as with the physical form and electronics of the device, software needs to be polished before it can be exposed to the real world.

After looking at what is involved in deploying software—both on the embedded device and for any online service you have built—we look at the various factors that require this polish: correctness, maintainability, security, performance, and community.

10.8.1 Deployment

Copying software from a development machine to where it will be run from in production is typically known as *deployment*, or sometimes, by analogy to physical products, *shipping*. In the case of the firmware running on the microcontroller, this will be done once only, during the manufacture of the device. The software will be flashed to the device in a similar way to how you updated your prototype device during development.

For a simple device like the Arduino which usually only runs a single program, the process will be identical to that in development. For a device like a BeagleBone or Raspberry Pi which runs an entire operating system, you will want to "package" the various program code, libraries, and other files you have used in a way that you can quickly and reliably make a new build—that is, install all of it onto a new board with a single command. The software for an online service will tend to run in a single location. However, it will generally be visible to the whole of the Internet, which means that it will potentially be subject to greater load and to a greater range of accidental or malicious inputs that might cause it to stop working.

Finally, as a malleable and user-facing software product, there is always the possibility of updating the code to add value by introducing new features or improving the design. As a result of all of this, it is not merely possible, but necessary, to update the online software components

more regularly than those on the device. Having a good deployment process will allow you to do this smoothly and safely.

10.8.2 Correctness and Maintainability

As a publicly available product, your software has to do what you claimed it would, and do it efficiently and safely. Testing your code before it is deployed is an important step in helping to avoid such a situation, and your chosen language and development framework will have standard and well-understood testing environments to help ease your task. As it lives in a central place, the server software is easy to update, either to fix bugs or to introduce new features. This is a real boon for web applications, as it removes an entire class of support issues. The embedded code in the device, however, is particularly important to test, as that is the hardest to update once the product has been sent out to the users. It may be possible, given that the devices will be connected to the Internet anyway, for the code to be updated *over-the-air,* and that is one of the selling points of the Electric Imp platform, for example. However, it should be approached with some caution: what if a firmware update itself caused a failure of a home heating system? And as Internet of Things products become more integrated into our lives and our homes, they become a tempting target for hackers. If your device can update its own code, the channels for delivering and authenticating any updates must be rock solid, lest they be compromised.

10.8.3 Security

Apart from securing the communication between client device and web server, you should consider the server itself; as an always-on, always-connected device visible to the whole Internet, it is the most obvious target. Following are some of the more important guidelines:

- Make sure that your servers are kept up-to-date with the latest security patches, are hardened with the appropriate firewalls, and detect and mitigate against password hacking attempts and rootkit attacks.
- User passwords should never be stored in plain text. If your database were ever compromised, an attacker could easily log in as any user. Passwords should be encrypted with a secure algorithm which is not known to be trivially cracked, and “salted” for additional security.
- Never simply trust user input. Check that anything that is entered into a web application fits the type of data you expect, and refuse or clean anything which doesn’t.
    - Although you may think input from your connected devices would be okay (because you wrote the code), it is possible that it has been compromised or an attacker may be “spoofing” it. In particular, be wary of passing user input to your database without checking it, or including unfiltered user input in your HTML pages, as this could allow a cross-site scripting (XSS) attack. Strip out all HTML tags or escape the output.

- Be aware of cross-site request forgery (CSRF) attacks from other malicious or compromised websites.
    - For example, if one of your users browses a bad site which uses JavaScript to open http://some.example.com/heating?switch=off on your site, and the user is already logged in, he may come home to a cold house.

10.8.4 Performance

The first thing many people think of when considering scaling up software is whether it will be fast enough and handle a large number of users, but in fact the other factors are usually far more critical. If your web service is running on a modern framework, it should be easy to scale up by deploying the code onto a more powerful machine, or by running multiple servers, with a front-end server or proxy managing the load. In general, you should concentrate on running your web application with appropriate standard infrastructure which is good enough for most problems. If you're lucky enough to have so much traffic that you need to scale, you should always optimise using this algorithm:

- Identify that you actually have a problem.
- Measure and profile the tasks which are slow and identify the problem.
- Fix that problem. The web community does a good job in sharing best practice in how to address such problems, so you often find that others have trodden the path before and documented their tried-and-tested solutions.

10.8.5 User Community

Whether you are launching a mass-market commercial product or an open-source community effort, your project will be successful only if people actually use it. While few large companies can boast Apple's famous commitment to "insanely great customer service", a small, focused startup can often match them for responsiveness and enthusiasm. At a minimum, you need a support email or a bug-reporting tool, but you will probably have a presence on various social media forums (Twitter, Facebook, and the like). As well as responding to queries, however, you should think about the user experience before launching: does your website have documentation, introductory videos, and tutorials? Blogging regularly about product development and related topics helps build a readership of users, both current and potential. Finally, some form of forum, mailing list, or chat room lets users support each other, which reduces your workload, but more importantly helps to build up a community and expertise around your product.

10.9 Summary

- Building physical products is hard, and products that connect with the Internet take up the complexity another notch. However, the finished items elicit a reaction that is hard to match with a website or an app.
- Moving to manufacturing involves many skills and tasks that are quite different from those you used to bring the idea to life, and it can feel quite daunting when you realise all that needs to be done.
- However, this process is something that the world has been perfecting for over a hundred years and even in the electronics industry is a few decades old. So if you find the right partners, it needn't be as difficult as you fear.

10.10 Review Questions

1. What options are available when deciding to scale up production of an IoT device?
2. Write a short note on designing kits.
3. Explain the process of designing printed circuit boards.
4. What are the software choices when designing printed circuit boards?
5. Explain the process of testing for a printed circuit board.
6. Explain the issues faced during custom casing or other sub-assemblies with an example.
7. Why is certification required/important for Internet of Things products?
8. What different certifications and tests are required for Internet of Things products?

Chapter 11
Ethics

Technology often creates new possibilities with one hand and new problems with the other. The society has generally two kinds of people, pessimistic and optimistic. We find more people to be more pessimistic rather than optimistic. The reason may be that the rejection of the new and the different is deeply embedded in societies and can be phrased in more visceral terms.

Take a moment to reflect on your own stance to technology in general. It may be that you are generally anti- or generally pro-technology. It is also possible that you are optimistic about some classes of technology (Internet, space travel) but pessimistic about others (genetically modified food, wind energy).

In many traditional disciplines, there are courses on ethics—for example, in university courses on engineering or computer science. The practitioners of the Internet of Things, as we saw in the beginning of the book, may come from varied backgrounds, from these technical fields, through design, to the fine arts. In this chapter, we do not present an entire course on formal ethics. Rather, we look at several aspects, many of which are related to such an ethics syllabus and consider them in relation to the Internet of Things. We look at many extreme and challenging ideas, some of which may seem like science fiction or may be politically uncomfortable.

11.1 Characterizing the Internet of Things

Let us start by summarizing what the Internet of Things is, to get a handle on what particular changes it can bring to humanity's relationship to the "good life".

We've stressed the fact that a certain powerful technology (computer chips) is suddenly cheap and plentiful. This is by no means a new observation, nor the first technology that has undergone this explosive transition. This availability of technology brings certain abilities within the reach of not just the powerful but the ordinary citizen. Examples can be found in the fields of publication, transport, and communication.

The advances in the Internet of Things are also primarily related to communication, but now allow the publication and transmission of vast streams of data, from the social to the environmental, without needing the permission or expertise of a technological or political elite.

The adage that "form follows function" applies primarily to the physical usage of the "Thing", its affordances, sensors, and actuators, and only minimally to its digital communications. This leads to objects that can look innocuous but have arbitrary and potentially unexpected capabilities. Connecting the Internet to the real world allows both your physical actions to be made public and, in the opposite direction, for events on the Internet to affect your environment. Applying this bidirectional communication to Things can lead to features that interact with the concept of privacy. When you switch on your Good Night Lamp, the "little lamp" at your mother's bedside will also turn on, letting her know you are home. When you leave the office,

the WhereDial at home turns to let your partner know you're travelling. We have repeatedly noted that the Internet of Things is made up of Physical object + controllers, sensors, and actuators + Internet service. Each of these aspects has a part to play in the ethical issues specific to the Internet of Things, and we refer to them in the sections that follow.

11.2 Privacy

The Internet, as a massive open publishing platform, has been a disruptive force as regards the concept of privacy. Everything you write might be visible to anyone online: from minutiae about what you ate for breakfast to blog posts about your work, from articles about your hobbies to Facebook posts about your parties with friends. There is a *value* in making such data public: the story told on the Internet becomes your persona and defines you in respect of your friends, family, peers, and potential employers.

But do you always want people to be able to see that data? Do you want not just your family and friends but also companies, the government, and the police to be able to see information about you, forever? A common argument is "if you've got nothing to hide, then you've got nothing to fear". There is some element of truth in this, but it omits certain important details, some of which may not apply to you, but apply to someone:

- You may not want your data being visible to an abusive ex-spouse.
- You might be at risk of assassination by criminal, terrorist, or state organizations.
- You might belong to a group which is targeted by your state (religion, sexuality, political party, journalists).

More prosaically, you *change* and your persona changes. Yet your past misdemeanours (drunken photos, political statements) may be used against you in the future. 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 don't 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 you would have had to take a photo, develop it, take the printed photo to your computer, scan it, and then upload it or take your digital camera to the computer and transfer the photo across via USB, now you can upload that compromising photo, in a single click, while still drunk. The ability to do something is present in a defined context (the personal) rather than locked in a set of multiple processes, culminating in a general-purpose computer. Even innocuous photos can leak data. With GPS coordinates, produced by many cameras and most smartphones, embedded into the picture's EXIF metadata, an analysis of your Flickr/Twitpic/Instagram feed can easily let an attacker infer personal details.

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 RunKeeper app on your smartphone. This data is incredibly useful to keep track of your 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 your house is and get information about the times of day that you are likely to be out of the house.

The idea of people knowing where you are can evoke strong emotions. Yet the idea of knowing that your loved ones are safe is a similarly deep-seated human emotion. To the extent that you allow your location to be shared with people you've *chosen* to share it with, there is no infringement of privacy. But the decision to give your mother a Good Night Lamp might seem less sensible months later when you arrive home late at night. Or you might regret giving your partner a WhereDial if later she becomes jealous and suspicious of your innocent (or otherwise) movements.

Even if these devices are themselves respectful of your privacy, their security or lack thereof might allow an attacker to get information. For example, if it were possible to read an IP packet going from the goodnightlamp.com servers to a household, could you find out that the associated "big lamp" had been switched off? Even if this packet is encrypted, could an attacker infer something by the fact that a packet was sent at all? These risks are to be considered very carefully by responsible makers of Internet of Things devices.

So far, we've looked at devices that you, as an individual, choose to deploy. But as sensor data is so ubiquitous, it inevitably detects more than just the data that you have chosen to make public. For a start, we saw previously that many "things" have little in their external form that suggests they are connected to the Internet. When you grab an Internet-connected scarf from the coat rack or sit on an Internet-connected chair, should you have some obvious sign that data will be transmitted, or an action triggered?

Urbanist and technologist Adam Greenfield has catalogued interactive billboards with hidden cameras which record the demographics of the people who look at them and vending machines which choose the products to display on a similar basis. He comments that, as well as being intrusions into public space, these objects are not just what they seem to be. Rather, they have an agenda of "predicting behaviours and encouraging normative behaviours", which is not transparent.

Perhaps a Thing that is implemented digitally will not give the subtle cues that its analogue counterpart used to give. Let us consider the electricity smart meter. The real-time, accurate measurement of electricity has many admirable goals. Understanding usage patterns can help companies to produce electricity at the right times, avoiding overproduction and optimizing efficiency. The aggregate data collected by the companies is useful for the noble environmental goals like conserving energy...but how about individual data? If you could mine the data to see subtle peaks, associated with kettles being switched on for tea or coffee, perhaps you could infer what television programmes a household watches. If there are four longer peaks in the morning, this might suggest that four family members are getting up for an electric shower before going to school or work. Now what if you triangulate this data with some other data—for example, the water meter readings?

The idea of analysing multiple huge datasets is now a reality. There are smart algorithms, and there is the computing power to do it. By combining both ends of the long tail (the cheap, ubiquitous Internet of Things devices on the one hand and the expensive, sophisticated, powerful data-mining processors on the other), it is possible to process and understand massive

quantities of data. How powerful this ability will be may well depend on what data you have available to compare. It is very important to note that even aggregate data can "leak" information. Some very interesting questions can be raised about this: should companies be prevented from trading data with each other? Should there be legal limits to what data can be kept or what analyses performed on it? Or do we have to think the unthinkable and admit that privacy is no longer possible in the face of massive data combined with data mining?

At the Open Internet of Things Assembly 2012 in London, there was a great deal of discussion about who owns sensor data. The electricity companies? The householder who has signed the electricity contract? Or all the stakeholders in the house? If you visit a friend's house, data about *you is* collected and "owned" by someone else. This might mean that in future you would be a stakeholder in this valuable data too. As sensors such as CCTV cameras, temperature meters, footfall counters, and Bluetooth trackers are installed in public and private spaces, from parks to shops, data about you is collected all the time.

The term "data subject" has been coined for this purpose. Although you may not own the data collected, you are the subject of it and should have some kind of rights regarding it: transparency over what is collected and what will be done with it, the access to retrieve the data at the same granularity that it was stored, and so on. While futurology is fun, it's also hard. As the visionary computer scientist Alan Kay famously said in 1971, "The best way to predict the future is to invent it".

It is clear that we are leaking ever more data onto the online world, and some of this data will be vital to dealing with human crisis of the near future. Rob van Kranenburg, founder of the think tank Council, predicts that the requirement to smoke out inefficiency, for the survival of our species, will lead to a state of post-privacy. Do we have the "courage to live in the light"?

11.3 Control

Some of the privacy concerns we looked at in the preceding sections really manifest only if the "data subject" is not the one in control of the data.

The example of the drunken photo is more sinister if it was posted by *someone else,* without your permission. This is a form of *cyberbullying,* which is increasingly prevalent in schools and elsewhere. While the technology itself doesn't cause any controlling behaviour, it could easily be applied by a spouse/parent/employer in ways that manifest themselves as abusive, interfering, or restrictive, in more or less sinister ways.

In the case of an employer, we are bound to see cases in the future in which a contractual obligation is needed to share data collected by some Internet of Things device. Already, companies and organisations are looking at mashing up data sources and apps and may start to offer financial incentives to use Internet of Things devices. For example, reductions in health insurance if you use an Internet-connected heart monitor, have regular GPS traces on a run-tracking service, or regularly check in to a gym. High-end cars already have Internet connected tracking and security systems which may even be a requisite in getting insurance at all.

As with questions about privacy, there are almost always good reasons for giving up some control. From a state perspective, there may be reasons for collective action, and information

required to defend against threats, such as that of terrorism. The threat of one's country becoming a police state is not merely a technological matter: institutions such as democracy, the right to protest, free press, and international reputation should balance this. But, of course, with the processing power and information available now, there is a much greater temptation and capability to become an effective *Big Brother* regime.

Now that surveillance equipment is cheap, and the processing power required to *analyse* the mountain of data produced by this equipment gets ever more accessible, the simple logistical difficulty of an absolute dystopia vanishes.

Commentary on the mourning at Kim Jong-Il's death noted that the, no doubt genuine, grief of many may also have been amplified by the social and political compulsion to express the "histrionics of grief ".

In a world where pervasive body-blogging is becoming feasible, it is conceivable that not only the *display of grief but also the physiological* symptoms of it could be verified. It is not only authoritarian states such as Iran and China which are intent on controlling their Internet but also democratic ones. The US, UK, Canada, France, and others have already enacted various laws to give the state and its favoured corporations greater control over its citizens' use of the Internet, and every month one hears news of other suggested legislation which, to a technical specialist, may seem not just badly thought out and unworkable but also immensely dangerous.

Just as the printing press gave the state a greater degree of control via propaganda, the Internet gives hitherto unknown possibilities for propaganda and monitoring. Even in a democratic state, with the readily accessibly technology in place, it is up to the institutions that safeguard democracy and the will of the people to become the main bulwark against the threat of authoritarian control. Of course, it may not be "the State" that profits from the control but corporations. Companies have the expertise and the technology to interact with the Internet. This is particularly true of the Internet of Things, which has largely been driven by monitoring and logistics concerns within large businesses.

11.3.1 Disrupting Control

The other major possibility that Eaves suggests is that "The Internet Destroys the State". This is also a hard and uncomfortable scenario to imagine. However, toning down this idea a little, we can see a more likely one of "the Internet" fighting back against an attempt by the state or corporations to co-opt it.

When we refer to a technology as "disruptive", we mean that it affects the balance of power. If one of the fears about the Internet of Things is that it will transfer power away from the citizens, the subjects of technology, to the elite, then perhaps it can also be used to redress that balance. One extreme example of this would be how surveillance and fears of the Big Brother state (CCTV cameras, remote-controlled helicopter-drones) might be mitigated by "sousveillance". Here, activists might have compromised public cameras, or perhaps installed additional spy cameras, routed through self-healing networks in people's homes, hidden in public spaces, flying in the airspace, or even crawling the sewers.

11.3.2 Crowdsourcing

One fascinating feature of modern Internet life is "crowdsourcing", from knowledge (Wikipedia, et al.) to funding projects (Kickstarter, Indiegogo) to work (Mechanical Turk). In the Internet of Things world, this concept has manifested itself in sensor networks such as Xively. Founder Usman Haque has said that their original intent wasn't simply "making data public" but also letting "the public making data".

Figure 11-1: Crowdsourcing to seek wisdom

Governments and companies simply do not and cannot have a monopoly on all recording of data: there are infinite combinations of data sources. Choosing which data to record is a creative and engaged act, as well as, perhaps, a political one. After the Fukushima Daiichi nuclear disaster, there were fears that insufficient information was available to track the spread of the leaked radioactive materials. Many hackers around the world built Geiger counters, and Xively was a focal point for Japanese engineers to publish their data to. Perhaps the Japanese government or the management of the Fukushima plant would have provided that kind of accurate, widespread data if they could. But power or financial interests might have worked against this.

Andrew Fisher, a technologist with interests in big data and ubiquitous computing, has written persuasively about a quiet revolution of the "sensor commons", his term for this collaborative voluntary effort to provide environmental data. Fisher's original definition observed five critical requirements for a sensor commons project. It must

- **Gain trust:** Trust is largely about the way that an activist project handles itself beyond the seemingly neutral measurements; understanding local issues, being sensitive about the ways that the sensor network itself affects the environment, engaging the public with accessible and readable information about the project, and dealing with the local authorities to get access to the systems the project wants to measure.

- **Become dispersible:** Becoming dispersible means spreading the sensors throughout the community. Getting mass adoption will be easier if the proposed sensors are inexpensive and if the community already trusts the project.
- **Be highly visible:** Being visible involves explaining why the project's sensors are occupying a public space. Being honest and visible about the sensor will help to engender trust in the project and also advertise and explain the project further.
- **Be entirely open:** Being open is perhaps what distinguishes the sensor commons from a government project the most. The openness makes up for this because all the facts about the devices and the possible errors are admitted upfront and can be improved by anyone in the community. The project should also have an API and permissive licensing so that the community can choose to do different, complementary things with the data from the network.
- **Be upgradable:** Finally, the project should be designed to be upgradable, to enable the network to remain useful as the needs change or hardware gets to the end of its working life.

While Fisher writes specifically on sensor networks, the principles he proposes are, we suggest, relevant to consider for any ethical project in the field of the Internet of Things.

11.4 Environment

We have already touched on several environmental issues in the preceding sections, and we'll come back to the themes of data, control, and the sensor commons. First, let's look at the classic environmental concerns about the production and running of the *Thing* itself.

11.4.1 Physical Thing

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 the manufacturing plant to the customer. It's 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 moulding. Calculating the energy costs for manufacture is harder. You may need to consider other environmental factors, such as emissions produced during normal operation or during disposal of the object.

11.4.2 Electronics

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. Considering the potential cost savings, even a responsible manufacturer may find it reasonable to offset the extra carbon emissions. More worryingly, many electronic components rely on

"rare earth minerals" (REMs) which have been extracted in China or from other locations worldwide.

The mining process must be managed properly; otherwise, 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 the raw material from mine to refinery to manufacturer has its own carbon cost too. Perhaps little can be done to reduce this environmental cost currently, but as companies in other countries start to mine REMs, being aware of the parameters may allow improvements in purchasing choices.

Every electronically enhanced Thing that you produce will incur these costs and will also need to be powered to run. Speaking to the Internet (via WiFi or 3G) is a major part of the power cost of running an Internet of Things device. Anything that can be done to reduce this cost will make the devices more efficient. Choosing suppliers of WiFi chips wisely and following the low-power IPv6 developments (6LoWPAN) closely will be helpful here.

11.4.3 Internet Service

As Nicholas Negroponte (founder of MIT's Media Lab) preaches, "Move bits, not atoms" (*Being Digital,* Vintage, 1996). In the digital world, moving data rather than physical objects is faster, is safer, and has a lower environmental cost. Of course, "data" doesn't exist in the abstract. The stone tablets, parchment scrolls, and libraries of paper books or microfiche that have historically been used to store analogue data always had their own environmental cost.

Now, running the Internet has a cost: the 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. As well as the cost of transferring the data across the Internet, running your own web server uses power. 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. So, of course, will having more customers, although of course we won't go as far as suggesting that you cut down on those!

11.5 Solution

Compared to a simple, physical object, an instrumented Internet of Things device does seem to use vastly more resources in its production, daily use, and waste disposal. Considering our starting point—that this kind of instrumentation is now cheap enough to put *everywhere*—it seems as though the mass rollout of the Internet of Things will only contribute to environmental issues!

From a more optimistic point of view, it's also true that the realisation that the number of Internet-connected devices will be exploding in the coming years is spurring massive research into low-power efficient chips and communications.

Figure 11-2: IoT as part of the solution

11.5.1 The Internet of Things as part of the solution

Gavin Starks, former CEO of amee, has spoken convincingly of instrumenting the world precisely to *save it.* While Starks's lectures are timely and necessary, as a good hacker, he prefers to do something about the problem: try to solve it through technology, information, and awareness. We already discussed distributed sensor networks as a social and political act: the potential for global environmental action is also massive.

A UN Environment Programme report warns that the lack of reliable and consistent time-series data on the state of the environment is a major barrier to increasing the effectiveness of policies and programmes. If community-led sensor networks can help supplement government and international science measurements, then we should be doing everything we can to help.

Instrumenting production lines, home energy usage, transport costs, building energy efficiency, and all other sources of efficiency might seem extreme, but it may be a vital, imperative task. Other technologies which aren't principally linked with the Internet of Things will also be important. Instrumenting the supply chains, measuring to be certain that new methods really

are more efficient, and reducing inefficiencies by automation could well use Internet of Things solutions to help measure and implement the solutions. The Internet of Things could become a core part of the solution to our potentially massive environmental problems.

In the face of these suggestions—collective sensor networks and massive business process engineering not for profit but for environmental benefits—you might wonder whether these calls to action amount to critiques of capitalism. Capitalism's great success has always been how it routes around problems and finds a steady state which is the most *efficient to the* market. There is no reason why capitalism as-could-be should not be part of the process of striving towards efficiency on an environmental as well as monetary level.

Van Kranenburg also makes alternative, starker proposals: not only may privacy become obsolete, but even those currently personal possessions such as cars might also become communal, through the increasing move from ownership to rental models. As resources become ever scarcer, a greater percentage of income might be spent on covering rental of all goods—cars, food, possibly even housing. This kind of futurology leads to scenarios such as the death of money itself: a fixed proportion of income to rent needed services from a commercial supplier is more or less indistinguishable from taxation to pay for communal services. Whether the death of privacy and of money sounds like utopia or dystopia to you, it is worth considering the impact tomorrow of the technologies we implement to deal with the problems of today.

As a counterpoint to these messages of doom, Russell Davies of London's Really Interesting Group (RIG) often tries to bring the discussion of Things back to *fun.* Although this may not sound as engaged or political an attitude, by looking for the unintended uses for technologies, the end users, rather than the political elites, can turn them into platforms for human expression. Similarly, the World Wide Web was originally conceived to share academic papers but has taken on the roles of brokering business on the one hand and publishing pictures of kittens on the other without breaking its step. The Internet of Things will also, if we let it, become a platform for whatever people want it to be. Although this may be less *important than* saving our species from environmental disaster, perhaps it is no less ethical in terms of asserting our humanity through, and not simply in spite of, the technology that we might have feared would dehumanise us.

11.5.2 Cautious Optimism

Between the tempting extremes of technological Luddism and an unquestioning positive attitude is the approach that we prefer: one of cautious optimism. Yes, the Luddites were right—technology *did change the world* that they knew, for the worse, in many senses. But without the changes that disrupted and spoilt one world, we wouldn't have arrived at a world, our world, where magical objects can speak to us, to each other, and to vastly powerful machine intelligences over the Internet.

It is true that any technological advance could be co-opted by corporations, repressive governments, or criminals. But technology can be used socially, responsibly, and subversively, to mitigate against this risk. Although the Internet of Things can be, and we hope will always be, *fun, being aware of the ethical issues around it, and facing them responsibly,* will help make it more sustainable and more human too. As a massively interdisciplinary field,

practitioners of Internet of Things may have an opportunity or responsibility to contribute to providing moral leadership in many of the upcoming ethical challenges we have looked at.

We should remember an important lesson on humility from Laura James's keynote at the OpenIoT assembly: *Don't assume you know it all. The [I]nternet of [T]hings is interdisciplinary and it stretches most of the individual disciplines too. You will need help from others. Be ready to partner with other organisations, collaborate with people from very different backgrounds to you.*

When designing the Internet of Things, or perhaps when designing *anything,* you have to remember two contrasting points:

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

This tension underscores the difficulty of trying to figure out overriding concerns about any complex area, such as the Internet of Things!

11.5.3 The Open Internet of Things Definition

The Open IoT Assembly 2012 culminated in the drafting of the "Open Internet of Things Definition". An emergent document, created after two days of open discussion, it seeks to define and codify the points of interest around the technology of the Internet of Things and to underscore its potential to "deliver value, meaning, insight, and fun". A particularly interesting consensus in the definition was that, even though the Data Licensor (usually the person who has set up the sensor or paid for that data) should quite reasonably own the data from that sensor, some rights should also apply to individuals whose data is recorded (the Data Subjects). They *must* be granted licence to any data that regards them and *should* be allowed to license the anonymised aggregate data for their own purposes. We can summarize the main goals of the definition as follows:

- **Accessibility of data:** As a stated goal, all open data feeds should have an API which is free to use, both monetarily and unrestricted by proprietary technologies with no alternative open source implementation.
- **Preservation of privacy:** The Data Subjects should know what data will be collected about them and be able to decide to consent or not to that data collection. This is a very strong provision (and most likely unworkable for data which is inherently anonymous in the first instance) but one which would provide real individual protection if it were widely followed. As with any information gathering, "reasonable efforts" should be made to retain privacy and confidentiality.
- **Transparency of process:** Data Subjects should be made aware of their rights—for example, the fact that the data has a licence—and that they are able to grant or withdraw

consent. In addition, where the data is collected from a *public space,* the public should get a right to participate in decision making and governance of that data.

The importance placed by these principles on data is unsurprising: the Internet of Things brings the gathering and collation of data into the everyday world and has real consequences on individual privacy and power.

11.6 Summary

- Technology often creates new possibilities with one hand and new problems with the other.
- We have looked at the capabilities of connected devices and noted how new technology fosters new business models. Yet just as these new possibilities were born of the defining characteristics of the Internet of Things, so the ethical concerns raised spring from the same essence.
- Inherent in the physical "thing" produced are environmental concerns: the carbon cost of manufacture and of the raw materials required for its complex electronics. Then there is a moral question over whether the workers tasked to construct it are treated fairly.
- Taking complex electronics off the desk and putting them into the fabric of the world and your personal possessions allows sensors to read sensitive data on an unprecedented scale.
- Combined with the ability to aggregate this data via the Internet, it becomes possible for companies to monitor their customers through their telephones, electricity meters, or their air fresheners.
- Similarly, government bodies can monitor their citizens in locations and ways that might not be obvious. This raises questions of transparency and privacy and the issue of our right to have access to data that concerns us.
- As technologists, we don't have to see these concerns as prophecies of doom but as a reminder to consider our responsibilities while we are engaged in designing and building our magical objects to delight our customers and bring us profit or amusement.
- We have also touched on how the technology of the Internet of Things can act as counterbalance to its own potential flaws.
- The widespread sensors and powerful processing can lead to better information and possible mitigations to our impact on the environment.
- The accessibility of the technology permits crowdsourced sensor networks which tips the balance of civic power to let citizens make their own informed decisions about issues that concern them.
- As the field of the Internet of Things develops, we will see more clearly the next most immediate steps to take. Perhaps some of the futures we've envisaged will come to pass, and perhaps they won't.

- With a strong understanding of the ethical decisions we need to make as technologists, we should be ready to help guide the Internet of Things in such a way that it need not become an invasive, oppressive instrument but instead a framework in which we can learn better what it is to be human.

11.7 Review Questions

1. How can privacy be an issue in Internet of Things devices, explain with suitable examples.
2. What are the reasons for giving up some control for IoT devices data.
3. What is “sensor commons” project? What are the critical requirements for a sensor commons project.
4. What are the environmental concerns about producing and running an IoT device?
5. How can Internet of Things be a part of the Solution to reduce environmental waste?
6. Write a short note on cautious optimism as solution for IoT.
7. What is “Open Internet of Things Definition”? What are the main goals of the definition?