

YOUR SITUATION AWARENESS

*How digital technology can make it
better*



Karl Jeffery and
Dimitris Lyras
September 2017

Published by Software for Domain Experts Ltd, 39-41 North Road, London, N7
9DP, UK. Tel +44 208 150 5292. E-mail jeffery@softwarefordomainexperts.com

Copyright Dimitris Lyras and Karl Jeffery

Your brain is the result of billions of years of evolution.

You use it to be aware of and to understand situations. This is how you work and stay alive.

But the society around you – and your working environment – doesn't usually understand this.

If you have a typical 2017 job, your working environment will be designed around helping you complete tasks.

We know why this is – as far as your employer is concerned, you are there to get stuff done. And your employer is probably choosing the digital tools which you are given to get your work done. So the digital tools are designed around helping you get stuff done.

And you do want to get stuff done. But in order to get stuff done, you need to understand what is going on around you in as much detail and depth as possible.

So your digital tools are not being as helpful as they could be, and you are not as comfortable or productive as you could be.

And so our companies, organisations and all of society are not as good as they could be.

This book aims to show how the situation could be improved.

We want to show what the world would look like if everyone was supported in their work by better tools – which would help them maintain situation awareness – and how to get there.

Table of Contents

| | |
|---|----|
| Your situation awareness | 1 |
| How digitech could work in a hospital..... | 2 |
| Captain Taskmaster – and why it may never work..... | 14 |
| Musicians – and working with situation awareness..... | 16 |
| Situation awareness in different roles | 21 |
| Different sorts of situation awareness..... | 29 |
| How to build and sell situation awareness software | 33 |
| Captain Taskmaster hits back..... | 39 |
| How to beat Captain Taskmaster with better digitech..... | 42 |
| Common digitech myths which won't help with situation awareness | 51 |
| Don't push Captain Taskmaster | 58 |
| But that doesn't mean to stop trying | 60 |

HOW DIGITECH COULD WORK IN A HOSPITAL

Let's start with some illustrations of how digitech might work – if it was designed for nothing other than supporting people's situation awareness, using some examples of roles in a hospital.

Emily, Camilla, Andrew and Mark are recent graduates. Emily is a nurse, Camilla is a junior doctor, Andrew is a purchasing manager and Mark is a junior manager.

Emily, the nurse

Emily, a nurse, is starting her night shift in a hospital ward. She has 30 patients in the ward. There are certain tasks she needs to do at regular intervals, like administer medicines or take blood pressure. She also must be available to respond to requests from patients, or concerns. She has to monitor how well patients are improving.

Someone just visiting the ward might just see 30 people asleep. Emily sees people who need to be fed, given medicines, checked up on, and who might ask for help going to the toilet – also people who might demand more time than she can allocate, or might suffer in silence unless she checks up on them. She also needs to spot problems emerging.

YOUR SITUATION AWARENESS

Emily has all the information she needs to plan her next task and carry it out on her smartphone. There is a computer system which takes all the relevant information about patient monitoring, tasks completed, reports from the previous shift, current requests for help and current high priority demands (including emergencies). It makes a suggestion of the top 5 uses of her time right now. Because the computer analytics is never perfect, the final decision is made by Emily, not the computer.

Emily can also adjust the software herself. For example, if a patient makes an urgent request which turns out to be not very urgent, she can program the software to automatically reduce the 'weighting' of that patient's request next time, as most nurses would probably do if they were not supported by smart phone tools.

If Emily gets behind on a scheduled activity (for example if she needs to check a certain patient's blood pressure every 4 hours), that task can gradually rise up her priority list, until these scheduled tasks can take priority over a non-urgent request. But if Emily disagrees with the computer's calculation of her priorities, she can reduce the weighting.

The phone does not bleep warnings at her unless it is a real emergency, because this can add to her stress levels – but it assumes she will be checking the screen every 10 minutes or so, when she decides what her next task is going to be.

Emily also wants to improve her performance, and she sets herself goals. She needs to balance responding to urgent requests with doing all the 'scheduled tasks' she needs to do during the shift. She can monitor how well she has done that. She is worried about forgetting some quiet patients because so much time goes to the noisy or demanding ones, and the software can tell her whether that is actually happening.

The app can also monitor her fatigue levels from the way she walks, or how many steps she has made, and make sensible recommendations when she should have a rest – and it can spot when her movement patterns change quite strongly indicating she may be dangerously fatigued.

The software can also gently monitor her stress levels, although it doesn't share the stress data with her every day because she doesn't find it very helpful. But she can check it when she wants. She is aware that her decision making and personal interactions get weakened when her stress level rises, and is trying out different ways to monitor them, including her sleep patterns before the night shift, whether she stops problems escalating or upsetting her during the shift.

Camilla, the junior doctor

Camilla is a junior doctor in a hospital. She has to make her way around the wards and do a series of checks on patients, and also be available if something bad happens.

She has a smart phone app which tracks her movements and makes sure she doesn't miss anyone. When she is about to see a patient it can present her with information she might need to know, such as what happened on the last shift. If the patient is being monitored with electronic devices, and there is some pertinent data, it appears on her screen. If there is something urgent, or the possibility of something urgent, it will appear at the top of her priority list, or her phone will alarm if she needs to suddenly run to see something. And if she has a problem beyond her ability, or where she needs help, she can automatically alert the most appropriate person.

The hospital also has many semi-retired ex doctors who are happy to be available to help and advise, even in the middle of the night. At any time, Camilla can speak to someone with appropriate experience, already knowing that the person has agreed to be contacted at this time.

Camilla wants to continually get better, and so sometimes she has after shift review discussions with an experienced doctor (who may be semi-retired) about the decisions she made and what she had to do. She has certain areas she wants to improve on, like her bedside manner, expeditiousness or accuracy of diagnosis, and she has set up different apps to show how she is doing in these areas.

Her phone app also monitors her stress and fatigue levels, and if necessary informs a hospital manager that she might benefit from some time off.

Camilla can also, if she wishes, get a sense of the quality of her decision making and judgement, by bringing up details of decisions she made 3 months ago, and what happened after that, and whether it turned out to be a good decision. This way, she can continually improve.

Andrew, the purchasing manager in the hospital

Andrew is a purchasing manager for blood analysis technology in the health service. His role is to find the items which are fit for purpose at the lowest cost. He also gets reports about all the purchases he made six months ago, showing how well they actually worked, and what the people who use them every day really think. How well did he get the balance right between cost and suitability? Were there problems from a manufacturer which claimed to meet the criteria, but actually didn't, and then blamed someone else for this? He continually learns how to do it better.

Andrew may want to keep track of how satisfied customers were with what was purchased, six months, one year ahead for major equipment and keep metrics of what went wrong so he can continually improve.

Mark, the junior hospital manager

Mark is a junior manager of a hospital. He's set specific areas where he wants the hospital to improve its performance, including reducing waiting times, making sure beds are occupied by people who need to be in a hospital, making sure the operating theatre is working to full capacity. But his software tool goes much further than just telling him the numbers. It can help him try out different approaches to improving the situation – and helps him to learn what is working well and what isn't.

Working out where a hospital is stressed and where it is on the edge of collapse is not easy – nearly all hospitals are stressed. But he's put together some digital models to get a better understanding of what is going on. He builds his own software tools to look at the data in different ways, creates his own visualisations so she can understand.

He can get an overview picture of how many people are waiting in the accident and emergency department, where the queuing times have got particularly long, and where the wards are under particularly high levels of stress and where possible, re-allocate resources.

He knows that the pressure for beds is higher in the winter, when more people are sick, and can plan less urgent treatment accordingly.

He can get a sense of the stress levels of individual staff members by monitoring their workloads and any particularly stressful situations they needed to deal with, and then perhaps recommend an extra rest day at appropriate times.

He can see where recruitment problems are happening or emerging and make plans for them, advising other departments.

He can see where people are struggling with poorly built IT or old IT systems and make recommendations for upgrades.

He doesn't have much financial resources, but he can see that there are other ways to maintain staff morale which don't involve spending money - for example the biggest motivator is that people feel that they are learning and becoming better at their job. It is not expensive to put together staff training workshops and mentoring schemes, including with semi-retired hospital staff who are happy to come and contribute. This is all carefully matched with what people need to know now, and letting people slowly develop.

None of this exists

Actually none of this exists.

Emily, the nurse, has some software to keep track of tasks – lists of jobs to do, shifts to fill, which nurse is filling them. The hospital has a range of alert and bleeper systems – essentially for passing a task onto someone else.

Camilla, the junior doctor, just sees a string of patients for 10-minute slots, diagnoses them to the best of her ability and then writes notes in a computer when she's done.

Andrew, the purchasing manager, has software to send out requests to suppliers, gather together what he receives back and see if they match his criteria and pick one – and he never knows how well the stuff works.

YOUR SITUATION AWARENESS

As for Mark, the junior manager, his role doesn't even exist. The hospital management software does nothing more than put stuff into databases and taking it out of databases - so he has software for managing wards, appointments, consultations, electronic medical records, payroll, and (outside Europe) billing.

Why does none of this exist?

There are many reasons this sort of software does not exist. It is very expensive to build using current software development methods, the health service probably does not have a great deal of competence in this kind of software (or the ambition or vision to develop it).

IT projects have failed in the past, and are very expensive. Budgets are tight.

And also, in the not too distant past, none of this was technically possible - software could do little more than be a tool for entering and retrieving information from databases - and most software was designed around trying to make value from doing this.

And management have been mainly thinking about getting the work of the hospital done for as little resources as possible, not about how technology could do more to support the staff. In the mind of a manager, staff fill slots in schedules, mainly.

But as the examples aimed to illustrate, situation awareness is the main factor to how well the job gets done. If a job is done well, the work is more efficient.

Our idea

So the main idea behind this book is that all of us work with situation awareness, and there are better ways that modern digital technology can help us with this. Conversely most of the digital tools we have are designed to help us complete tasks, or help other people check that the tasks are being completed.

If digital tools were geared more to situation awareness, it would help us to do much more, leading to better results for our organisations, a continuous learning environment, and happier employees.

And the capability of the software industry is changing fast, so it can do far more to support situation awareness. There are better tools to model ways people work, low coding development methods, much richer coverage of the domain variables by the software, more use cases. In short, more brains in the software.

Meanwhile, software companies are improving their business software by adding more ‘analytics’ and ‘intelligence’ to their software, but often this ends up solving a big company’s big data problem without actually improving Emily and Mark’s situation awareness at all.

Putting this together, we see room for something of a change of direction from the software industry and everything around it – the people that develop software, the people who commission or buy software, or make decisions about it – and Emily and Mark should get more involved with it, too.

Why we are discussing this now

There is nothing remarkable in the idea that any job can be looked at as either a “task” or a “situation awareness challenge”. Ancient seafaring, agriculture, hunting and fighting wars all involved awareness of a situation in order to achieve a task.

But the reason the subject seems worth discussing now is that we have the technical capability to build software that can calculate, and remember and coordinate things which only people could do before – but this software does not seem to be finding its way into the working world.

Nearly every time, we (as people) have a better understanding of the situations than the software tools we work with – something which may have crossed your mind every time a digital tool makes an unhelpful suggestion.

Some more examples of situation awareness and tasks

Here are some more examples of jobs people do and the situation they need – based on discussions with neighbours about the jobs that they do and the challenges involved.

Kate is a London bus driver, taking a large bus through thick traffic hour after hour. Keeping the bus moving means not only knowing what is around her, but also knowing, or making a reasonable guess, what the vehicles, people and bicycles around her are going to do. She gets better at this with practise from seeing what they usually do. She is building up a rich case history of problems and solutions.

Angela installs and fix office telecom systems. With many years of experience, she can get a quick understanding from a talk with colleagues or customers about how the system in an office she has never seen before is set up. She can also figure out problems quickly from matching her understanding of how systems are usually set up, with her understanding of common problems, with what someone is telling her. She has a rich case history of problems and solutions.

Louise is a salesperson for a home improvements services company, installing new kitchens. She can get a quick understanding of her customer – what their budget really is, what they are really looking for, and whether she can provide it. She also understands different levels of workmanship for people who actually build the kitchens. If there is a disagreement about the work she can quickly understand if the customer is making a reasonable compliant and resolve it accordingly.

Andrew is setting up a peanut importing business, bringing peanuts from Turkey to the UK. He's previously worked as truck driver and got contacts and understands the market. He knows the supplier, he knows some potential buyers, he knows what it costs to deliver the peanuts and what might go wrong, and he is gaining and understanding of the food business. He is building an understanding of the goal of the actors and the obstacles in his new business.

John is thinking of setting up a flight club in the Caribbean islands, which would own a number of small aeroplanes, which would then be made available to the members, who are all qualified pilots. John understands the cost of operating aeroplanes, he has a rough idea of how many people with private pilot licenses who might be interested in going to the Caribbean, buying a membership of the club and

gaining access to the planes. He understands what the competition might be, and what the obstacles might be to getting it going.

All of these people are working with what we call “information” – the bus driver's view out of the window, the detailed assets and obstacles of the office telecom networks, the lists of customers, the market information.

They are bringing their own goal model of the situation to the “information” in order to achieve a goal – a bus moves through thick London traffic with hundreds of actors making decisions that the bus driver has to anticipate to avoid serious risk, the office telecom system has hundreds of use cases that need to be matched to the customer, someone has a new kitchen at a quality and price they are happy with, the new business gets established.

They don't consciously regard it as “information” – their conscious minds are focussed on their view of the world, which is being continuously improved and updated.

And yet their software – if they have any – does nothing to help them. The bus driver's company has software to generate the drivers' rota and see where the buses are. The telecom engineer has a software telling him his 'jobs' for the day. The kitchen sales person has spreadsheets of different customer leads to call, software to make contracts, something showing how close she is to the target. The peanut business man has software to manage orders deliveries and hold financial data.

It would be great if the bus driver had software warning her about a cyclist in a dangerous position and in her blind spot; the kitchen sales person had a list of leads which are potential buyers based on behaviour hints that match past experience of serious interest. The telecom engineer has software telling him about a tricky job in his schedule today, but here's how the last person who had the same problem solved it. The peanut importer is warned about a competitor offering a higher price to his supplier based on recent information, not just a ferry strike which may delay a shipment. The flight club manager can monitor the utilisation levels of his planes, and any members thinking of leaving, or who might be damaging the planes through their flying styles.

Bonus benefits

If people can do their jobs better the multiplier benefits are huge, because jobs are more satisfying, people can learn more, organisations are better run and achieve more.

Situation awareness is how we learn, in its purest sense – we have expectations based on our understanding of the world, we see what happens and how it changes the situation, and we learn from this and adjust our expectations. This improves our chances to attain our goals. And learning at work is one of the strongest motivating forces for Emily and Mark – as well as being good for their employers.

Working with situation awareness is more human – we can use our personal judgements – and so we feel much happier and have a sense of ownership over our situation – all leading to us to bring more effort and focus to our work.

A working environment geared around situation awareness is a far more pleasant, adult working culture, where the people and systems around you are more geared to checking you have everything you need, not what you have done.

And with organisations doing better work, so many broader problems can be solved or reduced – child safety online, unemployment, safety (including road accidents), financial problems, security (including cybersecurity and terrorism), settling refugees into new homes, climate change, grumpy people voting for populist politicians, food and water shortages.

We hope this book can start something

The change which this book seeks to achieve – more software for work geared around situation awareness – can only happen if many people see the strength in the idea.

We hope that this book can help encourage more development of software which supports situation awareness. We hope that people like Emily and Mark, who work with software, can have a better understanding of what is possible, so they can ask for it in their company discussions. We hope that software companies will see this as a

direction to think more about when they plan their products, and that software buyers in organisations will recognise it is a worthwhile direction.

This book is written in very general terms with a very broad scope, but the ideas would have more value if the discussions focussed on specific domains, such as cybersecurity, internet of things, financial technology, upstream oil and gas. Or about software for specific organisations.

We hope that you will encourage your friends and colleagues to read this book and perhaps discuss how the ideas can be further applied in your organisation.

We also plan to continue with the series of conferences we have been organising exploring these ideas further (see www.softwarefordomainexperts.com) – you may like to come along, or register for our newsletter, or read our conference reports.

If you work in software

For our readers who work in software companies, we hope that this book can give you ideas about new products for your customers – and ways to keep your final customers happier, more human at work, continually learning, and more engaged with their software. The software industry for organisations could be double the size, leading to many more business opportunities, if it could do much more for people. But to get there, software needs to be far better at monitoring what is going on, and managing process and scenario variations.

Many companies are talking about ‘business transformation through digitalisation’. This book also shows how businesses can be transformed through digitalisation – although in a rather different way to the model which most people are suggesting.

Developing this kind of software is not easy, but not impossible either. How can you make software do more with less development effort? It could end up being developed and implemented in multiple layers – with a top layer controlled by Emily and Mark, a second layer provided by a company specialised in their domain, a third layer provided by a platform company (perhaps providing ‘low code’), a fourth layer provided by a larger software company, all running on a cloud system.

YOUR SITUATION AWARENESS

Please note – as of 2017, and with a couple of exceptions, Silicon Valley and mainstream software companies are not developing software tools which support people's situation awareness. This is because they see situational awareness as being industry specific and they prefer to deal with many domains together. They are looking for large markets which will all buy or use the same software products (Facebook, Microsoft) or trying to develop business models that replace established processes with ones that involve software (driverless cars). Or finding underutilised value (Uber, Airbnb).

Silicon Valley is not looking to support domain experts because the business model they anticipate does not play to their advantages. So the business opportunity is wide open.

CAPTAIN TASKMASTER – AND WHY IT MAY NEVER WORK.

Preventing us, and the people around us, from focussing on situation awareness is a belief which exists inside all of us, which tells us that we need to hurry up and get stuff done.

So that we can get better at tackling this belief, we're going to personify it, and give it a name – "Captain Taskmaster".

And some people have far more "Captain Taskmaster" in them than others. Perhaps your manager, or other people in your working world, spend so much time playing Captain Taskmaster it would be fit to call them Captain Taskmaster.

If we're at work, Captain Taskmaster can see that the more we get done, the more money the company makes – or the bigger contribution we're making. So Captain Taskmaster makes no excuses for pushing us to do more tasks.

If Captain Taskmaster is buying software for us to use at work, his priority is making sure we've done stuff, or we're doing stuff. He's happy to buy software to control and track purchases and transactions, track tasks completed, sort out work schedules, optimise the use of something, manage spare parts, manage sales leads, track sales calls, log completed jobs, line up the tasks which require your response, calculate your 'key performance indicators', show you what tasks you have

YOUR SITUATION AWARENESS

outstanding with some traffic lights. And that could be a pretty good description of most of the enterprise software which people are using.

If Captain Taskmaster is your manager, and he's low on empathy, perhaps he can't even see the world in any other way. Are the staff at work? What have they done today? Can I identify the people who haven't done much and remove them?

You can also see a bit of Captain Taskmaster in Donald Trump, the president who "gets stuff done" and fires people who don't. And you can see some Captain Taskmaster in the UK Brexit vote, the people who see simple solutions to the country's problems if only they were allowed to get rid of the pesky EU which gets in the way.

Captain Taskmaster is fond of providing "solutions" to any problem, and Captain Taskmaster often believes that some of the solutions are much simpler than they actually are.

MUSICIANS – AND WORKING WITH SITUATION AWARENESS

What does work look like if you don't have Captain Taskmaster shouting in your ear or building you tools?

Perhaps the ultimate situation awareness workers are musicians. Music in its purest sense contains no task at all – it is all about achieving a shared sense of awareness with the other musicians and the audience, leading to a sense of connectedness which everybody loves. And that's more of a destination than a task.

Musicians might have tasks to complete, such as performances, recording and teaching, but any artist musician will treat this as secondary (perhaps a close second) to the art itself, and art needs awareness.

It doesn't matter if a musician makes one song or a hundred songs, if the one song achieves the sense of awareness and connectedness, it is worth more than all the 100 songs.

Bringing our real self to work

YOUR SITUATION AWARENESS

When we work with situation awareness, we also bring our whole self to work. We are fully there, engaging our sense of conscientiousness, which is essential to feeling human and something Captain Taskmaster can't understand.

We are not trying to squeeze ourselves into a corporate structure, wondering why it doesn't work and letting out our frustrations by dressing up as Japanese cartoon characters on weekends. We are happy at work, engaged in what we do, and feel valued.

Understanding the patterns and learning

The way we learn – some people would say the only way we learn – is by understanding a situation, doing something, and seeing how the situation changes in taking us towards our goal, and then getting better at understanding the situation. Understanding is not the same as memorising.

This is the way that people have learned to do just about everything. Certainly the only way we learned before formal classroom teaching. And you could argue that we learn this way as well with formal classroom teaching, in that we don't memorise what the teacher has told us, but use what we hear to build our own mental picture which enables us to have a better awareness of what is happening in the subject.

And when we have mentors or advisors at work, what these people are usually doing is looking at our world with their ability and understanding. With their situation awareness skills and experience, they can see things we can't see. So they can say, "you might want to watch out for .. " or "this is what I think is happening".

If we are learning at work, we are motivated and feel that we are making progress. Perhaps learning at work is a bigger motivator than what we earn or our status.

From seeing how the situation changes as a result of the decisions we make, and from also understanding how the situation changes over time, we learn how the situation works. We know the patterns – what tends to influence what – and the stories – the sequence of events which make the situation like it is. Over time we can go into the situation and understand it in depth very quickly. We become expert.

We can judge if an unusual situation is harmless, or is in a danger zone and demands rectifying action. We have a good sense of what rectifying action would be the best.

An expert can have various modes of operation – whether making contingency plans in case something goes wrong, saying that everything is working but needs some input to keep it working, saying this time we are going to do it this way, or saying now we are going to do this.

These are on a scale of most comfortable to least comfortable.

The key component for this to work is situation awareness – which means that the expert understands what is happening – which means he is given the right information at the right time. He understands how it was generated, and he can trust what he can see.

Human beings have an amazing capacity to develop in-depth understanding of just about any subject, which can be used to make better decisions in order to achieve these goals.

People with understanding of a domain are able to break down difficult questions into simpler ones, and are able to quickly assess whether the current situation is within acceptable parameters, and if it isn't, they can figure out what might be possible to do, to bring within acceptable parameters.

How software supports and hinders our situation awareness

Building on this idea, we can see that software might support our situation awareness by giving us a useful picture about what is going on, which can be data, insights, alerts, dashboards.

Software should also support our sense of 'flow' – so we can get more absorbed in what we are doing. It can do this by making sure we can work out our speed – and the

YOUR SITUATION AWARENESS

software is not doing anything which dictates the speed, or interrupts us as we work out how to do something.

It is important that we understand what the software is telling us, and feel that we can trust it. Otherwise we're not learning anything. The more difficult the task is, the more important it is that the software can explain how it comes up with its diagnosis of the situation.

The software needs to know what we are looking for in order to be helpful. If we are searching for a plane ticket online, are we actually looking for the cheapest way to reach a destination on a certain date – or the cheapest way to get to a certain destination on any date – or something in between?

Also, the more complex our software gets, the more likely it is that we just see enormous amounts of data, which doesn't feel real to us at all.

Perhaps above all, the software should not push us into 'reactive' mode. People can become reactive very easily, perhaps as an evolutionary development so we would immediately allocate all our mental resources to an immediate threat and wait till we understood it before taking action. But we are not usually doing much useful work in reactive mode.

We can move to a reactive mindset if we can't figure out how to do what we want to do with the software, and focus on that rather than the goals of our jobs.

Software companies have learned how to push us into reactive mode, giving us an endless stream of alerts and notifications. It is fair game in the consumer sector, where we get given lots of free content in exchange for letting someone try to sell us something. But it is not fair game in the working environment, where we have a job to do and awareness to maintain separate to the software.

Keeping the organization flexible

An organisation organized purely around situation awareness is almost completely flexible – just like a human tribe 100,000 years ago. We do what we can, and if we can't do it, we try something else or go somewhere else.

It is only when we introduce tasks that the inflexibility arrives. We define who does what, how they do it, and what they do when they've done it – and keep people focused on getting the tasks done rather than anything else.

. It is only when the tasks come in that it gets rigid.

SITUATION AWARENESS IN DIFFERENT ROLES

In this chapter, we'll look in a little more depth at how people in different roles use situation awareness, and how it might be enhanced. The different awareness people need in real life jobs is very different in different jobs in the same sector, for different people doing the same job, or the same person doing the same job on different days, so this chapter might not have much value beyond illustration purposes, and will probably not have much value to a specialist in any of the fields mentioned.

The shipping industry

We'll begin with the shipping industry, which the authors of this book know well. Just about everybody who works in the deep-sea shipping industry is – at the heart of their role – monitoring situations and solving problems as they arise.

To put that in reverse, if there were no problems, there would be very little for anyone to do. The ships would cross the oceans on autopilot, the crews would be supplied by an agency, the loading and unloading would all be straightforward, computer systems would make sure the ships had all the supplies they need automatically delivered to the next port, and accidents would never happen.

In real life, you'll have technical problems with the ship, concerns and inspections from customers (cargo owners), a need to continually look for ways to improve safety,

ongoing training, a need to minimise use of fuel and other costs, a need to make sure the ship has all the supplies it needs and not more. Perhaps sick crew, or port inspectors asking for bribes, and much more. The company finances need to be managed. There will be extensive administration work.

The shipping company has evolved to put staff in various roles based more on what they should be aware of, and fix should something go wrong.

Onboard the ship, you have a chief engineer responsible for the ships engines and other equipment, officers responsible for navigation, and a captain with overall responsibility. In the office you'll have staff with responsibility for safety, overall technical issues, managing the fleet, purchasing, crewing, and accounting.

There will be many staff handling commercial issues. There are staff buying and selling ships, and dealing with investors, banks, and brokers. There are staff working with customers (the cargo owners), working with brokers to set shipping rates and make sure ships are not sitting idle, making sure the customers are happy with the service, and working out who pays what.

The software which has developed so far is around supporting tasks – navigate the ship, make purchases, read safety procedures, do maintenance of different items according to a schedule, arrange contracts associated with a charter (contract with a customer), work out who pays what, keep accounting records.

But here are some questions situation awareness digital technology might help to answer

What safety issue happened last time we were following this procedure, or in this port? Which of the spare parts in this purchase order really are critical? Where are the gaps in training for key crew members? What happened the last time this inspector checked our ship? How could the engine be operated more efficiently to save fuel? Are we likely to be able to pay crew salaries this month bearing in mind how long regular customers usually take to pay? How should a captain better allocate his time during a busy period in a port? What is the best country to keep our company funds, bearing mind where will need to transfer it to next and minimising exchange costs?

The upstream oil and gas industry

In the upstream oil and gas industry, like many industries, you win based on the strength of your position. What oil fields or 'acreage' you control, the quality of your engineering competence, the quality of your assets. Leading companies are highly skilled in building and continually improving their overall portfolio, including discarding assets which are no longer the best ones - passing them onto other companies.

This requires highly tuned situation awareness in many domains. Getting government agreement to access assets requires political judgement and communication skills, and financial skills to put together bids, where they are required. Financial planning skills are required to make sensible assessments of what might be possible for revenue from a project, also bearing in mind huge uncertainties in the price of oil and the amount which may be produced. And operating assets requires supreme situation awareness about safety, geology, integrity of equipment.

There are some 'tasks' here - drilling wells, processing batches of subsurface data, doing maintenance work, recruiting and training people, building infrastructure - but the quality of doing these is much less important than the overall situation awareness - and indeed all of these tasks are often outsourced.

Good situation awareness software might help answer questions like, What data do we have in the company which might help understand this new potential oilfield? What happened last time we were trying to drill through this sort of rock? Is our offshore production facility operating as it should be or are there any problems emerging? What corrosion problems should I be aware of?

Where is our company taking high risks? Were there any risks last time we did this particular procedure?

Engineering

Engineers might be proud to consider their work 'task based', getting the job done as effectively as possible. But to be a good engineer requires enormous situation awareness about the problem and the means the problem might be solved.

A good engineer has a vision for what might be possible – in the way that Brunel could see the potential of building a railway line from Bristol to London, where other people saw several days by horse and cart along a dirt track.

A modern day engineer could be given a challenge of designing a car that can be built for £1000 or which can run on extremely low fuel, and have to work out multiple optimisations to get something with the required strength, engine with the required speed, achieve required safety and keep the cost down. She'll also have to look into manufacturing methods, and different materials.

Engineers are often doing diagnostics on systems already running, trying to understand what is actually happening and what might be going wrong.

Software can support situation awareness perhaps by comparing sensor data to what the sensor data looks like when everything is running normally, including the relationships between different sensor readings.

The software might be able to bring up written reports or even videos showing what happened last time there was a fault like this one and how it was solved.

Maintenance management

Many jobs involve maintenance of something or another, whether it is teaching (maintaining the classroom atmosphere), police (maintaining a sense of security), or some kind of machinery.

With experience, we can master how to maintain it. An experienced teacher knows whether to calm a noisy class or let the noise continue, an experienced police officer adjusts the level of severity she talks to a possible troublemaker, a car vehicle technician can provide useful advice on what we should do with our car.

Maintenance is a big area for software – although sadly most maintenance software to date reduces the task to a regular schedule – and reduces the maintenance work to a list of tasks to do today, like change a bicycle chain every 18 months and the back cogs every 3 years.

But maintenance management systems for complex assets like aeroplanes, oil rigs and ships are gradually becoming more sophisticated. Data tools are available which can give technicians much better insights about what is happening and why it might be going wrong – not just comparing sensor readings with yesterday’s, but comparing the way sensor readings interact with each other with how they usually interact.

Military

In the military and air force, our understanding of the situation can lead to life or death. And there are plenty of steps in between – decisions which are made based on an assessment of the situation which can have a life changing impact on ourselves and the people around us. And modern battles don’t just have winners and losers, they have many dimensions.

Software might help a soldier by providing information like, “this road you were about to go down was mined last week”, or “we think there is someone threatening behind that doorway”. And it could also search through all the information collected across the organisation to provide a small piece of useful information at the right time.

In the modern military soldiers have been asked to set up schools, train people and have meetings with local chiefs. The software could usefully provide information about other soldiers’ experiences doing the same thing.

Police officer

The better the police officer, the better their situation awareness. You and I can walk down a city street and see a few people and cars – an experienced police officer can spot drug dealing, an opportunity for a thief, or a thief looking for an opportunity.

But police software, as far as we understand, does barely anything to support this. Police officers usually have databases they can enter and

retrieve information from – for example for crimes, vehicles or missing people. They may have systems which hold all the information about complex cases. They access software to retrieve their work schedules.

What kind of software could work better? Perhaps better integration between the various database would help – so that information only needs to be entered once, and the computer can automatically say if there is something in a different database relevant to the current work, rather than rely on someone looking it up.

There could also be software tools to help prioritise tasks, and help you keep track of the location of colleagues and what they are doing.

In North America, many police forces use predictive software, which aims to work out where crimes are likely to happen before they do, so human resources can be allocated. But note, here the software is not supporting the police officer's situation awareness, the software is aiming to maintain an awareness of its own – and generating a list of tasks for the human to complete, such as “go to this part of town at this time”. This seems likely to make the human miserable (who wants to switch their situation awareness off at work?) and will only work in reducing crimes if the computer's ability to maintain situation awareness is better than the human's.

Cashflow planning

Predicting and planning for cashflow is a major challenge for small businesses and some large ones. Accounts software can predict cashflow on the basis that all bills are paid on time both incoming and outgoing, but as we know, customers don't pay on time which means that suppliers can't be paid on time.

A more useful system could use recent payment data from customers to make a better prediction of what cashflows would really be – and then enable you to work out how much you might be able to push suppliers, to see what cash flow comfort level really is.

Company purchasing manager

In a typical purchasing system, employees are given rigid catalogues of items they are allowed to buy, and then enter their purchases and the price, and then send it for various 'approvals', which means a task for someone else (to press 'approve' on a screen).

A better system might provide awareness about how much the company actually needs this item, what the costs are of not having it, how useful it has been in the past. For complex items, it could gather feedback reports several months after the first installation. These could be displayed next time someone is planning to purchase this item. If the item proved not fit for purpose, and the problem was not rectified by the supplier, this information can be kept for a long period.

Many of us have had bad experiences with company purchasing departments – either not getting approval to buy something of obvious value, or the wrong items getting agreed, or the process of getting approval so complex people don't buy something, or the company buying the wrong thing because it was cheaper, then doing it again.

Loan officer

A loan officer or underwriter in a bank may have a list of applications for mortgages she has to give final approval or decline to. Useful situation awareness software might present a range of relevant information – if the bank account is in overdraft, other loans, income, financial records, mitigating factors to something which looks bad. It could also enable her to learn by giving advice about past performance. How many loans were approved which turned out to be bad debts, incurring expensive costs? How much money did the company lose from turning away loans where the customer went elsewhere and turned out to pay them back on time?

Lawyer

Legal software is available for 'case management' storing all the documents related to a certain case. IBM's Watson is promising to automatically 'read' thousands of documents, for example to search someone's e-mail archive for something related to the case.

What more could a computer do to help a lawyer maintain situation awareness? Perhaps a computer could show a lawyer about similar cases happening now in the country, what arguments are being made and what the resolution was. A computer could help a lawyer manage work among a number of different clients, ensuring that adequate attention is being given to all of them. Or help a lawyer make decisions about what task is best to spend time on now, thus improving productivity.

Recruiter

The recruiter has a job to fill and a number of applicants. Many software companies are attempting to make task based tools which do the job, distilling the job description into a number of boxes and then automatically parsing CVs to see how many of the boxes the candidate ticks.

Alternatively – perhaps the candidates should provide a short video about themselves. We can get a much better sense of the quality of a candidate from watching them speak and see their passion for the subject, than seeing what they have written – because we understand in much better resolution from what we see than what we read.

DIFFERENT SORTS OF SITUATION AWARENESS

This section is an attempt to create a mini classification system of different types of situation awareness involved in different jobs – both paid and unpaid. It may be valuable in indicating how many different types of situation awareness there are (and there is more than on this list).

The categories are “situation awareness many jobs need”, “situation awareness certain jobs need” and “situation awareness not usually connected with paid work”.

Situation awareness many jobs need

These are categorised into situation awareness around things (physical objects), people, markets, organisations, driving, financials, security and healthcare.

“Thing” awareness – needed if you work in maintenance, engineering, product development, equipment, design, construction - including cost management, fit for purpose, fault finding, reliable operations. It can include managing things. Managing inventories. Moving things. There are many domains of things - cars, ships, factories, buildings, construction.

“People” awareness – needed in customer service, hotel + restaurant work

“Market” awareness – needed in real estate, business development, marketing, purchasing, sales. What is happening in the market, what are people paying, what new products may sell. Who may be interested in buying.

“Organisational” awareness – needed in management, admin. Public services and government. How does an organisation's behaviour need to change? How can it be changed – what are the levers? Is it working? What needs to be done now? Are the right people in the right places? Managing organisational change.

“Driving” awareness – needed if you operate planes, ships, big road vehicles, cars. Includes both understanding external environment and how to operate the machine.

“Financial” awareness – needed in financial management, accounting, people who need to know the financial health of this business or organisation. Are the funds in the right places to pay bills. Also – running financial services such as banks.

“Security” awareness – needed if you work in physical security, policing, cybersecurity. Understanding biggest threats and best ways to mitigate them – and organisational and personal methods to do that.

“Healthcare” awareness – needed if you work in nursing, doctors, dentist. What is wrong with this body? What does it need? Managing associated systems / services.

Situation awareness certain jobs need

Just about every job is specialist in some way, and the person doing the job needs situation awareness of that specialist domain. Here are some examples.

Information technology awareness – keeping IT systems running, building software, fitting in analytics systems, devices, into an IT network. System interoperability. Technology developments. ALSO Hacking into IT systems and not being caught.

YOUR SITUATION AWARENESS

Data analytics - ability to get an awareness of what is happening from complex data.

Educational situation awareness - teaching, training. Understanding what is happening in a class.

Risk management - insurance, business, government. What should you be aware of? What is the risk level of this client / potential client?

Legal awareness - lawyers, managers. Understanding the laws which are relevant to the current case, what legal arguments can be strong which are relevant.

World affairs awareness - journalism, politics, some business

Public infrastructure awareness - electricity, road networks, rail networks, shipping systems, air traffic control. Is it working smoothly?

Environmental awareness - understanding environmental systems, emissions. What is the state of this region environmentally? How can it be fixed?

Scientific awareness - research, development, technology. How can the limits of science be pushed? How can new science developments be applied to a different field? What are the latest developments?

(usually) Unpaid work situation awareness:

Here are some areas of personal situation awareness which many of us develop, although not necessarily as part of paid employment:

Understanding animals - what they want, what are they doing

Understanding your own family - current health, what makes your children happy, what factors often lead to problems? This is particularly demanding when you have children with special needs

Personal health and relationships - am I looking after yourself properly. What factors often lead to negative outcomes? (this is known as 'cognitive behavioural therapy')

Social health - do I have a good collection of personal relationships? How does my behaviour, dress, and other matters affect it?

Personal situation awareness - how is my presence and behaviour affecting others?

Criminal behaviour - what can you get away with?

Artistic - what can you create?

Survival situation awareness - what are the biggest threats here? How do I make sure I make it?

HOW TO BUILD AND SELL SITUATION AWARENESS SOFTWARE

Building software is very expensive, and the money needs to be returned somewhere. And some of the software ideas we've outlined would take a lot of building, and don't have any obvious rich pocketed customers to pay.

But the question of how to get this kind of software built and sold makes more sense when seen in the context of today's software industry. We can look at what software is currently being paid for, where the advances in technology are, and where software could do more than it currently does for similar money, or where it could do more and generate more revenue by doing so.

Yes, we are looking at making incremental improvements to the status quo, not "disrupting" it. Most of the industries and fields we are discussing do not change quickly.

Can we make software less expensive to build – perhaps by getting more of the work right first time so less time is spent testing? (Software companies can spend half their time testing). Can we make the software more transparent and robust at its core, so it is easier to understand what it does and there are less crashes, and it is easier to integrate with other software? Can we make better use of automated software development tools?

Also bear in mind good situation awareness software does not necessarily take a lot of building. A car indicator light is a very simple piece of software that tells someone something they need to know at the right time.

Situation awareness software might just be something which tells you something important when you need it – for example a police officer wants to know if he has any colleagues nearby.

Some of the software ideas in this book use computer analytics, but you don't necessarily need analytics – the software could support situation awareness with the right “if-then” logic. “If my holiday flat has a high humidity level for a prolonged period, let me know”.

Another software approach is to develop a “digital twin” – a digital version of what is happening in the real world – which can be used to get a better understanding of what is happening in the real world. For example, Singapore is building a “digital twin” of the entire country with software company Dassault Systèmes, which will include all the available information – buildings, roads, weather, infrastructure. It will enable planners to answer questions like how the humidity will change if another tower block is built in a certain spot.

Another example of sophisticated situation awareness software is US company Maana, which builds software tools which companies can install in-house to run on their existing data, connecting different data sources together. As a result they can answer complex questions like “did I ever have any safety incidents in the company doing a task like the one we are planning to do tomorrow”.

The limitation is usually not computational power or communications capability – it is more likely to be the cost of programming, the difficulty of updating / upgrading software and making new features, and challenges with interoperability with other systems.

Types of situation awareness software

YOUR SITUATION AWARENESS

It may be helpful to try to classify different ways that software can help someone maintain situation awareness. Here are 7 ways.

Understand what is currently happening in your environment. We can get real time information from e-mails, sensors, updates from other software systems.

Co-ordinating with others. You need to know what other people are doing, you need to do tasks which go together with tasks done by other people, you have a manager allocating tasks to multiple people, you are involved in a complex activity with multiple tasks, such as an emergency response. You need to see where other people are on a map and what they are doing.

Problem solving. A software system can help you diagnose a problem. This is often used by car engine and aviation technicians.

Information from the archives. Your company has archived information which might help you understand your current situation. Perhaps the company was previously in a situation similar to the one you are in now. For example, you are an oil drilling engineer and the company has previously drilled similar rock and you want to know what happened.

Personal productivity. You want to know what would be the most useful thing for you to do next, or the 10 most urgent tasks right now, and if you are keeping to your schedule.

Alerting. Has something happened which you should be aware of (for example, a warning on your phone telling you that someone logged into your Google account from a new computer)?

Analytics. The computer runs some analytics on the data to try to derive more insights from it.

Using analytics to improve situation awareness

Analytics is a tricky subject to discuss in the context of situation awareness.

Many companies have tried using analytics to help them make decisions, usually working with data scientists, individuals with expertise in querying large complex data sets and understanding the results.

We have run a number of conferences about using analytics in the oil and gas industry – and a common refrain is that you have to have a domain expert – someone who understands what the data is showing them – involved in the analysis, perhaps as part of a team together with a data scientist.

If you believe that analytics alone can provide your company with answers to difficult questions, then perhaps you do not need to read any of this book.

Most of the stories we have found illustrating real value from analytics are where you have a domain expert with a complex question to solve, and large amounts of data in the company which might help, but difficulty getting the answer out of the data.

For example a planning department of a multinational company might want to analyse data from different countries to work out where to best invest money next. A safety manager might want to analyse multiple safety reports to work out where money might best be spent on a safety awareness campaign. A rail operator might want to get a better idea of the best time to conduct maintenance or track repairs, taking into consideration the wear an out-of-condition track is making to rail trucks and the cost of doing the repair.

Oil and gas people can use analytics to help them make decisions such as which oilfields should they bid for (or aim to get rights from the government to drill in), where specifically should they drill, how best to use various assets, how best to develop oilfields with platforms and pipelines, how to maximise safety, and keep old equipment running reliably.

When software detracts from our situation awareness

We do our best work when we can get into a flow, entirely focussed on the task in hand and working at their own speed. Our software responds to us like a musical instrument.

If we have anything other than this – the software setting our pace and restricting what we can do – our productivity and work quality is much impaired. We spend our time and mental energy trying to make the software work (and accept the data which we need to enter).

It is similar to trying to do your day job when you are a little sick, or trying to drive a car while talking on a phone. Your focus is taken by something other than the task in hand – and so your ability to do that task is much impaired.

Some people at work have software which takes minutes to login (and then logs people out while they are working). Some people must continually log in to different databases which don't integrate with each other to do different tasks. Many people need to use multiple software applications for their work, with no-one to help them move data from one to the other.

Some of the worst software in 2017 is in railway ticket machines – you'll have experienced bad software if you needed to buy a ticket but the machine couldn't explain what you needed, or accept your coins or credit card.

The reasons for bad software are often apparent, although depressing. The company doesn't care. The current system works 'well enough' (although the decision maker cannot see the frustration it causes, or the time wasted). There is no money to pay for upgrades.

The software has been made by people used to making consumer software (with its endless distractions), although it will be used by people at work. The programmers have emphasized a nice "webby" interface but it makes it harder to work out how to do what you want.

Returns on software investment

Whoever is spending money on software will want to see a return, and it won't always be provable – for example if you have helping someone to be a little more productive, or to improve security, the business benefits might not ever show up in the company accounts.

But here are two examples from where better situation awareness software might show more immediate financial results.

If you run a *call centre* – can you provide tools which give your staff much better awareness of the problem the person phoning in has. Can you bring up immediate information about the caller without asking for passwords and so on?

Can a call be automatically diverted to someone who dealt with the same caller last time? If there are several calls, indicating a problem, does it make more sense to “escalate” it to someone with more experience? Can you create an online diagnostic system of the most common reasons to call, enabling your staff to get much faster to the reason for the problem?

Or instead, you could invest your software budget in tools which monitor the phone calls as ‘tasks’, monitor how long each call is taking, and how long the phone queues are.

If you run an *estate agent* – can you give staff tools to get better understanding of the environment, automatically analysing data about properties on the market and seeing how your agency compares to the others? Can you monitor the funnel of properties coming into your agency, because you can only achieve a higher sales volume if you have a steady stream of properties to sell?

CAPTAIN TASKMASTER HITS BACK

Unfortunately Captain Taskmaster is not impressed with any of these ideas for how to make better software. To him, it is all more expense, and it does nothing to help people get their tasks done.

And to him, all of these discussions are distracting people from their work.

For Emily and Mark, the employees, Captain Taskmaster is not your friend by the way - or the friend of his organisation come to that. Because the more influence he gets in the organisation, the more rigid it becomes.

If your company has too many procedures, and the procedures seem to inhibit using common sense and judgement, probably you have a Captain Taskmaster behind it somewhere. You don't need situation awareness when you have procedures, he thinks.

Captain Taskmaster has also heard that automation and analytics might be able to replace people at work. He loves this idea. He is aware of people's limitations (why don't you catch more criminals, file reports on time) and doesn't mind the idea of computers being better.

He wants to make the organisation more efficient. It will make detailed plans and stick to them. It will control purchasing, with fewer suppliers. It will make better use of its resources and make sure all work is aligned with the main objectives. And for

problems, he will come up with some 'solutions'. They won't work because he doesn't understand the problem, but he's at least doing his task and feels good about it.

In education, Captain Taskmaster loves testing. It shows he's getting something done. And when he builds software for schools, it is all geared around the test.

Warren Buffett likes Captain Taskmaster

Warren Buffett, one of the most successful investors in the world, and probably the most copied, famously said, "If you've got a good enough business.. your idiot nephew could run it."

That sounds like the ideal of a company where everyone knows what task they need to do and how to get on with it. There are very few complex decisions to be made. For a company to function like this, it probably needs to be very standardized, doing everything in the same way.

And here in 2017 we have a lot of companies – probably a majority of companies – which look like this. And some are the sort of companies Mr Buffett invests in. Airlines, car manufacturers, insurance, banks, petrol stations, large retailers, consumer telecoms.

Warren Buffett is perhaps the most admired investor of 2017. So investors copy him, and company managers develop the sort of companies Warren Buffett might like to invest in. All standardized so "your idiot nephew" could run them.

And at the same time, there are plenty of tasks which companies could be doing more of and aren't – because they involve staff building an understanding of a situation and making a series of complex judgements correctly. Running healthcare practices of all kinds, doing sophisticated design services, working with real estate, accounting (understanding a company's actual finances). These can be profitable businesses, but Warren Buffett and his friends are probably not so interested. And we are in times where money can be borrowed at such low rates.

But Warren Buffett and Co are not asking themselves where good businesses could exist and currently don't. They are asking where is the best place for them to allocate

YOUR SITUATION AWARENESS

their money - and looking for solid, task based companies. But that creates space for you to try to work out where good businesses might exist which rely on people's situation awareness and judgement.

HOW TO BEAT CAPTAIN TASKMASTER WITH BETTER DIGITECH

Captain Taskmaster is powerful, logical, and looking for businesses in a certain way. There's some Captain Taskmaster in every company, you can't avoid him completely. You need to find a better way to work with him. Can you find a way that the company can use digital technology which will help staff have a better understanding of what is going on – so they do their tasks better – and it doesn't cost too much?

In this chapter, we'll look at some of the advances in digital technology, and software development techniques, which can help build better software tools to help people maintain situation awareness at a manageable cost.

Modelling

In order to build situation awareness tools which help someone understand their world, the software needs to be built around their world.

We use the word 'model' here – to basically mean creating a map of how someone works, which can then be used as a basis for software.

The modelling can be done on multiple levels.

On a deep level, you can model how the company actually works. For example, a shipping company keeps the ships in good condition, which involves inspections and regular maintenance, and doing the maintenance needs spare parts and skilled personnel. Entire books have been written on this, under the term “domain driven design”.

On a more day to day level, you can model what individuals in the company need to know. Shipping with the ship maintenance example, the shipping company’s technical management might want to know whether the company is up to date with its overall maintenance plan, if there are any major outstanding work, any major concerns. They also need to schedule when major maintenance work needs to be done (taking the ship out of service) and how this might impact commercial aspects of the company.

The day to day software might also be able to understand if something is happening out of the ordinary, which someone should be alerted about as he starts his work in the morning.

This sort of modelling is similar to what people in the consumer media do. They try to get an in-depth understanding of the concerns of ordinary people, so they can tell them (or sell them) things that might fit with it.

Making software easier to integrate

Giving people good situation awareness through software often means bringing together data from many different software systems. It makes it a lot easier if the software is easier to integrate.

For example, to create the software for Emily, the nurse, which we envisaged at the beginning of this book, would require an integration of patient monitoring systems, work scheduling systems, call bell systems and more.

Many people's work involves endlessly logging onto different software systems to do a different task.

Perhaps Amazon's Jeff Bezos had the best idea about how to make software easier to integrate. He famously once demanded that all of the different IT systems making up Amazon had a standard web interface, so it was easy to connect it to others. No shortcuts and workarounds were allowed. He did it because he was fed up of paying programmers to do complex integrations every time they wanted to connect one database to another.

A by-product of this decree was that Amazon had a robust interface into its cloud hosting system, which meant that it would be possible to sell access to it to other companies. This made it possible to create Amazon Web Services.

Many IT systems are overly complex, and simplifying them means a lot of difficult upfront work. Perhaps it takes a manager with the credibility, IT competence, drive and resources of Jeff Bezos to get the interfaces simplified. Or could any IT project could do it?

A further benefit of software with simpler interfaces is that it is easier to change. A common problem of today's software set-ups is that companies have a number of software packages working together, and if they make a small change to one, the whole thing falls apart. So we end up with companies using very old computer systems (also making them vulnerable to hacking).

Good situation awareness software needs to be very easy to change – because it is very hard to make software which perfectly supports someone's situation awareness right first time.

Too often today, we only see software integrating in a 'parent + child' scenario, such as someone making applications for Android phones.

Let Emily and Mark build the software

YOUR SITUATION AWARENESS

The idea of letting the end customers (such as Emily, Mark, Camilla and Andrew, who we introduced at the beginning of this book) get involved in building software will make software engineers shudder.

But surely it makes logical sense, if it is at all possible. The people who work with software have the biggest understanding of what they need, and the biggest motivation to spend time building it.

It is also far more efficient if they can get the software they want just by doing it, rather than working through the usual chain which exists between end software customers and developers - you know, product managers, sales people, corporate IT managers and more. There are no complex financial negotiations involved, because the only cost is their own time. And it will keep them far more interested in the software and the software processes.

Low code

One way to make it easier for people like Emily and Mark build the software is to make more use of 'low code' - tools which promise to eliminate the hard work of the coding.

Tools to automate coding development have been around for decades, and yet there is still plenty of hand coding going on, so it is hard to get a feel for how advanced they have become.

There are companies which promise to generate code automatically from a model (see Outsystems and Mendix, two examples of many). Low code systems can create code automatically as a result of someone describing what the code should do.

So far, low code systems appear to be mainly used for making online form type apps, such as a booking system, or retrieving data such as payslips or requesting holidays. Submitting expenses. So very task based. But the concept could surely be extended to include situation awareness tools.

Low code software should be far more reliable than hand coded software, since it is created by a machine. It should also be easier to update. If you want to add a new

feature, you can describe it in the model, and the computer system will create the code automatically. If you want the system to integrate with another computer package, you explain it in the model and the computer will make the code.

The maker culture

The maker culture is a movement where people get together to talk about stuff they have built (often using electronics). There is an emphasis on keeping everything open, not holding secrets. There is lots of use of Raspberry PI and Arduino computers, and a lot of 3D printing.

If anything can be described as an innately human desire, it is the desire to create, to build something to solve a problem. The question is not “how do you inspire people to create”, it is more “how do you provide an outlet for the desire to create which already exists”.

There hasn't been a lot of 'maker culture' in the enterprise software market to date – because it doesn't fit well with the commercial culture, which is largely about a company getting an institutional customer to commit to paying license fees for decades. That doesn't work if the core software is open source (i.e. free).

A maker culture could work very well in the organizational software industry, because it is extremely hard to work out (in advance) what will work – and just about every customer is unique. In-depth discussions and exchanging ideas about what works is not just helpful, perhaps it is essential to achieving some of the goals outlined in this book.

The maker culture can share – firstly – ideas about what works and how to build it. But it could also share code and software itself, and collaborate to develop data exchange standards which would make it much easier for software tools to integrate.

This has been seen to a limited extent in the oil and gas industry, where there are tools where geologists and geophysicists, experts on rocks and the subsurface, build tools to do specific types of rock analysis, and share them (sometimes sell them) online.

Self aware computing

For computers to support someone's situation awareness, it would be helpful if the computers had an understanding of what they are actually doing. We are calling this 'self aware computing'.

There is a (largely academic) movement to develop self aware computing, which you can read about if you Google it. But this is using a very broad definition of 'self aware', when the computer is aware of itself as a thing. Examples are a data centre which can manage its energy consumption, or an autonomous car.

We would like to use a more human definition of self aware computing, where the computer is aware that it is a thing amongst other things (i.e. people). The Wikipedia definition is "the ability to recognize oneself as an individual separate from the environment and other individuals".

So if a self aware computer system gets an instruction from software installed under an unusual means to encrypt all of its files, perhaps it would be aware that it is ransomware and the customer might not be too happy.

A self-aware computer might also be aware, like a dog, of the desires of its customer or owner – and adjust accordingly.

The academic discussions about self aware computing talk about 'private self aware computing' (the computer acting as a self-contained unit) and 'public self aware computing' – where the computer interacts with others. Perhaps these ideas could be useful.

A self aware computing system supporting Emily, the nurse, might realise that there is something going on with patient in bed 15 she should know about, or that Emily is getting a bit fatigued and grumpy and it might be helpful to gently let her know.

It is very complex to add self-awareness to old computer software – it is probably something which needs to be built in from the start.

Using the ‘internet of things’

The ‘internet of things’ largely means network enabled sensors (and can also mean other devices like refrigerators, cars and mobile phones). In theory, the sensors can make a big contribution to enhancing situation awareness – you can see what is happening somewhere where you don’t have eyes and ears.

The risk is that you get too excited about the devices and collecting data, and end up with something which is confusing and does not aid your situation awareness at all.

This has been seen in the shipping industry, where companies have been fitting more and more networked sensors to engines for decades, but now discovering that to use the data to work out how to run the ship more efficiently is still exceptionally difficult.

Conversely the most useful information for someone trying to monitor a ship’s fuel efficiency could be simply to know exactly where it is at noon every day, and use that as a basis for analysis.

Making a digital twin

A digital twin means a model of an entire physical thing on a computer, updated as the physical thing is updated. It can be used to try out plans before doing them in real life, and get a better understanding of what is happening.

Some oil companies have built ‘digital twins’ of offshore oil and gas production platforms, a detailed 3D model you can view on screen. You can use it to understand how the offshore platform works far easier than if you were actually there – for example you could change the colour of pipelines carrying gas yellow, to separate them from other pipelines and understand the gas flows. You can also use it to plan out any maintenance work, and test if a new component you plan to install will fit properly.

Stop building software around the database

Much organizational software today is built around a central component which makes it very rigid in how it works with data – its relational database. You can see an example in any software which is ultimately about entering and retrieving information, such as accounts or scheduling software. A relational database is like a spreadsheet – a table of data.

With early software (and much software today), all of the logic of the software worked around the database. Think about library book management systems, standard accounting software, software recording bank transactions, planned maintenance management systems, purchase management systems.

As an example, see the evolution of ‘customer relationship management systems’. They were first introduced in the 1970s as customer database systems. Statistical analysis on data was started in the 1980s, perhaps identifying a class of customers which were responding to a certain sort of marketing. (The data stays in the relational database). From about 2000, the databases were available via cloud.

A modern customer relationship management system might be able to bring up information about one client onto a dashboard from different databases at once, for example one showing the sales contacts, one showing the purchases made, one showing the response to marketing e-mails.

As humans, we can enter all kinds of data, process it and retrieve it from our brains in many different ways, with barely any rigidity. If software is going to help us to this, it also needs to be much less rigid in how it manages data.

This is all possible to do – by taking the database away from the central logic of the software. The central logic of the software ought to match what the domain actually does (policing, nursing, helping people manage cashflows). This can be rigid, as it is in the real world. The databases can just be data stores which the central logic dips into as it needs to.

Industry experts working with programmers

Finally, one useful approach for making better software is to have more industry experts getting involved in the software development. This already happens, but it could happen more. People who have worked as nurses develop nursing software, and so on. We call this ‘domain expertise’.

The software industry works in a number of “platform” layers, with cloud hosting services, databases, software platforms, and configuration layers. But maybe the best place for the ‘domain experts’ to work is right at the top layer.

So if Emily and Mark have a problem with software, or a suggestion for improvement, they can phone someone who understands the software and previously worked as a nurse or purchasing manager – who can then see if it is possible to configure or adjust the software to do what they want, or convey their ideas further down the chain.

COMMON DIGITECH MYTHS WHICH WON'T HELP WITH SITUATION AWARENESS

There is a common narrative about what computers are doing in the working world which we commonly hear in conferences and books – basically about computers replacing people. We think much of this is wrong and also prevents computers from doing the most they could do to add value in the workplace.

Myth: computers can do people's jobs

Most jobs are done by people like Emily and Mark, who are conscientious, want to get the best understanding of the world they operate in, and make the best decisions and judgements. Because this is what most people are like.

Further, computers are nowhere near as good as people at understanding situations. With unlimited resources you could program a computer to do any small task. But it would get quite expensive to program a computer to understand the world

the way Emily and Mark can when most people cannot explain how they understand something. If the computer needs an alternative way of understanding a situation, it needs to be as good.

Computers are much better than humans at doing maths with a few variables, but computers do not handle many different variables very well, whereas people can – we can easily weigh up hundreds of different pieces of information into account when making a decision.

Most jobs are in fact bundles of jobs. The hotel cleaner has to clean multiple rooms to an acceptable standard on schedule and manage the playoffs involved (clean under the bed every time?). The dentist has to see multiple patients on schedule all with acceptable standard of care. The travel agent is creating a holiday within the budget not buying flights.

You might be able to program a computer to do one of these tasks – the ones which are more suited for computer, and done millions of times every day (like driving or searching for flights). Then there's tasks that only a handful of people can do – which would not be viable to program a computer to do. Then tasks which would never be viable. Then the meta-task of deciding which task to do next – which would be nearly impossible for a computer.

An Amazon courier or minicab driver has to find difficult addresses, keep to a schedule, find somewhere to park in off moments, put children into child seats, tell the passenger where he is, cope with dogs. A truck driver must make a complex decision about when the truck can park to the side of the road even if it causes inconvenience to other drivers for a while. All these need situation awareness beyond the ability of a computer. The computer might be able to drive down a straight road.

Myth: you can easily teach a computer a job

If the computer is being 'trained' by hand (such as being trained by a cancer doctor to understand cancer cells), there needs to be some cancer doctors with a lot of time on their hands (and the same for any other domain).

Consider autonomous driving, where billions have been spent making cars and trucks which can drive themselves – getting a 3D understanding of the road around them, to work out if it is safe to move forward.

But consider what a human driver does in the city. Other road users need to be considered – and not just what they are doing right now but what they might do. See that child on the side of the road who looks like they might step out into traffic. Can you see from her facial features that she may have learning difficulties? What about the cyclist coming up outside you – what is he going to do next in all probability?

And the ‘job’ of a driver usually involves far more than just driving.

Myth – computers can learn by themselves

But not to worry, many people believe that computers can learn by themselves like a person can. They’ve given a technical name for computers which can learn general things – “Artificial General Intelligence.”

but there’s not much evidence of it yet – except in situations where the limits of the problem are tightly constrained, and the computer gets instant feedback on how well it is doing (such as a computer game where the score always goes up). Such as with a computer playing a computer game.

Aside from that, as of 2017 computers have only learned how to recognise cat pictures but not as well as a person can. We can’t predict future technology advances very easily, so it might be possible, but if we restrict the discussion to what is possible today or with immediately available technology, computers cannot learn except in very limited artificial scenarios.

Myth – Moore’s law will change

everything

Digitech people are usually excited by advances in digital technology and what is possible to do with it – usually around ever faster processors.

But there's no reason to believe faster processors don't necessary do much to help people's situation awareness – unless you believe that computers can learn by themselves with the help of fast processing (and computers can't learn by themselves, although many people believe they can).

But the human brain does not do 'processing power' or 'flops per second'. It does have billions of years of evolutionary development to have the ability to understand situations (so a bit of an advantage on microchips). Brain power vs chip power is not a useful comparison.

Myth – Computers can make decisions better than people (and that's helpful)

Digitech people are fond of showing that computers can make better decisions than people – about who to let out of jail, or where to invest.

Even if this is true, a flaw is that there are not actually many jobs where skill revolves around ability to make specific decisions. And where decisions are a major part of people's jobs (such as investors, purchasing managers, oil companies), a lot of the quality of the decision making process is in working with available data and building models from it, and ensuring that in what you think are the worst case scenarios, you won't be in too much trouble. So it comes back to situation awareness, that thing which (we have tried to explain) computers can't do.

Many of us in our working lives make thousands of decisions and build mental models about how to respond to requests we frequently get.

Think of a bus driver with a one hour bus route through London – might regularly get passengers who want to board and not pay, or getting stuck behind a road cleaner and having to decide whether to overtake, or arguments over pushchair space. The company does not necessarily set rules about what to do in these situations, but each driver will probably set her own mental rules and follow them. So these might look like decisions, but in the driver's mind, are more situation awareness and mental modelling.

Myth – the best companies show exponential growth

Many digitech people believe that good companies in 2017 are growing exponentially, like Uber and Airbnb have done.

The flaw in this logic is that these companies are technology companies – based around a piece of successful technology. In these cases it is successful because it worked out how to help people better utilise underused assets.

There will be more successful technology companies in future, and they can't be predicted. There are not many under-used assets as valuable as cars and houses, and so it seems unlikely that another business model will come along similar to Uber and Airbnb.

Non technology companies are centred around people's situation awareness – and there's nothing exponential about that. It takes a long time to develop step by step.

Myth – agile techniques can be applied to all businesses

Digitech people love 'agile' and 'scrum' – a method of keeping software development teams tightly focussed on working hard to achieve a specific objective,

breaking it down into short phases of work and frequently re-assessing how it is going. It keeps everybody on track and productive.

If you're making software for agile businesses (rather than developing software using agile processes), this is the last thing you want. It needs to be solid, transparent and well understood and carefully planned from the start.

It might be quicker to build software by locking people in a room until they make something workable, but if you end up spending the same amount of time testing, because no-one knows exactly what they created, it may be cheaper overall to be more diligent from the beginning.

Myth – users are resistant to change and that's bad

A common complaint heard by digitech people is that the people they are making the software for are 'resistant to change'.

Users are resistant to change because they are people like Emily and Mark who want to be the best they can be at understanding the world they work in. This takes decades. Perhaps Mark wants to be manager of an entire hospital one day – and have the ability to spot in seconds what needs to improve.

He doesn't want any new software package particularly and the headache of learning how to use it. He might have some idea for a new feature which would help him.

Myth – we should focus on sales volume

YOUR SITUATION AWARENESS

Trying to sell software products on volume makes commercial sense, but the likelihood of making something which exactly meets Emily and Mark's needs is pretty remote. Even if something works perfectly for a nurse or junior hospital manager in another hospital. Making good software needs a lot of sitting down with the customer.

A better approach could be a layered strategy, with big software companies at the base making platforms, and very small consultancy type businesses at the top working closely with individual clients helping them get their tools working.

Software companies understand that their customers expect to be 'wowed' by new software, and so often develop whizzy graphics, fancy jargon and more.

But good "situation awareness" software is not dazzling. The opposite, usually. A "user interface" which tells people exactly what they need to know when they need it – probably does not look very exciting, need not look very exciting, and any exciting graphical features could easily detract.

Good situation awareness software can look like the electronic boards on underground trains which tell you which trains are coming next, where they are going to and when they will arrive.

Wearable health devices help monitor the situation

In theory, wearable health devices should be about situation awareness – you understand your own health. But do people really use them that way? Do you need a Fitbit to tell you how well you slept or how much you walked?

In practise people use these devices far more as motivational tools – in other words to set themselves tasks and try to achieve them – 6 miles running day.

That's fine but if you use a device like that you might like to question whether you are letting a bit too much Captain Taskmaster into your life and whether it might be poisoning your thinking. Running and walking are great times to think and get a better understanding of your situation. But not if you have a Fitbit distracting you.

DON'T PUSH CAPTAIN TASKMASTER

All of this – we hope – is strong logic for a different way to build software, which should help you go to the Captain Taskmaster in your company and explain why it is worth the company investing in it.

The problem is – if you push Captain Taskmaster, he pushes back.

He's got a vision for how the company should run, where everybody has his tasks, and it is all straightforward, the company has no waste, he knows who's doing their job, and the Warren Buffett style investors will like it. He knows you don't like it, and he's watching you. He's not ready to be persuaded of a better way. And perhaps – frankly – he needs you out of the company.

Perhaps you have too much time on your hands thinking of this stuff. Do you need a bit of competition maybe, to get your tasks done better? Is there a private company we could outsource your work to?

You know, he likes the idea of threatening that machines can take over your job, whether or not they actually can.

He's your boss by the way, and your bosses' boss. He doesn't have much empathy and may be a narcissist. Not all taskmasters are narcissists. But narcissists can become taskmasters.

YOUR SITUATION AWARENESS

If he threatens you, that will absorb your surplus situation awareness with worry, so you won't do much more than focus on the tasks. With no situation awareness, you are like a robot yourself. Then it is much easier to let the robot take over.

He's already achieving this by the way. Today's generation of workers are probably less focussed on their situations than the generation before them – because today's generation are too busy trying to navigate the complex world of processes and software and rules placed in front of them. That's where their situation awareness goes.

He might drop the idea of a bonus in front of you, money you can have if you get your tasks done just as he wants it. That will take over your situation awareness too. It is a dangerous strategy, if you stop being aware of the risks you might be taking in your efforts to get a bonus, but if it goes wrong you get the blame not him.

BUT THAT DOESN'T MEAN TO STOP TRYING

But life always has been about a fight against the taskmasters – and still, people have done wonderful things. Over history we've had taskmasters making us build roads and join armies, and narcissists in power, and yet great works of art have been created.

Perhaps humanity itself is all a fight against the taskmaster – who also lives within us – and wants to be fed.

We beat the taskmaster by focussing on our situation awareness – and letting the tasks get done as a by-product of that.

And there is stuff which needs doing sometimes and we can thank Captain Taskmaster for that.