The Abstraction Coupling

A better way to connect domain and digital expertise - so we can make organizational and industrial digitalization projects go further

KARL JEFFERY DIMITRIS LYRAS

THE ABSTRACTION COUPLING

A better way to connect domain and digital expertise

So we can make organizational and industrial digitalization projects go further

Published by
Software for Domain Experts

39-41 North Road, London, N7 9DP, UK

www.softwarefordomainexperts.com

Contents

1 Why the Abstraction Coupling

1.1 Short, dense introduction

"You can't transform anything with just digital components. You can't transform anything with just domain competency. The magic happens at the interface. By bringing digital and domain competency together, this is where the magic happens." - Karl Johnny Hersvik, CEO of oil and gas operator Aker BP.

This book proposes a way to bring digital and domain competency together, via developing the space in between with abstractions, using something we call an "abstraction coupling."

By 'domain competency' we mean people with understanding of how to make various parts of our organisations work. Just about anyone with a specialist expertise, including management expertise, and ability to operate equipment or do specific tasks, in an organisation, has domain competency.

We will show how our organisations would be able to do much more, with digital technology design and development, with a better connection between digital and domain competency. And we'll show how this abstraction coupling might be made.

In doing so, it may be possible to solve a lot of organisational problems, help digital technology and the technology business move forward, and help our organisations become more effective.

It could also create many opportunities for tech companies providing products and services to organisations.

It is hard to connect digital and domain competency together, because both worlds are getting increasingly complex, both worlds are very different, the richness in how we want to use technology is always going up. It is hard for digital people to understand how domain experts work and what they need from technology, and hard for the digital products to be understood.

It doesn't serve anybody well when people say (or, more often, think but don't say) that something is too complex for them to understand so they should not try to understand it.

The pathway forward is that people do not need to understand the full detail of something, often just understanding an abstraction, or sort of reduced version, is enough. Like when we can get a long way to talk to someone who speaks another language if we know a few words. But someone needs to create this abstraction.

There's nothing new about using abstractions to help people understand things. Leaders, writers, consultants and analysts have been doing this ever since they started wanting to explain anything.

Bear in mind that domain experts are usually seeking to make specific decisions such as to change something or buy something. Abstraction itself is not the goal. But it is a pathway to being able to make these decisions.

What we offer which is new is the idea of focussing on abstraction as a business process in itself, and the first steps to defining what this might look like. And we're also developing a new way to define what an abstraction can look like. Not just about simplifying reality, but also bringing out the core ideas behind it, describing it using written text or diagrams, showing what elements connect to what.

This abstraction coupling can be thought of as an invented space between digital and domain competency, which connects them together.

This “abstraction space” can include standard formats, because if we know something fits to a standard format it also gets much easier to understand.

Standards can take a variety of forms, such as a standard collection of data we request from a supplier of certain items, or a standard group of behaviours which we see in our street following certain events, such as snowfall. We could also see these as ‘templates’ or regularly seen pattern. We can work with the template, standard or pattern, rather than the detail.

What is specifically new here is the proposal that the development of abstraction couplings should be a discipline - evolving into a skill people can develop and a product people can sell.

When people do abstraction couplings now, it is done in a piecemeal way, not as a structured process. We could consider it like musicians making music by figuring it out as they go along, in comparison to musicians creating saleable products and being paid as part of a process.

The abstraction coupling also involves an understanding how digital people and domain experts work differently and see digital technology differently. Digital technology is ultimately about getting a machine to do what you want. Organisational domain competency is ultimately about getting an organisation do what you want.

Making a machine do what you want involves getting into the details of the data points and

code, making sure the instructions you send the machine are correct and precise.

People have skills to remove ambiguity from something and make it clear. But machines can't do this unless you "teach them" and that involves a lot of hard work.

Getting an organisation to do what you want involves understanding what is going on, what is driving what is going on, what may happen if you do certain things, and how something might be done better, and explaining it to others, although finally ending up with specific decisions.

So, the work of domain experts can involve abstracting, looking for the driving forces behind what is being seen. Digital experts may be using abstractions to understand something in their minds, but this does not usually connect with specifics in their software development.

The business opportunity here is that by using abstractions in a structured way, organisational digital technology, and its developers, could go much further in being able to connect with their customers, the organisational domain experts, supporting better understanding on both sides.

This is about more than communication. There have been projects going on for decades to try to encourage digital people and domain experts to understand each other better and talk to each other more. While this is moving in the right direction, it does take up a lot of people's time.

Instead, we propose that someone – a company, or a team – could take the role of creating abstractions which sit between domain experts and digital experts. This book shares ideas about how this might work.

It will probably need to be driven by technology companies and digital experts more than domain experts though. So, for this to work, digital people would need to buy into the idea that this method can help them go further.

1.2 Where we are going with this – better digital technology for organisations

Developing an abstraction coupling can open a pathway to much better digital technology in organisations.

It can take us to the point where people in all roles can better understand their current situation, make schedules and plans, get an understanding of what may happen if different things are done based on experience, and implement their plans for maximum safety, efficiency, value, or whatever their goals are.

We can imagine digital technology which can provide the right piece of information to the right person at the right time to support whatever decision they are making, with a mix of analysed data and experience. We use the word 'imagine' because it has never yet been seen.

Their technologies would all work reliably and give people information wherever they want it.

The technology would function at multiple levels, intersecting with the multiple related goals which people in all organisations have.

This would lead to more effective organisations, making better outcomes for everybody and everything which interacts with it – whether employee, customer, investor, member of society or the environment.

1.3 Organisational digital technology right now

Organisational digital technology, as of 2022, is often built and marketed with similar methods to consumer digital technology but a few steps behind.

Consumer digital technology is about entertaining, connecting, giving people tools, supporting transactions. It is often marketed based on how much it looks like magic.

Implied in this is that the digital world is expecting its 'users', as they call their customers, to come to their place and the world they feel comfortable in. This works OK for computer games, standalone tools, communication systems, social media. But not for domain expertise.

The domain expert has their own domain they want to understand. Domain experts are dealing with real life things – people, assets, tasks.

Digital technology people are dealing with the code which runs computer processors, the data, which is generated by people, sensors and software, which goes through communications networks, gets processed and analysed, and visualized on a screen, and the data structures behind it. It involves trying to keep track of the endless detail and logic of systems, so problems and be resolved or systems further developed.

The forces are not pulling in the same direction, and often go against each other, which limits how well digital technology and organisations can work together.

1.4 Abstraction couplings in other areas

The idea of an abstraction coupling may be new in the digital technology / domain competency space, but it is a concept which is very mature in other sectors, something people do but don't need to think about.

This is the way that connections between all kinds of groups happen. Diplomats explain their own country to the people of the country they are posted to, and the opposite, by making abstractions of the specifics.

Senior managers use abstractions to explain business matters to their more junior employees. People identify commonly seen patterns in commonly seen events and create shortcuts to describe them so that others can grasp something complex quickly, like a 'supply chain snarl-up'.

Making abstractions involves something which non digital people might call 'common sense', or what could be more precisely described as putting new things into perspective, using good judgement to prioritise what is important and would benefit from being shared with someone else.

Common sense is not something which can yet be programmed into a computer, and there has not been a well understood need for it.

1.5 How domain experts work

Right now, the goal of many digital technology developers could be to make a good 'dashboard', an online place where people can see whatever they want at whatever level they want.

Their 'abstraction' of the needs of a domain expert, is to have information at their fingertips.

But domain experts do far more than just look at information and make decisions about it.

"Domain experts" is our term for people who work in organisations who understand their field and make judgement. These can include expert professionals such as doctors, managers, teachers and engineers, but it can also include anyone who has a role in an industrial organisational context which needs specialist skills, including people operating equipment, making schedules or any other decision.

Many tasks involve a mixture of monitoring a situation, understanding the situation, making decisions and plans, and communicating them.

The understanding of the situation involves mixing what they see is happening with what they already know from experience or other discussions. They might be assessing risks, identifying trends, finding similarities, considering options, making predictions and deciding on goals.

Then they put together multi-level goals to get to where they want to go, issue instructions, make schedules and plans. They explain that to other people, including separating out what someone needs to know based on their current perspective or role in achieving it.

They may need to make a schedule for people, assets, customers, tasks. Again, there will be a lot of data, but it doesn't all help tell them what the best schedule would be, or where to add buffers for flexibility, or how best to change it when it needs to be changed.

Domain experts are relentlessly prioritising, because life so easily gets full of things which are not that important. You have to find the important 20 per cent of tasks, or pieces of

information, which does 80 per cent of the work, or informing.

Domain experts are probably simultaneously understanding a situation, analysing and making decisions. These do not separate in the brain. They may be drawing similarities between what they see now, what has happened in the past, or stories they heard about, and what happened last time, what they did or could have done which would have improved it, and then deciding about what is right to do this time.

A doctor is not separately understanding the health of a patient and deciding what to do about it. A teacher is not separately understanding the competence of a class and what is best to teach next. An investor is not separately understanding the performance of investments and deciding what to buy or sell next.

In doing all of this, they are making abstractions themselves of what they think is going on. This is a skill which is innate to people. We have always needed to understand complex situations to survive.

Often, domain experts have been working in the same domain for decades, with other people who have also been doing the same thing, with a continuous body of knowledge going back to the birth of their industry or organizational field – such as in shipping, law, property, architecture, government, police.

The most important part of a domain expert's work is often situation awareness, understanding what is going on. They can't see all the people, all the activities, all the technologies. And working in the 2020s, they will probably have a lot of data available, which isn't necessarily all helpful. Technology abstractions can model this granular data into scenarios, it may be

this or that, which can help a domain expert decide.

Described in the context of using technology, domain experts may need to combine information from different sources to get a picture in their minds of what is happening, so they can make decisions.

They may want to ask their own questions and follow their lines of thought according to their own mental models, not the models the software is designed around.

They may want a better understanding about how the information in front of them came from, and reasons that the truth may be different to what they are being told.

They build up a mental understanding of cause and effect - what factors are likely to change the situation, and how this change works. This may be negative factors which make the situation bad, or positive factors which can make it better.

This can include an understanding of the dependencies, how one thing may affect another thing. A manager of a supermarket will understand that if they are to stock a new product, they will need to no longer sell something which was previously there, because there's limited shelf space. If someone is given a new role, they will need to find someone to fill their existing role.

If domain experts have digital technology, they want to understand it – understand what to buy, understand how to do what they need, understand how its logic works.

The ultimate purpose of domain expertise in an organisation is normally to make decisions and achieve goals.

1.6 How digital people understand domain experts in 2022

Contrast this with the way the digital technology world usually tries to understand the world of domain experts and their needs for software.

We have systems for 'requirements gathering' up front. Software 'requirements' are often defined starting with the user interface. Then the software development company develops the software according to the "requirements".

They want to constrain the 'scope', what something will do and what it will not do.

To illustrate the limitations of this, imagine if a building architect worked the same way.

They would start by asking a client what kind of building they want, although the client knows very little about buildings. The next time the client gets involved is when they get asked how they feel about their experience of the finished building.

Building architects work in a long consultative process with the client, where they discuss what might work well or not work well out of what is being suggested, and bring a range of plans, models and mock-ups so the client can get the best possible understanding of what is about to be built during the process.

Or imagine a musician who made music by first asking the audience what they want to hear. Not many people know what music they want to hear, until they hear it. Musicians make good music by developing skills to hear their own music the same way anyone else would hear it, and the ability to discern how good something is.

What is obstructing software developers from engaging more deeply with domain experts in the process of making software, to make something which is as useful as possible? Perhaps it feels like an unaffordable luxury.

And much of the way domain experts work cannot be easily described using the modelling languages used to make software, such as UML, if they are used.

1.7 Real examples to illustrate the mismatch- QR codes and webinar software

Going into more detail, here's some examples of times when the digital and domain expert worlds go in opposing directions.

First, the QR code. There is a big difference between how software people envisaged they might be used when they were first widely introduced in around 2010, and how they are used now, that businesses have had chance to work out where they really add value in the 2020s.

In around 2010, we used to see QR codes on advertisements everywhere. This was a technologists' perspective about how the technology should best be used. In their minds, someone would see an advertisement, have a desire to go to the company web page. They would photograph the QR code with the phone and be taken to the web page.

But from the perspective of someone reading the advertisement, or a 'user', it means another fiddly way to go to the advertisers' website, should they wish to do that using their mobile phones. They could just type the company name into a search engine, or type in the web

address. Perhaps they would rather browse the website on a PC with big screen, than a phone.

Fast forward 10 years to 2022, and we see QR codes being used in areas where they provide organisational value. They are used to connect a moving thing, such as a parcel, spare part, or person, which carries the QR code, or has the code fixed to them, with data about that thing in a computer system.

So, we see them used on Covid apps for scanning, boarding passes and tickets, parcels, spare parts.

To reach this understanding of where QR codes really add value would need knowledge of both technology and challenges of organisational worlds, which hardly anyone has. Instead, we got there more by trial and error.

Here's a second example of digital and domain expert worlds going in opposing directions – how they might imagine a registration system for a webinar should work.

A digital expert thinks: this challenge is about updating a database with the details of someone who has registered, and a flag saying they are registered. How should this be best achieved technically?

An organisational domain expert thinks: how do I find out this webinar is happening, if it is relevant to my work? How easily can I find out more about it, whether the speakers know what they are talking about, or have something new to say? How difficult do they make it for me to register? Do I need to make a commitment of any kind?

A company organising technical webinars as a commercial service might be thinking, if this webinar is supported by a commercial sponsor,

how can the sponsor get more value from their financial support?

1.8 Can we define abstraction

To go further with our idea of the 'abstraction coupling', you might want a concise definition of what an abstraction is. But that's quite hard to provide.

Abstraction is central to all our understanding of our lives, our relationships, it is what makes us human, it is what gives us our human power and ability.

A working definition could be something like, taking the driving ideas or concepts out of the detail that we see. Or working with accepted standards rather than having all the details. This could be the standard notation used on a geographical map, or a standard 'bundle' of data points attached to an object.

Calling an abstraction a simplification or a reduction isn't necessarily correct, because the abstraction can be just as complicated as the thing we are abstracting. For example, we can abstract the knowledge we have of a person from our interactions with them into an abstraction of what we think their motivations are.

An abstraction is always done in pursuit of a goal, because otherwise there's no frame of reference to do it. It is usually understanding better how something works or what drives it to be as it is.

The abstraction needs to be done with what we might call 'common sense' – human judgement. It cannot be done by machine unless explicitly programmed what to include and what to leave out, in which case a person has defined the abstraction processes in advance.

A conversation, or the telling of a story, can be an abstraction, in the sense that we are not giving the recipient the full detail of what happened, but telling them about the driving forces behind the event we are describing. This makes our story more interesting. And just like we could never define precisely how to have a conversation, we can never define precisely how to do an abstraction.

A conversation is also an abstraction coupling in that we are looking to connect something with the person we are talking to. So common themes for conversations are themes which are meaningful to many people, such as the weather, football, Christmas, shopping. We learn how to do this instinctively and were never taught to do it.

1.9 Some things we want digital people to believe

A big challenge with building interest in the ideas in this book is that they go against some strongly held, if unspoken beliefs, which people often working in technology sectors often have. We are not envisaging, for example, any 'singularity' where "technological growth becomes uncontrollable and irreversible" and robots will replace nearly all jobs.

We are envisaging the complete opposite, in talking about working based on abstractions. This is something digital technology people may not be in the habit of doing, because they are used to working in a highly precise, granular way, because otherwise a computer system will still crash, due to the thousands of potentially conflicting features and settings.

We want digital people to accept that healthy society in the 2020s relies on healthy organisations. Healthy organisations rely on people and their practical judgement, and machines cannot make practical judgement.

We also want digital people to accept that not everybody needs to have detailed technology understanding, and if they don't understand the detail of technology that does not make them less valuable people, because they may have detailed understanding of other subjects which are just as important.

1.10 Why we cannot connect technology directly to decisions

Many people in the technology world imagine a future where decisions can be made directly by digital technology and see this as their goal.

While this may be possible within highly automated industries, most of reality is just too complex for this to ever happen.

Consider the complex decisions we make at home, particularly if we have children. We are simultaneously making decisions about how to spend money, how to steer our children to grow into capable adults, while also agreeing to what our children ask us for, to some extent. We want our children to eat and sleep well but without forcing them too much. We need to make a lot of judgement.

There is nothing at all impressive or surprising about this ability to make judgement, since it happens in every family around the world. But it would be a big challenge to program a computer to make all these decisions for us. And how would the computer handle a situation which has never been seen before, something which happens to a parent almost daily?

In our organisational lives, in comparison with our home lives, the complexity levels can be just as high. But because everybody has a different experience in the working world, there is less opportunity to talk about them with others who may understand it.

The volumes of data are much more - if we are dealing with tens of thousands of patients in a surgery, say, rather than 3 children. We need computer systems to handle this data. But that still doesn't mean we should default to computer systems to handle the decisions about them.

1.11 Comfort zones and abstraction

A core reason that people don't create abstraction couplings is because it takes them out of their comfort zone.

It is a core desire of any person to be comfortable, and that doesn't just mean physical or emotional comfort, it also means being comfortable with our ideas and ways of thinking about things.

Without directly intending to, we build up walls around our comfort zone, so we only let people into our inner world where they share our ideas and ways of thinking about things, and other people do not feel welcome.

Having a sense of status also increases our sense of comfort, and again is a core desire of just about all people, although it may not be consciously intended. But a sense of status also denies us the humility to work out how to abstract something we understand so that someone else can understand it. To the contrary, we may be subconsciously using over complex language

and acronyms to make our working discussions hard for an 'outsider' to follow.

These are all reasons why an abstraction coupling can be hard to make, and why it can take people with a certain mindset to make one.

1.12 If we do not have technology abstraction

Our current pathway is ever increasingly complex technology and increasing lack of understanding of it.

We have technology specialists who do not understand much in the world beyond technology, and who might believe that technology understanding is the only understanding worth having.

The rest of us lose the ability to engage deeply with anything or understand how it works, because we become pawns of the technology industry, whether on computer games or social media, getting sucked in although not going anywhere.

The value of domain expertise itself gets lost as everyone becomes either a programmer or someone being 'programmed' by their use of digital technology.

When it comes to politics, people expect simple solutions, because they are not used to trying to understand something in depth. Politicians who do try to understand complex problems do not get any respect in return. Populists who sound like they know what they are doing can go a long way.

Environmental problems cannot be solved because they require either an in-depth, careful understanding of a situation, and a willingness

to change, or both, and the society does not support either of those.

The digital technology we have is designed around data flows, not domain understanding. Developers are focussed on technical features like whether the software can run on the infrastructure they have, and what features they need to get the data they want. Software designers do not have time to go further than this to understand the domain where their software will be used.

Digital people are often not that interested in how people make decisions anyway and may believe the decisions will soon be made by computer.

2 Some abstraction coupling concepts

2.1 Abstraction couplings are mundane in other industries

To help understand how an abstraction coupling can be used in digital technology development, we can consider how they are already used in domains other than the connection between digital technology and domain expertise.

In many areas, making abstraction couplings is mundane. Many people have jobs which largely involve connecting one thing with another.

An architect's role may be largely about connecting the wishes of the building owner with the practicalities of what can be built.

A marketing manager's role is largely about connecting desires or requirements of people in the market with what the company can supply.

The role of a musician, or any creative person, can be largely about abstraction coupling – creating something which makes someone feel a certain way or which takes them to a certain place. It is not just about creating something.

In our own industry, maritime, people have highly evolved means of communicating to each other what is going on, which convey large amounts of understanding with very little potential for misunderstanding, in a small number of words.

The communication goes further than to explain a situation, it indicates to someone what is different this time and what they need to do. This could be considered like a language.

For another example, consider two plumbers talking to each other to describe the challenges of a job. They would do it in a completely different language, and with much more efficiency, than a plumber describing a problem to a homeowner.

Human driven organisations, like government, can work at all abstraction levels.

When government is done well, this is what happens. People get taken care of, the details get sorted, the big decisions are made well. If you ask a British government person what they think are the top ten strengths and weaknesses of the UK, a highly abstract question, they could probably answer.

The human brain, anthropologists have said, evolved to what it is, due to the pressures of understanding our place in 150 person tribes. Our relationships, our status, where we stand with everyone. We had to do a lot of abstraction and integration in our minds.

2.2 The abstraction coupling can be a conceptual place

Our idea of the abstraction coupling for digital technology is a 'place' where domain experts can 'go' to understand what the digital technology is doing, or design what they want it to do.

Domain experts can express their models about how they think the technology can work, and the technology can be built according to this model, without further judgement needed from programmers.

It is also a place where digital experts can go, to understand what they should build. Nobody is overloaded with information they do not need.

The abstraction coupling is something which only exists in our minds. There are many other things which exist only as things in our minds which we work with every day in the digital world - such as the cloud, artificial intelligence, digital twins.

We could never fully define what a 'technology abstraction coupling' looks like, because domain experts begin with different levels of competence and understanding of digital technology, so the specifics of what should be included in the coupling will vary.

2.3 Technology's lack of abstraction

People developing digital technology are mainly absorbed by the challenge of making the technology work.

But if they don't go any further, that's like an architect whose sole focus is making a building stand up, or the musician whose focus was playing the notes in the right place. Or creating an artwork without a care of how someone might or might not react to it.

People in digital technology companies may think about abstraction the most when they are marketing, telling a story about what their technologies can do. At this point there is a big incentive to think hard about abstraction to connect well with the listener.

But this is only a small part of technology abstraction. It may not even be an accurate one, if they are describing what their intention was when making the technology, or a story they think will resonate, not what it can actually do.

Digital technology does not abstract itself at all, unless explicitly programmed to do it. This

is because abstraction needs judgement. The only way a computer can do judgement is if someone has already encoded the necessary judgement into rules. Such as the people who programmed the algorithms which drew up Google Maps from satellite photos and other sources. Or people who program a computer to generate a company balance sheet from a mass of transaction data. Or some very planned and sophisticated machine learning system.

Computer coding languages add in some abstraction, where the relationship between abstract and specific is fixed by the designers. The program itself has no awareness of the relationship or why they were set this way or that. They must still provide all the detail, which is needed for the system to run, since the software in most applications cannot exercise any judgement itself.

Technology can also work at an abstraction level through trial and error. Like when thousands of internet companies focus on the details of their businesses, and the market decides which ones win, based on how much customers like them, which can be an abstract concept.

For example, supermarkets run multiple experiments to learn how people react to different products put different places. Conclusions can be drawn if there is enough data and products are kept in the same place for enough time.

But this technique could not be applied to purchasing software for industry. There can be a very complex gaming between buyer and seller, and to win the game involves knowing which products are seen as most valuable and most prone to gaming, but also what circumstances are relevant.

2.4 Practical judgement in technology abstraction

When making an abstraction, practical judgement is important. We can observe that some people have very good practical judgement, and all people have it to some degree, otherwise they could not survive.

Practical judgement is about knowing what the goals are, and then what move would best help achieve it. Then separating things out, so that we can focus on specific tasks, or give the tasks to other people to do, with the result of all the tasks taking us closer to our goal.

Practical judgement is important for people who build abstractions of digital technology systems, so they provide the most useful information to people.

For example, an abstraction of the data in a corporate systems user directory, which would tell you that you have 10 ex-employees who still have user access to the corporate network.

A system that could tell a maintenance manager which parts of their world would most benefit from maintenance work today, or to tell someone responsible for reducing CO2 emissions which parts of the facility get the least value for the fuel they consume.

It might can do this by showing things which I find interesting because they relate to my goals; showing me where to find things which relate to my goals; showing elements from other software which relate to my goals; presenting a solution which is useful to me.

2.5 Goal modelling

Successful organisations can have many layers of tasks or goals. Designing these is something we could call ‘goal modelling’. Although it is rarely done as a conscious activity.

Goal modelling involves understanding the supporting process to get to the most important goals.

Most organisations have been doing what they currently do for some time and were developed by people who had previously done something similar in another company, so the goal models persist over many years.

Domain experts often use the same processes and sub-processes for years, so can describe them to others in a very straightforward way. They don’t need to think about the full details every time.

War goal models are readily understood because we learn about them in history. Primary goals may be occupation of territory, blocking an aggressor, or behind that keeping a politician in power. Sub goals relate to soldiers, their morale and fatigue, equipment, population, adversary country population, fuel. Trying to stop more than one battle happening at once.

An abstraction coupling between the digital and domain expert world should convey the various goals at different levels and the means that the technology can align with the goals.

For example, in industrial safety, a domain expert may know that a certain sensor is particularly crucial in giving a warning of a pressure increasing in a tank, and so the digital systems need to be designed to alert domain experts if that is seen. At the same time, there are other sensors which are not particularly important to any goal and need to make their information available only to someone who is looking for it.

A domain expert may look for certain patterns in pursuit of goals. For example, a doctor's goal is to diagnose a patient. While different root problems may cause the same symptoms, the root problem could be identified from a pattern in the symptoms. The digital systems can be programmed to spot if these patterns are occurring, not just report on the symptoms.

Technology abstraction could help provide better help systems in software, if they can connect with an abstracted version of what the person is currently doing, based on an understanding of their goal models. It can then give them more relevant advice. But normally software applications don't have the ambition to have a broader understanding of the world than the one which involves a person's interaction with the software.

2.6 Domain expertise ends with specific decisions

With all this discussion about abstracting, it is important to remember that the goal is something specific. Domain experts are not understanding something for the sake of it. Their end goal is to do something. To take the organisation somewhere, develop an effective schedule, work out what to buy, or what to change, for example.

It is one of the amazing things about the human mind that we can do this, yet we do it all the time without thinking about it. For example, 17-year-olds make decisions about what university course to take based on abstract understanding about their skills, their interests, and perhaps the job market. And they are not even aware that they are making such a big and difficult decision, they are just doing something which feels right to them.

In the domain expert world, the decisions we make are not necessarily directly related to the digital tools we use but can be indirectly related.

Examples where they are closely related could be a car speedometer which tells us which speed we are driving at, which informs our decision about whether to speed up or down. An example of where they are loosely related could be a decision to transact (such as to invest money), which has no relation with the digital systems we use to execute the transaction.

2.7 Writing can be an abstraction coupling

Where do we start looking for people who have skills to do an abstraction coupling between the domain and digital domains? One starting place is writers.

The abstraction coupling doesn't necessarily need to be written words, but most written words are abstraction.

This may not be how you perceive writing. You may say writing as an act of putting words on paper or describing something.

But the act of writing also involves understanding whatever it is you are writing about, and figuring the key points, and then writing them down. This is what universities ask for when they ask someone to write an essay about something. It is also what journalists do.

We could make technology abstractions by having domain experts and digital experts sitting in the same room trying to explain what each other does in a way that the other can understand. But it may be more efficient and effective to have a skilled writer meet the domain experts and digital experts separately, understand what they

do as an intermediary, and then write it down so the other can understand it.

The writing can itself form the abstraction. An abstraction does not have to take the form of words, it can also be a map or connection diagram, or description of a standard. But words can be used for these things too.

Writers can also separate concerns, an important part of many technology abstractions. An organisation has people in different roles, and the digital technology work also has people in different roles. They don't all need to see the whole thing, they can just see the part which relates to what they need, but in a way that all the components come together to make a bigger entity which works.

Good writing creates enormous efficiency, in the sense that it may take several hours of someone's time to understand what someone else is thinking or doing, if it involves going to visit them and sitting down with them. If a writer spends the time doing it instead, and accurately selects the parts of their story most relevant to someone else (in other words, makes a good abstraction), multiple people can read the written output in a few minutes.

And the writer does not need to be a domain expert themselves. This could be a role for someone relatively junior, perhaps straight out of university. The level of skill needed to understand what someone is doing, or something that they made, are much lower than the level of skill needed to actually do or make it.

The writer will probably need to be a friendly and pleasant person, which a domain expert would like to welcome into their world and spend time with. The writer may need some basic domain or technical knowledge.

The writer should be curious and eager to learn. Being a good writer normally requires being very receptive – whether to people talking to you, or things that you read. The writer should have practical ability to separate wheat from chaff in what they hear.

Most importantly, there are people who enjoy writing. Or at least enjoy the idea of being a writer. And these are the people who may enjoy technology abstracting, if they also have the ability to understand organisations and technology.

Good writing is about delivering to a goal. The goal is to serve the reader, which means helping the reader meet their own goals, whatever they are. For writing, it is usually to understand something better.

There are other ways to share knowledge than writing with independent writers. For example, many instruction manuals use pictures rather than writing. Domain experts can record videos of themselves speaking or write themselves. But writing is a powerful communication means, available to nearly all of us, as creators and receivers.

We can also imagine, in future, a technology which can automatically read natural language writing like computer code and create software from it.

2.8 Abstraction coupling as a business proposition

For these ideas to become real, someone needs to be able to make money out of them, or they need to provide tangible value for an organisation. It takes investment in people to make an abstraction coupling.

A good place to start is in identifying where tangible value could be added from an abstraction coupling. This would be different for every type of customer.

As an example of where there is direct business value, consider a company involved in heavy manufacturing, which is facing new demands about emissions.

This company is asked by its own customers and regulators to provide data about emissions involved in providing the product. It may also need to buy credits to emit. It also wants to find low-cost ways to reduce emissions.

It would be most helpful if senior management had a better idea about where their emissions are – from their purchases, their own operations, and any gas leaks. They need situation awareness. An abstraction coupling could help provide this.

Not all the data they need is available – the abstraction coupling would also make this clear.

Now there is a pathway to a commercial offering and a project with a tangible value. People's time can be allocated according to the budget.

Perhaps we can envision an 'abstraction coupling creator team' – a group of people whose role is to talk to domain experts and ask them what they think their software should look like, and then talk to digital people and show them the abstraction to discuss whether it can be built.

Then they could go between one group and the other until they have developed an abstraction which defines digital technology which domain experts think would help them, and which digital technology people believe they can build.

Perhaps one day this abstraction can be automatically converted into digital technology.

3 Technology concepts

3.1 Introduction

In this chapter, we'll go deeper into some specific technology challenges and how abstraction can help solve them. Digital integration, making software adaptable and extensible, understanding cyber physical systems and AI.

We'll also look at ways that abstractions can be used to help technology to do more, with modularity and executable models.

3.2 Digital integration and domain expertise

By developing technology abstractions aligned with the needs of the domain, we see a pathway to solving one of the most complex and important digital issues of 2022 - data integration.

The basic idea is that while software products for the same domain can be very different, the domain itself is probably the same everywhere.

So, the best way to connect two software products designed for the same domain may be to go via the domain's needs.

Every digital technology system has elements of an internal map of how it works, connecting elements in the technology to elements in the real world
To integrate technologies together, you can draw these maps, and then show how the maps connect, where both technologies describe the same thing in the real world.

The real-world changes only very slowly. But software companies are always trying to do more with their software to incorporate the real world.

And once you have these maps, they can be integrated quite easily. Integrating one map with another is usually easy to do – whether it is layering a map of roads onto a map of the shape of land or connecting one person's plan with another.

To illustrate – most companies do their accounting and financial management in a similar way, such as making reports showing their balance sheets, profit and loss, cashflow forecasts, outstanding invoices and outstanding bills to pay. These reports drive their decision making. But accounting software packages have different data structures inside them.

So, if you wanted to integrate one accounting software with another, it is much easier to do if you just bring together the reports of 'outstanding invoices' from each software package, rather than try to map the individual data fields together within the software.

For another illustration of data integration via the real-world domain, consider a group of musicians from completely different musical backgrounds trying to play together. When they play music, some use music notation, some use other systems, or no notation at all. This is their equivalent of data.

But they all have one thing in common, the real-world demand for creating music that people like. If they focus on whether they are making music which someone likes or does not like, "integrating in the real world," the lack of integration in their processes is no longer a problem.

3.2.1 Problems with the conventional approach to data integration

The conventional approach to data integration is all in the 'data domain' - connecting a data field in one database or software package with a data field with another.

This can involve enormous amounts of difficult labour. It can be very fragile, because as soon as one of the digital systems is changed, it no longer works.

When integrating data by mapping together fields you can also run into problems that someone could not anticipate if they were not a domain expert. For example, the same industry may use the same term to mean slightly different things.

In the maritime industry for example the time of the 'end of voyage' can be different depending on your role. If you are a charterer hiring a ship it means the time the ship is no longer 'yours'. If you are a seafarer, it means the time you can leave a ship.

If you are a crewmember staying onboard, the end of voyage may mean the time you reach the destination port, or when unloading is finished. If the ship needs to travel unladen to its next loading port, the 'end of voyage' might not happen until the ship is ready to load its next cargo.

3.2.2 Some analogies to show weaknesses in the data field matching technique

Here's two more analogies to explain why the conventional approach to data integration, mapping together data fields, doesn't work.

Consider a plumbing problem involving combining two pipes. Initially it looks like a connection problem – are the pipes the same size, can we get a connector? But the problem is much more than that. A plumber would need to know about the pressures in the pipes – very different if it is a drain to a sink (no pressure) or a pipe at the bottom of a 30-story building (very high pressure). It is the context behind the pipes which matters (the building they are part of), not the pipes themselves.

Or consider a writer combining two pieces of text. You could do this using grammatical rules, which also a computer could follow. But that is unlikely to make any sense to the reader. The writer would think about what the writing is aiming to convey and the best way to do that, with the best flow of ideas to make something easier for a reader to absorb.

So similarly in the world of data integration, the strongest way to connect two different software systems is to understand the context behind the data, how the data was created, what it is part of, how it will be used.

And bear in mind that although the technicalities may change (a different pipe fitting, a different choice of words), the background context will only change very slowly if at all (the structure of the building, the interests of a general reader). So, focussing the integration on the background context means we are creating an integration which will probably be more robust and long lasting.

3.2.3 People are master integrators

People are master integrators. We can integrate mental data about a whole range of different things – the weather, the urgency of our work,

our need for physical exercise, our children's demands - and effortlessly decide what time to get up in the morning.

But that only works when we as people can understand the data we are looking at. And that is the difference between organisational data being integrated conceptually by a domain expert, who does understand the data, or a data integration specialist, who doesn't.

If it is data from industrial software, a domain expert probably would not understand the core data – as an accountant would not be able to tell much from just being given access to the databases behind the accounting software. But an abstraction created from the data – such as, in the accounts example, a list of overdue invoices, is something that would tell something.

The domain expert understands cause and effect in that domain – what might indicate that the baby is getting better or sicker, or what the causes might be. This understanding is informed by the different digital systems and based on a career of training and experience.

3.2.4 Integrating at the domain level

To explain this suggestion more technically, instead of trying to integrate software using APIs, which theoretically allow software packages to be linked together, we should integrate software at the level domain experts interact with it.

Integrating each application at points where it interacts with the real-world domain expert level is scalable – the integrations can get bigger and bigger. It is not limited by the

technical resources available, and multiple systems can be integrated at once.

Where some applications have a different process, the abstraction can be extended for this.

3.2.5 Some tough problems we could solve

Good data integration could go a long way to making our world better, although it may be something you have never thought about.

Consider some of the hardest social and organisational problems of our time, such as tracking the spread of a virus, accommodating refugees, minimising climate emission.

They all involve data and decisions. But the decisions involve enormous amounts of disparate data, which does not directly connect to a decision. In these cases, the decisions are about governments calling lockdowns, which parts of a country can best accommodate refugees, which choice out of several options is best for climate emissions. Data might be about virus infections 3 days ago, refugees spotted on boats, or emissions from a specific industrial process.

Or consider the most persistent, seemingly hardest to solve, data related problems in 2022. These often relate to data integration. You have a long queue at passport control. Your train comes to a sudden stop because one system could not ‘talk’ to another one. The opening of a new train line is years delayed due to problems integrating signalling. Your Wi-Fi does not allow you access. These problems can all relate directly to data integration problems.

The idea can be taken further by making the integration 'executing', i.e., where the output

of the integration informs other software systems, so they can run processes based on it.

3.3 Adaptable and extensible software

Software built from abstractions should be itself easier to understand, easier to extend and adapt.

Much software in use in 2022 has been built gradually, sometimes over decades. It may do the job because people have tweaked it until it runs reliably enough. But that does not mean anybody knows how it works.

This software is like a building where additions have been made gradually over decades, until nobody knows how anything connects, and what is supporting what. Although with the building, at least you can usually see it clearly.

Code can be invisible if there is enough of it, just like detail of any sort can be invisible if there is enough of it.

3.4 Cyber physical systems

“Cyber-physical systems” is a name for a system which involves physical and digital components and a relationship with people.

The term is mainly used for a system which gets too complicated to understand, and where that becomes a problem, for example when we are talking about smart grids, autonomous cars and planes, medical monitoring, industrial control systems and robotics, or organisational cybersecurity.

Often, the system is managed by learning cause and effect, such as this problem sometimes occurs when this sensor breaks, and this is how

we fix it. But that means we are not necessarily aware of what is going on, or problems emerging, until they become problems.

It is very hard for one individual to fully understand how an autonomous car works, for example, so hard to understand the cause of any problem.

An abstraction coupling could be a solution to the problem, if it can present what us with the driving forces behind the software, rather than all the detail. This would make it easier for someone to understand it.

3.5 Abstraction and AI

Abstraction can help us understand artificial intelligence systems.

If the AI is doing harmless things, like suggesting spelling errors or filtering our e-mail, then it isn’t so important to understand it.

We don't understand the details about how a spam filter works, but we can see that it works reasonably well and get a sense of how it does it.

But when we come to using AI to make decisions about whether to release someone from prison, or where to invest money, then it gets more important that we have domain experts which do understand it.

It is insufficient to say, the algorithm needs technical competence to understand which you don't have. Few people if any have knowledge of both how to make a fair adjudication process and how to directly understand an algorithm.

If we can write abstractions which explain how the AI works, that would be very useful.

3.6 Modularity and abstraction

One way to make the digital world easier to understand is to use modules – which we could also see as standards or templates.

Think of this like making a model of the real world using Lego bricks. Defining anything in the real world can be extremely complex, but we can make an instruction book for how to build something from Lego which a child can follow.

This works because we have standards. With Lego the standard is in the brick size and the size of the bumps which push a brick into another one, making it stick together while also possible to pull apart again.

In the digital world we can have standards for anything. For example, a standard for a group of data which need to be associated with a transaction, or a way to describe carbon emissions comprehensively.

You don't need to understand how a module works; you just need to understand how to put them together. In the same way as you only need to know how to plug a mouse into a computer to use it, you don't need to know how the computer or the mouse work.

3.7 Executable models – technology understanding us

Where technology abstractions could end up is where the technology can run automatically from the abstraction.

Imagine if someone who knows the domain makes a design for how the software can run, and the software can run automatically from that design.

If the design is detailed enough, a computer can follow it directly.

This is beyond state of the art in 2022, but not too far beyond.

We are starting to see good systems for understanding and translating written word for example.

If people are writing in a way where all ambiguity is removed, or on a narrow range of subjects, systems can understand the written word.

Similarly, if code is written in a very clear and unambiguous way, it can be possible to express it using written word.

We are seeing a big growth in 'low code' systems which promise to make it possible to create code automatically from models, not necessarily made by a programmer. As of 2022 low code is useful for simple programming tasks, such as viewing data from a database or straightforward transactions.

A design for software could be written in other forms which are easier for a computer to follow, such as a diagram showing what leads to what and how things connect.

The design contains all the logic which the computer needs. Since code is a machine-readable form of logic, there is no need for any code if the machine can read this logic directly.

We can call this 'executable' designs, since the design can be automatically 'executed' by a computer.

To make such executable designs, domain experts must know what they want and be able describe what they want in enough granularity for a computer to run it, with no further judgement required. And there needs to be a mechanism for a computer to read it.

This would be very different to current working practices for designing software, where we might start by thinking of what user interface we want and how to make that user interface display the data in the way we want it.

That could be equivalent to asking a power station engineer how the power station should work, and they start to reply by asking you what page layout you want the answer to be written on.

4

5 Some domain specific ideas

This chapter shows how abstraction couplings might be used to solve three big organisational problems in 2022 - reducing organisational CO2 emissions, improving industrial safety, and improving cybersecurity.

Take these ideas as illustrative only, particularly if you already have domain expertise in one of these sectors. An actual abstraction coupling which was valuable would need a lot more detail than we have here. The purpose is to show what might be possible.

5.1 Reducing organisational CO2 emissions

The best way to reduce organisational CO2 emissions could be to have a system which will tell us the CO2 impact of every decision, any time we make a decision. But how could we have such a system?

Understanding the levers behind reducing CO2 emissions is immensely complicated. To illustrate this, imagine if you wanted to reduce the CO2 emission associated with your house, including from heating and electricity consumption.

You know your electric and gas bills. You have a vague idea of where most of the emissions are made - probably heating the house and hot water. But you don't want to tell your family that the house needs to be colder, they should have showers rather than baths. You don't want to spend time adjusting heating in different rooms. You could spend your time switching off lights and unplugging chargers but you're not sure if it really makes much difference. More insulation in the walls and ceilings would be very expensive.

In short, you have many different options with multiple factors involved, and it is very hard to go further than guesswork in making decisions about what to do.

Industrial life is like this but more complicated, with many different emission sources, much bigger emissions, more potential levers, and more difficulty knowing which ones make a return on the effort involved.

Software can probably do a lot to fix it. But to be helpful, it needs to be software which does more than tell you where the emissions are. It needs to be software which tells you what the carbon impact would be of any option you might make, before you make the choice.

The way to solve it could be done with a mixture of domain expertise and digital technology, and intermediaries conveying understanding between the two worlds.

We can map the decisions which we commonly make, and the biggest CO2 emissions associated with the decisions, and whether we have information available to us to tell us what these CO2 emissions are.

5.2 Industrial process safety

Industrial process safety is the challenge of preventing fires, explosions and accidental chemical releases in chemical process facilities. Big disasters in other words.

Accident reports show that these disasters are rarely caused by one single factor which should have been obvious at the time. They are caused by multiple factors, which somehow found their way through all the various safety systems.

The way to improve the situation may be to have domain expertise and digital expertise working together in the right way, with abstractions to improve understanding between them.

Domain experts can explain which pieces of data, such as from a sensor, can be most crucial in information of a problem emerging, or how else they might be able to talk about something important from data.

Digital technology experts can build systems which convey the right alert at the right time. So, they are not just sharing a 'status', they are telling us something important and specific.

5.3 Cybersecurity

Improving industrial cybersecurity often comes down to situation awareness. Do the organisation IT departments know what is going on? Are the IT systems capable of effectively determining if someone giving them instructions is legitimate?

Our software will happily give entry to a hacker, just because they have the right password, although what a hacker is doing with the software is completely different to what the legitimate user is doing, such as downloading all the data and sending it to North Korea.

This is not how it works in the real world. The doorman of a hotel does determine who is allowed access to the hotel based on asking them to show their rights to entry. There are many 'common sense' based means of determining people who may intend harm and denying them entry.

Domain experts can design systems which would help improve cybersecurity using practical judgement. If this happens, then send an alert. If you see this, then block access. Tell us the

list of people with user accounts who have left the company.

6 Living with or without the abstraction coupling

If we don't do anything differently, we'll see an ever-growing gulf between the digital and domain expertise world.

This could lead to all kinds of problems – people feeling less connected to how the world works and a lack of faith in democracy; many people finding it more difficult to find jobs which make them feel empowered and part of society; good environmental decision-making being much harder to make.

This is a world where digital technology companies can thrive though, particularly if they can help organisations manage without pesky people.

On the other hand, if we can better integrate our organisations with digital technology, we can have organisations which are most effective - safe, productive, reliable, profitable, that do an enormous amount to keep society going.

These organisations are more likely to be able to thrive in a more difficult world, manage with more restricted resources, or achieve their goals with less environmental damage.

We can create good roles in life for people who are good at coupling and working out abstractions. We can have healthy organisations, and people able to develop new skills.

Hopefully in this book we have made a case that this is worth thinking about and offers opportunities. Even if we have not provided all the answers, we have opened the door to some ideas and concepts which can lead to a better future.