



Estimating and forecasting

This publication forms part of Value, Flow, Quality® Education. Details of this and other Emergn courses can be obtained from Emergn, 20 Harcourt Street, Dublin, D02 H364, Ireland or Emergn, 190 High St, Floor 4, Boston, MA 02110, USA.

Alternatively, you may visit the Emergn website at <http://www.emergn.com/education> where you can learn more about the range of courses on offer.

To purchase Emergn's Value, Flow, Quality® courseware visit <http://www.valueflowquality.com>, or contact us for a brochure - tel. +44 (0)808 189 2043; email valueflowquality@emergn.com

Emergn Ltd.
20 Harcourt Street
Dublin, D02 H364
Ireland

Emergn Inc.
190 High St, Floor 4
Boston, MA 02110
USA

First published 2012, revised 2021 - printed 16 July 2021 (version 2.0)

Copyright © 2012 - 2021

Emergn All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, transmitted or utilised in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without written permission from the publisher.

Emergn course materials may also be made available in electronic formats for use by students of Emergn and its partners. All rights, including copyright and related rights and database rights, in electronic course materials and their contents are owned by or licensed to Emergn, or otherwise used by Emergn as permitted by applicable law. In using electronic course materials and their contents you agree that your use will be solely for the purposes of following an Emergn course of study or otherwise as licensed by Emergn or its assigns. Except as permitted above you undertake not to copy, store in any medium (including electronic storage or use in a website), distribute, transmit or retransmit, broadcast, modify or show in public such electronic materials in whole or in part without the prior written consent of Emergn or in accordance with the Copyright and Related Rights Act 2000 and European Communities (Copyright and Related Rights) Regulations 2004. Edited and designed by Emergn.

Printed and bound in the United Kingdom by Apple Capital Print.

CONTENTS

Introduction 1

1 The need for estimates and forecasts 3

- 1.1. What is an estimate? 3
- 1.2. Why we need estimates and forecasts 7
- 1.3. Why we get estimates wrong in the first place 7

2 Estimating techniques 13

- 2.1. Past data 13
- 2.2. Estimation by comparison 17
- 2.3. Decomposition and recomposition 21
- 2.4. Wideband Delphi – the Wisdom of Crowds 25

3 Estimating value 27

- 3.1. Fast benefits assessment 27
- 3.2. Small bets 29
- 3.3. The value of variability 32

4 Forecasting 36

- 4.1. Pipelines 36
- 4.2. Learning from history 37
- 4.3. Building models 38

5 Conclusion 42

Bibliography 44

Appendix 47

INTRODUCTION

How many piano tuners are there in Chicago? This is a question that Enrico Fermi asked his students. He was famous for teaching his approximation techniques to students in situations where they couldn't validate or confirm the results. He wanted to teach people that they did know something about most domains they want to explore. By using fairly simple questions, making assumptions visible and exploring logical associations "Fermi Questions" could provide lots of insights for people to make decisions.

For example, in Chicago between the 1930s and 1950s there were approximately 3 million people. There were probably between two or three people in each household. He then went on to suggest that there were probably between 1 in 10 and 1 in 30 households who had a regularly tuned piano. It is unlikely that there are many people who have their piano tuned more than once a year. A piano tuner could likely tune between four and five pianos a day with travel time. And, finally, the tuner might work up to 250 days per year.

This turned a question that most of his students originally exclaimed, "We can't even guess that!" to a range of between 20 and 200. Most of the students ended up guessing around 50 piano tuners. When Fermi then showed people how many there were in the phone books, they tended to be fairly close to a reasonable estimate.

Fermi went on to show his students that this approach is helped by making assumptions and probabilities visible. Communication increases because a group of people can question which variables have the biggest impact on an estimate.

Fermi, who won a Nobel Prize in 1938, put his skills to good use in 1945 when he demonstrated, using ripped up pieces of paper and some wind, the blast effect of the Atom bomb – he determined the lower limit, and determined the yield estimate to be 18.6 kilotons. He was keenly aware that using simple observations and experiments to build up pictures of the world helps with planning and execution.



Figure 1. A piano tuner at work

In his book, *How To Measure Anything*, Douglas W. Hubbard, explores many more of the challenges and problems people have with their approach to measurement, estimation and prediction. He demonstrates that people really are poor at estimating, and that they have an unrealistic view of their ability. A great illustration of this overconfidence is the famous experiments involving drivers and their driving ability. In every example of this experiment over 69% (and in the most extreme case 93%) of people claim to be above average drivers. That is, more than 50% of people believe they are above 50% of people. Illusory superiority (also known as the Dunning-Kruger effect) is extremely common, and takes place on a daily basis in work too.

At the end of this session, you will be able to:

1. Describe why and where estimations and forecasts are useful.
2. Understand and appreciate the strengths and weaknesses of different estimating techniques.
3. Explain why using crowds (groups) and data is useful in improving estimates.
4. Design an approach to improve your ability to forecast future value throughput.
5. Create strategies for collecting, identifying and providing key value and effort information in to the planning and execution processes.

1 THE NEED FOR ESTIMATES AND FORECASTS

No improvement in management, project, product or software development methods over the last three decades has magically fixed the essential problem that we need to make decisions now that will affect the future.

Both forecasting and estimating are attempts to make guesses and assumptions about the future. We assign a great deal of importance to this activity; often we make promises and commitments on the back of our estimates, and yes, we often get it wrong. Sometimes we get it wildly wrong – we've discussed some of these infamous projects in the Why Change session: projects that overrun by years and millions of pounds, with estimates out by 200% or more. These disasters end in public embarrassment and often court cases.

These are the situations we want to avoid. Rather than needing to be accurate to within days, we need to avoid the massive inaccuracies! Or at least, we need to provide planning and governance methods that help us manage the inherent uncertainty and risk present in all development and change initiatives. That's why this session focuses on the tools designed to make our assumptions, guesses and forecasts more visible and as robust as they can be when we have little data, and help to improve them over time. We want to avoid the really costly errors and we want to examine ways that we can balance the need for fast information gathering rather than highly complex equations which often provide only a spurious sense of accuracy.

1.1. What is an estimate?

The first problem with estimation is that few people really agree on a definition. And even if they do, what they say they mean by an estimate and how they then treat it is not always the same thing. The other major issue is that when people are talking about estimation they're really focused on predicting costs. Estimation puts equal (and arguably more) importance on figuring out the value side of an equation, and the associated probabilities.

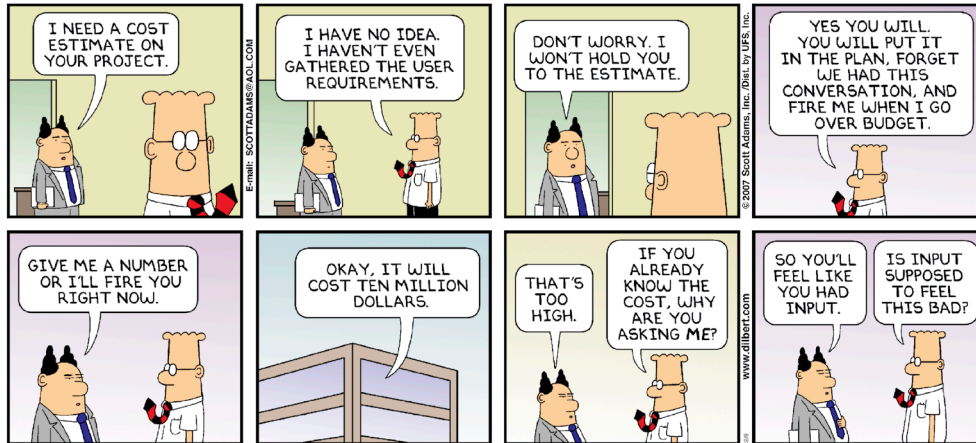


Figure 2. How not to cost estimate your project
DILBERT © 2007 Scott Adams. Used By permission of UNIVERSAL UCLICK.
All rights reserved.

You, or your team, probably have had an experience like this:

"Go on," pleads the Project Manager. "Just give me a rough idea... a back of the envelope kind of calculation. I won't hold you to it, I swear."

With a vast reluctance you grudgingly say "maybe three months. Maybe. If everything goes well and nothing changes and no one goes off sick."

In three months time, the Project Manager shakes his head at the meeting. "Guys, we are really running late on this. We told the customer we would have a release by next week – now we look like we can't be trusted."

What went wrong?

The Project Manager asked for an estimate, but what he really wanted was a commitment. He wanted something that he could guarantee to customers or internal departments. By the time this had been communicated to an external customer who began to build plans around it, the estimate had transformed into a target.

Let's begin with definitions for each term and general advice on where each should be used.

Guess:

Not a word we typically associate with the world of work, but it is probably in use more often than anyone would care to admit. A guess is to predict an outcome or event without sufficient information.

Estimate:

Estimating can be defined as 'to roughly calculate or judge the value, quantity or extent of something'. Words like judge, approximate, gauge, appraise or form an opinion of are used when dealing in a circumstance requiring an estimate. In the world of work, this is typically a prediction of how long something will take or how much it will cost based on the estimator's current knowledge and understanding. It is tentative and liable to change as we learn more. Estimates are normally based on some previously defined insight or knowledge.

Forecast:

An estimate is not quite the same as asking about a prediction or forecast. The main difference is one of choice. If you are asking, 'should I rent this bouncy castle for the school fair?', then you need to estimate how many children you think will pay to go on it in order to decide whether it is worth hiring it or not. Once the bouncy castle is up and running, if the head-teacher asks how much you think it will make, then you can forecast based on the number of children paying at the moment, how much money you will make by the end of the fair.

Forecasting is the process of making statements about future events or outcomes that have not yet been observed. They are typically made up of estimates at certain specific future times.

Target:

This can be either a business goal or a constraint. Strictly speaking it should be set by an external (customer) need rather than an internal one. For example, if there is a trade show, a legal compliance issue, a customer deadline or some other goal this could act as a timing target. This should be supported by some planning around what happens if we miss the target and thus what we might put in place to mitigate that risk. For example, would we increase our capacity or have a minimal product which we could add to as time permitted?

Commitment:

This is quite simply a promise to deliver by a given date or against a fixed price. We use commitments a lot in projects, but we shouldn't. A commitment or guarantee means defaulting to the certainty flow choice that we discuss in the Trade-offs session. As an option, this comes at a much higher cost since in order to meet it we will build in large buffers and normally nail down the scope in advance, but the commitment will likely be based on estimates and guesses which could be wildly out depending on how much information we had when making the estimates. In other words, commitments usually mean we deliver slower, at a higher cost and resist change. There is still a significant risk that we will not meet even the most conservative of commitments. These should typically only be used as a last resort when there is no other viable option.

In many workplaces, we often make the mistake of confusing these terms, treating them as if they were interchangeable and then being surprised when the estimate we used as a commitment turns out to have been wrong. One of the most important steps in estimating is to check whether you have been asked for an estimate in order to help make an investment decision, a prediction of when you think something might be ready, or a plan to hit a target.

Activity 1: How good is your estimation?

Have you ever played a guessing game? Popular ones include: guess the number of sweets in the jar, or guess the weight of the cake... the games work because people are not actually as good at estimating as they think they are!

Try the following questions. You need to set your ranges so that you have a 90% confidence of covering the real answer.

(Answers to the questions are in the appendix at the back of this session).

Question	Low estimate	High estimate
How many moons are there within our solar system?		
What is the average life expectancy of a man in Peru?		
How many books are there in the New Testament of the Bible?		
What % of the ocean is salt?		
How many people live in Timbuktu?		
When did the last person who spoke Cornish as their only language die?		
How many copies of Mao Zedong's Little Red Book have been sold worldwide?		
How many train stations are there in India?		
How many miles of coastline are there in Alaska?		
What is the average number of correct answers in an estimation quiz?		

Commentary:

If you were genuinely 90% confident you should have got 9 correct answers. How did you do?

We hope you didn't feel the need to search online for the answers! The point is not about whether you know obscure facts of geography for a quiz, but how accurately you measure your estimation skills.

Most people are not very good at it. In fact, according to studies on the subject, when people say they are 90% confident, they are really only 30% confident (getting just under 3 answers right).

Bear this in mind the next time someone asks you for a 'gut-feel' of how confident you are in your estimate.

1.2. Why we need estimates and forecasts

We need to provide estimates when we are making decisions. These might be about whether we should invest in something or not, or figuring out when we might need to start a specific activity in order to be on time for a customer event. In a sales environment we need to forecast our potential for future business – in all businesses income and outgoings need to at least balance. We need to look ahead to understand what sources of funds might exist for future investment. In other environments we need to take a look at how much things might cost or how long they might take so we can figure out if it's worth investing in. For instance, if we work in the retail industry and we're trying to prepare for the holiday period – we might need to estimate if a special offer could be achievable. If you can't make it, there is no point – if we can't get it to the customer in time. The estimate needs to take into consideration the value of the idea or feature and the effort required.

There is a law of diminishing returns that we need to keep in mind. Spending an hour to figure out that it seems like a valuable idea and we might be able to achieve it versus spending a month to get every variable nailed to basically give us the same answer doesn't seem like the most sensible use of time. If the estimates and forecasts are for a bank manager or investor, we might need to have more detail about what we plan to do and why we think our idea is accurate, but precision will still rarely help the overall usefulness of estimates.

1.3. Why we get estimates wrong in the first place

We've said this before – we get things wrong because we don't know what will happen. That holds true of estimating as much as of planning. The earlier we are in a project the less we know, and the larger the project, the greater the degree of uncertainty.

In traditional project management terms this is sometimes called the 'cone of uncertainty'. This is a theoretical model that seems to resonate well with people, but isn't grounded in any empirical evidence. In essence it states at the beginning of a project it is impossible to offer certainty about an estimate because so much is unknown. As decisions are taken, more is known and the number of sources of variability decreases. The cone narrows and thus estimates become more accurate. In many organisations this cone ends up being represented in a stage gate process that allows a Very Rough Orders of Magnitude (VROM) estimate that can have a tolerance of +100%, and at the next stage Rough Order of Magnitude (ROM) that might have a tolerance of +/-50%, with a final gate that suggests we need to be within +/-10%. These numbers are not typically based on any real facts other than more detailed requirements and costing assessments. Typically, they won't contain real exploration to help us understand if the customer really wants what is being proposed, or if the solution is feasible.

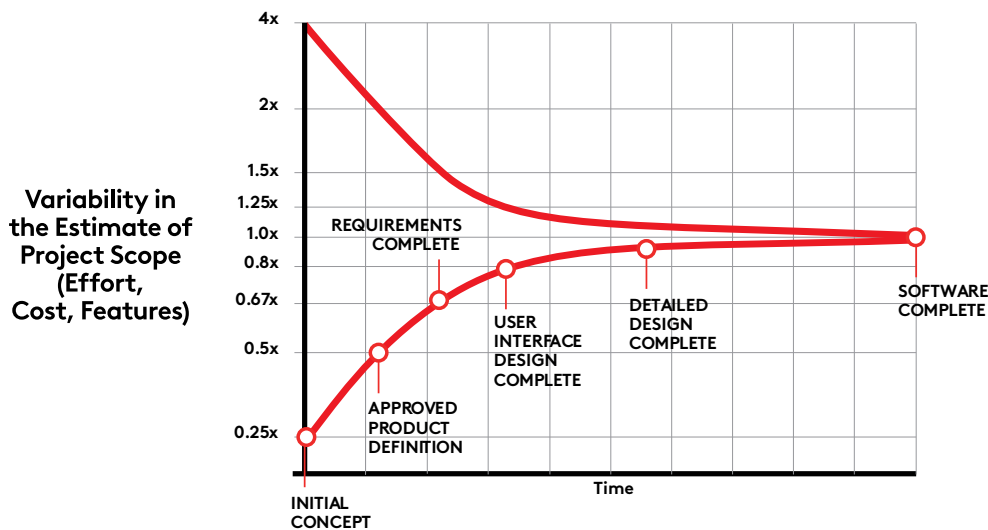


Figure 3. The cone of uncertainty

The problem with a model like this is that when we convert it into tolerances we suggest a level of precision that we don't really have. What if your estimate is more than 100% out? What if it is 400% out? How would we know? What is it that allows us to move forward with confidence?

Note we said estimates become more accurate, not completely accurate. If you make changes to your product and increase the sources of variability again, then the uncertainty of your estimates will also increase. One of the key benefits of iterative and incremental delivery is to shrink the cone of uncertainty to that particular cycle. In other words, you may have very high uncertainty (and thus potentially inaccurate estimates) about the much larger product, but good estimates about what can be done in the next month. This view of estimation fits in neatly with the planning onion, explained in the Building Plans session.

Sources of variability or 'things that will throw your estimates'

1. **Additional requirements:** If you add in new requirements then you need to update estimates on time and cost. This should be so blindingly obvious that it is hardly worth writing down. Actually, however, numerous projects are blind to the impact of changes or additions. That's not to say the changes aren't a good idea – they may be essential – but if they impact on predictions of delivery date or cost, then everyone needs to be aware of that.
2. **Missing requirements:** Lots of requirements are casually forgotten, even though they may be necessary. We covered many of these in the release planning section of the Building Plans session – is the code tested, integrated, documented, supported, etc.? Does the code perform to the required level (that is match all the non-functional requirements)? If these kinds of questions are not answered as being only the 'details' then teams can be very surprised by how much time they take on top of creating the functionality.

3. **Size of solution:** There is a close connection between really big projects and really long projects. Forgive us if this sounds unbearably trite, but it is a point that is worth making. If you only need to write five lines of code, you know a great deal more about how long it will take than if you need to write 1 million. The problem tends to be that at the beginning, with just an idea of a valuable customer need, it is not always possible to know how many lines of code will be required. Because any large solution will need so much maintenance, integration and complexity management, the length of time required increases exponentially. Diseconomies of scale – whether in team size or system size – are a very real problem in software development that contributes to inaccurate estimates.
4. **Idealised estimates:** When you said something would take two days, you often mean two ideal days. In reality the planning meeting took half a day, Joan was off sick and the boss called you in for your review. So the feature was actually ready after four days. Most teams now explicitly state that estimates are 'ideal developer days' and team leads will calculate capacity according to how much uninterrupted work time there is available.
5. **The unexpected:** Things change, unexpected problems occur, key staff leave, etc. All the types of uncertainty that we discussed in the Planning session can occur at any time – even as you approach the narrowest point of the cone of uncertainty!

Underestimating

A crucial reason why we get things wrong, as studies have shown, is that humans consistently underestimate what it will take to do something. It is called the planning fallacy and was first proposed by Kahneman and Tversky. A 1997 study of Canadian taxpayers, for example, showed that they sent their tax forms a week later than they had predicted. This was in spite of the fact that they recognised that they had underestimated the time it would take in the past.

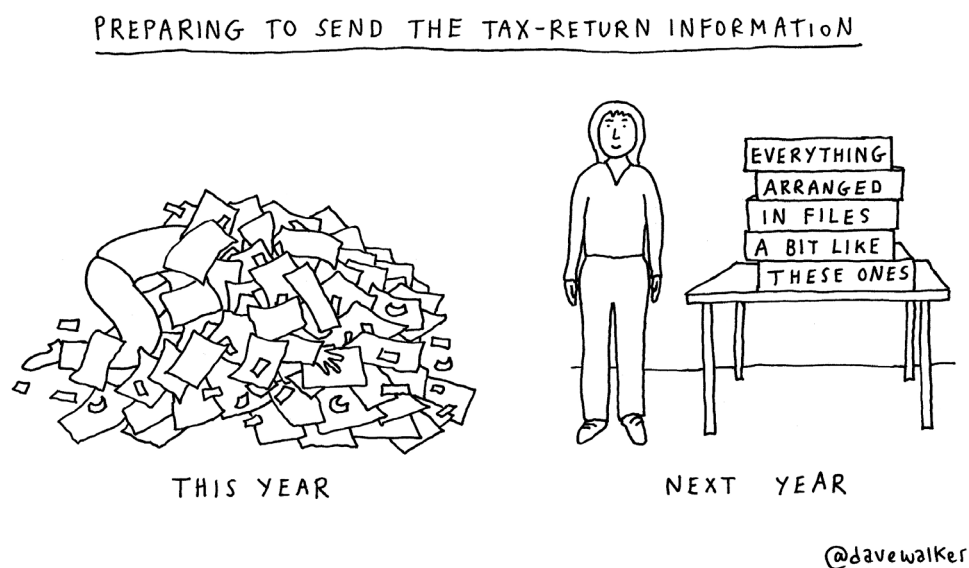


Figure 4. Completing tax returns never seems to get any easier

In the last activity, when creating the ranges for estimates, most people make them too narrow. This is the same instinct at work as the planning fallacy – even though in this case there was no real pressure to create a narrow range. You could have set a date from the Big Bang to the present day, for example, and been absolutely certain of getting the answer... but you probably didn't.

When we are genuinely under pressure of the expectations of our managers, our customers and our own professional pride, the effect can often be even stronger. According to a study quoted by Steve McConnell in his book *Software Estimation*, developers typically estimate 20-30% lower than their actual effort.

What happens when we underestimate?

1. **We plan badly:** As soon as we have a false estimate we may begin to build on it – deciding on a smaller team size, for example, which is actually insufficient to meet the product goal. Now, if we are doing everything else in an incremental manner, this is probably not too serious because after our first iteration we shall realise our mistake and adjust upwards. If we are not working in an incremental fashion, or if we are determined to continue within our blind optimism, we might just continue as normal, hoping that we will magically make up the time somewhere else. As this goes on, we get further and further behind the expectations of others. This could cause coordination problems further down the line.
2. **Quality reduces:** In an effort to claw back time or to complete features within the initial unrealistic estimates, it's easy to compromise on quality that undermines the value of your product. It may mean shipping without making valuable changes based on customer feedback because 'there wasn't time'.
3. **Late projects get later:** As soon as a project is identified as 'late', a whole set of activities snap into place which takes up more time, thus paradoxically making the project even later. For example, more time is often spent in status meetings, customer negotiations and re-estimating to ensure that the new 'target' is achievable.

Overestimating

Akin to the problems we suffer with certainty, many of us have a natural tendency to add in plenty of 'just-in-case' time. Rather than delivering early, this contingency or buffer typically gets used up. It's also commonly known as Parkinson's Law – work expands to fill the time available.

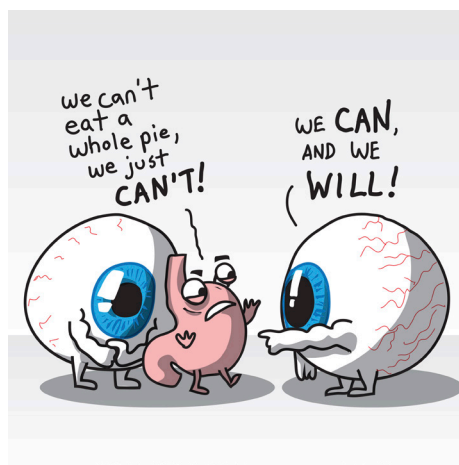


Figure 5. Eyes bigger than belly estimating

What happens when we overestimate?

1. **We take longer and it costs more:** If the work has expanded to fill a larger than required amount of time then obviously it is slower than it needs to be and usually costs more (the extra burn rate as well as the cost of delay caused by not having the product in the market).
2. **Power play:** There is nothing more destructive than the kinds of arguments that occur when managers think a team is overestimating and are determined to squeeze out the fat, and the team fear last minute changes and want to squirrel contingency time away to stop themselves from being blamed for being late. We have seen it happen both ways – developers telling business people a simple job will take a long time and business people putting unreasonable pressure on developers under the illusion that this will make them work harder. It is just a waste of everybody's time, it erodes trust and motivation, and it often leads to a lower quality outcome. Unfortunately, it is not uncommon.

Getting it right – where is estimating important?

Let's be clear – all the actual evidence suggests that we underestimate, not that we overestimate. That is not to say that projects do not contain plenty of waste – we know they do!

In general, however, we want to do two things. Firstly, help managers make investment decisions with our estimates – obviously decisions work best when based on good assumptions. It is hard to prioritise one project over another if our projections of value and effort are both flawed. Secondly, we need to offer our customers and our own businesses some predictability – in order to liaise with customers, marketing, operations and many others, it's important that we can predict relatively accurately when we think things will be ready.

So where do these two things matter?

When we know that something is really valuable to us and we don't think doing it will be very difficult, then we don't really have to bother estimating at all. Of course, we'll check that we're right about its value (testing our assumptions), but once we know it's the right thing to do, then we will probably use a technique for prediction based on past data (described in the next section).

If we think something is not very valuable but we believe that delivering it might be very difficult or time consuming, then we know that either we won't do it at all, or we need to break the idea down and capture value in smaller pieces of work.

The time when accurate estimation really matters is in a narrow band where our gut feel is that the value and effort are more closely matched. At this point we will start thinking about metrics such as cost of delay to inform us about the value, and we will also want to try and predict the effort the job will require in order to assist in our decision or prioritisation. This could also be true when we have a lot of competing ideas and we want to help communicate priorities.

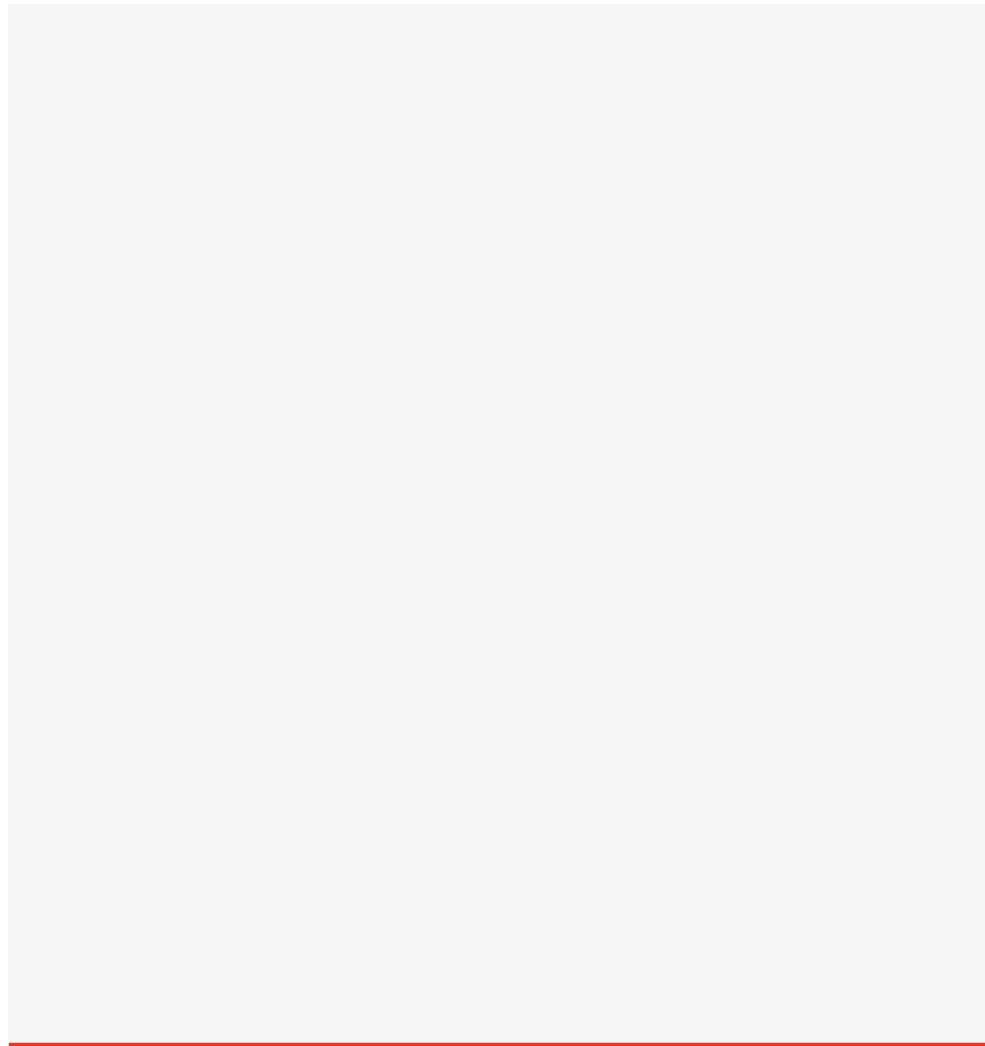
Activity 2: Find the data

This activity and those which follow all require taking information from a project that you are working on. We suggest selecting a reasonable size project (larger than 3 months work) and for which data is available of past performance).

Look back to the start of the project – were individual features estimated? Was there a total estimate for the project?

Consider progress to date. How many features have been completed? Did these take more or less time than the original estimate?

If you are not doing so already, ensure that you record and track this data.



2

ESTIMATING TECHNIQUES

In this section we go through the key methods for estimating. All have benefits and also disadvantages. We list these as the pros and cons, which should provide a guide as to when a certain method is most appropriate. This is intended to help you with the most important outcome of the section – choosing which technique is right for the particular need of your project or task.

In general, all the techniques require involvement from the development team, whether in recording their existing data or estimating likely effort. This is also far more likely to lead to the development team ‘owning’ the responsibility for the estimate – either fulfilling it or offering early warning when things seem to be off course.

Estimates should usually involve a team rather than an individual because you are looking for convergence and deviation – points at which everyone agrees (except for a couple of outliers) and points at which there is widespread disagreement. Don’t ever ignore the outliers – normally when one person’s estimate is at huge variance with the rest of the group’s, it is because they have spotted something unusual or there is an ambiguity in the requirement that should be explored. When no-one can agree it is often because the item is too big to estimate reliably or there is a lack of clarity about what the requirement really intends. In either case, the group can offer greater value than an individual, however expert their domain knowledge.

Finally, all teams should collect actual data and reflect on the difference between reality and estimation as a way of improving the process in the future. This is true even if you are not estimating, but simply wish to examine unusual pieces of data. If most stories take between 5-10 days, for example, and a particular story took 15, then the team might want to explore why and derive some learning.

2.1. Past data

This is the simplest of all estimation techniques. In essence it is equivalent to asking ‘what was the weather like yesterday?’ and assuming that the weather will be the same today. Now, we all know this isn’t entirely the case – we know, to quote Shakespeare, that ‘rough winds do shake the darling buds of May’ from time to time. But on the whole, if you were to base what you wear today on what you wore the day before, you would not be as far wrong as if you were to don woolly scarves in the summer or a bikini in the winter.

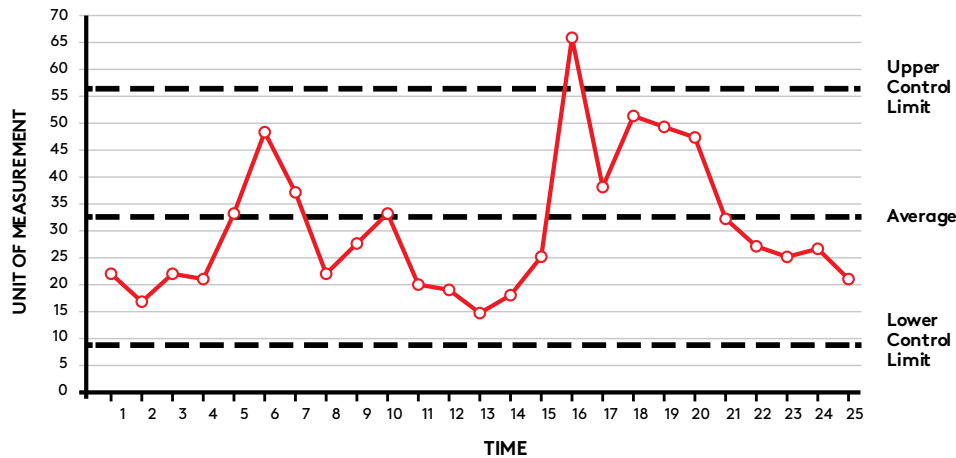


Figure 6. A Statistical Process Control Chart

Product development, software development and other types of projects that bring to life our ideas are not really so different. If you were to assume that you would complete as much work this week as you completed last week, you would probably not go too far wrong. This even works where there is variability as long as you are taking a sufficient slice of time to average this out. Suppose we have a difficult task that takes longer than expected – this might throw the data a little, reducing the number of task completed that week. But over a month, the team can probably expect to have at least one difficult piece of work to complete.

Thus as an average, the forecast of what we will complete in a given month is probably reasonably accurate if based upon what we completed last month.

Of course, this method only works if the team is already working steadily and has some historical data. If the team is new, or has recently changed composition, then the method will not work. At this point the only options are to use either industry- average data or the organisation's historical data. These are both far less accurate since the team and exact project work are both different.

This way of estimating has sometimes been called 'No Estimates'. Vasco Duarte, a speaker and writer on Agile, has championed the No Estimates movement. He simply suggests selecting the most valuable work, breaking it down into small chunks and then developing, iterating and refactoring each chunk of work. As data on the duration of tasks builds up a clearer picture emerges. Normally, if teams are breaking the work down into similar sized chunks, most items will take roughly the same amount of time. From this information, the team can predict how long it will take to complete a given number of task – all without estimating in advance.

The pros

Past data or 'No Estimates' is easy. It is fast and simple and completely transparent to everyone. It can save a great deal of time in arguing about estimation points, technical possibilities, ideal 'developer days', etc., because it simply assumes that everything will be the same as the previous iteration.

This means that we can eliminate some of the optimism bias so common in estimating. It insists we base our productivity, not on what we would like to achieve, but on what we actually achieved last time.

It reduces some of the politics that can make the process of estimating so painful, and in doing so saves even more time than by simply removing the hours spent estimating discrete tasks. There is no longer any need to debate whether a team is more productive or less productive, produces more or fewer defects, or takes more or less holiday than some mythical average – it is simply the same as it was last time.

The cons

Past data cannot be transferred from one team to another since that instantly removes most of the advantages described above. That means it does not scale across teams or across organisations. It is particularly dangerous if people wish to misuse the simplicity of the idea to make false, simplistic predictions such as – if the current team manages 30 requirements a month then if we create a second team we will deliver 60 requirements!

If your predictions are based on industry standard data or data from another team, then your forecast could be so broad in its range as to be meaningless for planning purposes.

It also means that the teams must measure and record their data honestly – lead time, amount produced (whether measured by features, requirements or lines of code), effort and number of defects.

There can be significant debate about how these things are measured. What units do you use for effort (hours or days)? What is included when counting up lines of code or stories? What about support or maintenance work? Is a change request the same as a defect? There are hundreds of potential disagreements simply around the data and these need to be considered to ensure that the measures are consistent so that the team is predicting based on like-for-like data.

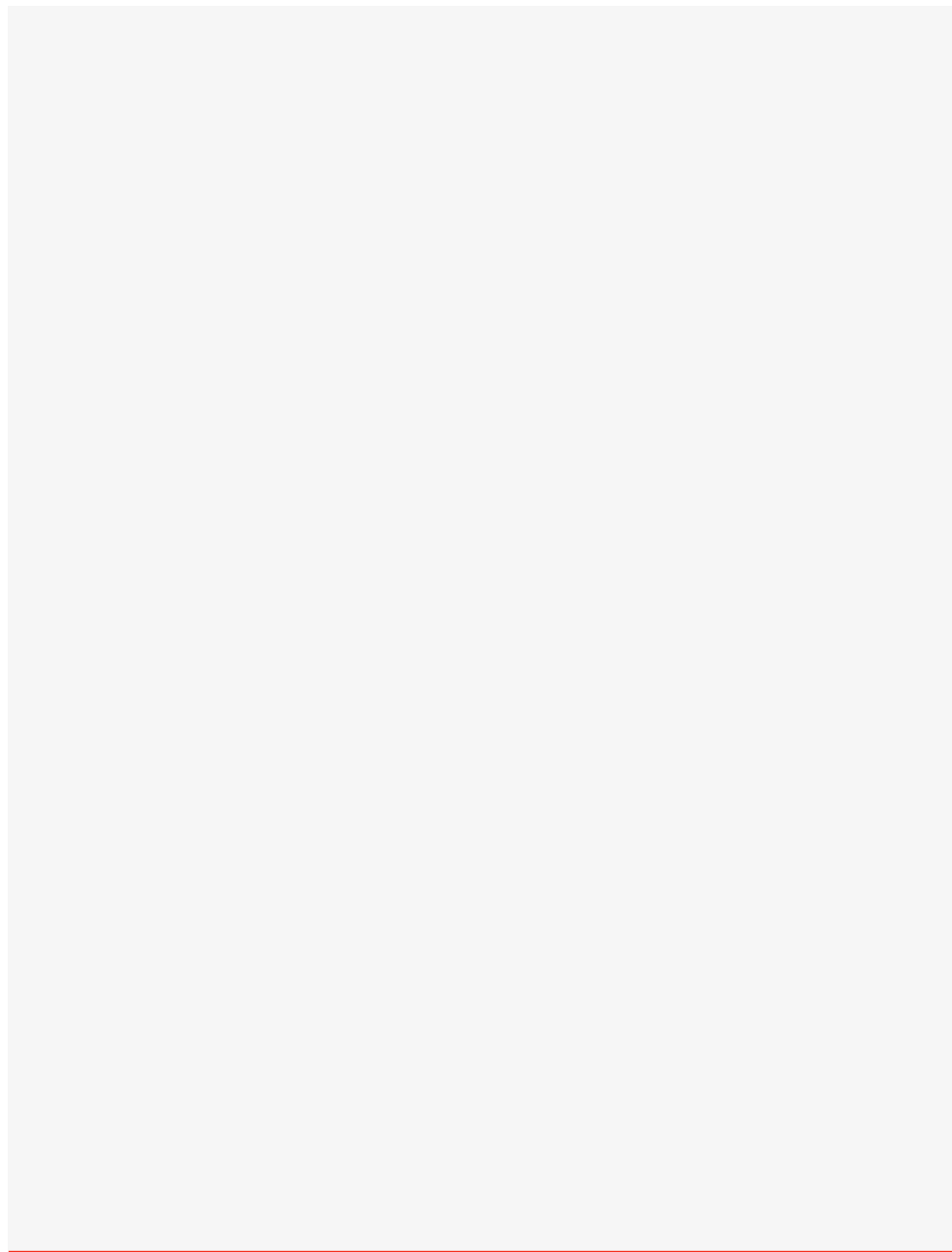
While past data is a helpful method for forecasting how much will be delivered by a certain date, it is not a good method for estimating the likely effort a project may require when little is known in advance. Being able to say that 'on average we write 400 lines of code a day with an average of x defects' may help with planning, but rarely with investment decisions when it may be highly uncertain whether a project is 4000 or 40000 lines of code.

Things required	Historical data recorded for a specific team on size, effort, time and defects. The forecast can be shown as a predicted velocity line on a burndown chart.
Best for	Probabilistic forecasting: given our current rate, how long until we can achieve x?

Activity 3: Look to the past

Using the data you collected in Activity 2, consider how many features were actually completed in the first month (or months) of development. Use this figure to predict forwards how long it will take the team to complete the project at the current rate of progress.

Is there a difference between this figure and the original prediction? If so, how many people are aware of it? Has the overall prediction changed, and if not, what measures are being taken to suggest there will be a reason that the team can increase their rate of development?



2.2. Estimation by comparison

How tall is the Eiffel Tower?

Which is taller, the Eiffel Tower or a giraffe?

People are not very good at creating accurate size estimates of something in isolation, but we tend to be very good at size comparisons. When you think about a task – do the washing up – you may not know how long it will take you, but when you see a sink-full of crockery you will be very good at saying whether it will take you more time or less time than washing up three coffee mugs.

We can harness this natural ability by asking ourselves to make comparative estimates. We can ask if requirement A is bigger or smaller than requirement B, or roughly the same size.

We can keep this really quick by assigning a simple relative estimate label: small, medium or big, for example. Some go a step further and use T-shirt sizes, extending the number of estimate labels to XS, S, M, L, XL, etc.

In 'proxy-based estimation' we go one step further. Here we turn the 'label' into a numeric proxy that equates to value or effort. The most usual is to assign each requirement a number of 'story points'. Story points are just a way of sizing something up – they do not equate to actual hours or days of work. Instead they refer to relative effort.

In this case the thought process would try to pick a medium-size requirement and assign it 5 points. Each other requirement would then be compared to this – does it feel a bit bigger? A 6? Or is it twice as much effort – a 10?

Finally, a popular method for doing estimates (whether relative or absolute) is to use planning poker. Each individual will usually be given cards with numbers in the particular sequence. A typical deck has cards showing the Fibonacci sequence including a zero: 0, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89; other decks use similar progressions. The reason for using the Fibonacci sequence is to reflect the inherent uncertainty in estimating larger items.

Several commercially available decks use the sequence: 0, ½, 1, 2, 3, 5, 8, 13, 20, 40, 100, and optionally a '?' (unsure) and a coffee cup (I need a break).



Figure 7. T-shirt size label



Figure 8. Planning poker cards

The Product Owner will briefly explain the idea behind a feature or user story and everyone will place a card face down. At the same time everyone turns the cards over. Any outliers – scores much higher or lower than the norm – will have a chance to put their case. After a short discussion everyone votes again. Often an outlying score exists because an individual has picked up on a point that might make the story particularly difficult, or it has exposed a misunderstanding that could have led to problems later.

Once we have this comparative data, we can sometimes then go on to calibrate and compute it, by comparing it with historical data. This may be a simple rule on the usual time taken to complete a small story, a medium story and a large story.

If using story points then it might be slightly more nuanced. The team will have a 'velocity', an average number of story points that they worked on in the last iteration or in a given time period. Thus they can predict how many of the stories they have just estimated they are likely to complete in the following iteration.

The pros

Relative estimates are usually quick to apply and developers often feel more comfortable using them than assigning absolute estimates which feel like commitments.

It can offer a more flexible understanding for Product Owners on the size of story. Where it is impossible to break a product into equally-sized small chunks it can be important for stakeholders to understand the impact of a story's size. This helps with prioritisation decisions as well as coordinating work between teams.

Because developers think more about the stories and their size the process of estimation can sometimes be part of exploring the idea and gaining important clarification in advance of beginning work actually creating code.

The cons

Although it is quick, comparative estimation still takes more time than no estimation. The meeting (usually a planning meeting) can be a lengthy one as people argue over whether a requirement is small or extra small. Often there is very little value in this discussion.

Relative labels, whether t-shirt sizes or story points, still do not necessarily help business stakeholders make the key investment decisions for which estimates are more important.

How small is small? How soon is soon? These are quite reasonable questions and in essence, by linking points or sizes to past data, teams are simply adopting the same solution as past data, but have done more work to get there.

The estimates are still open to the usual issues with optimism bias and the inevitable uncertainty. At the other extreme, teams using story points need to be careful that the labels are not taken to mean 'days of work' by stakeholders and then turn into a commitment.

By introducing proxy labels, we have abstracted our work away from real value and real time. This can lead to all sorts of business failures as we deliver lots of story points, none of which equate to actual cash, or we build up lengthy queues of 'effort' points without realising that this means delay.

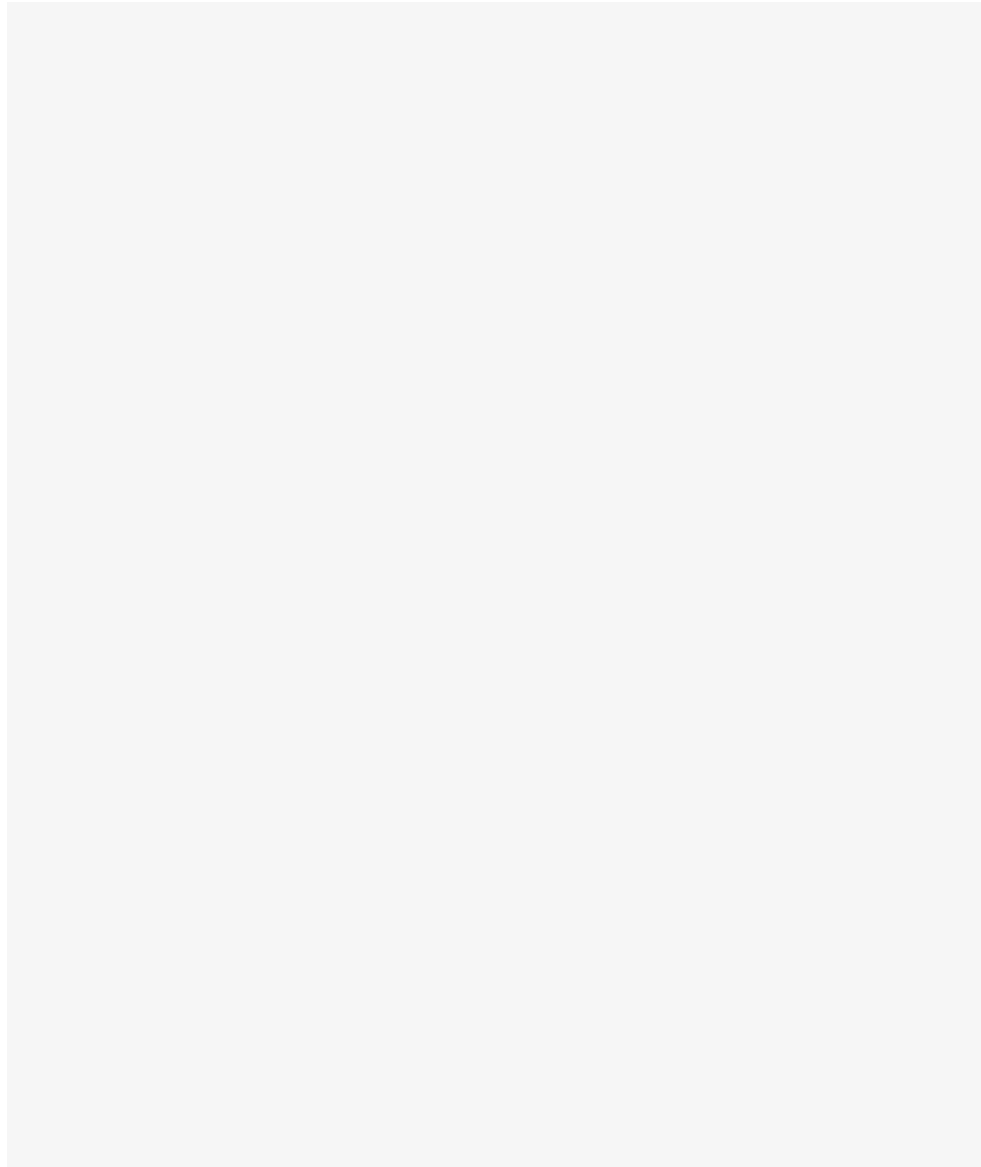
Things required	Developers need to have a level of expert judgement, but those estimating should be those who are doing the work themselves, not an 'estimation expert'. Everyone must understand the assumptions behind what constitutes a 'size' whether expressed as a label or number and apply this consistently.
Best for	Probabilistic forecasting with more understanding of the impact of variation in size of tasks or requirements.

Activity 4: Size it up

Take a random sample of features that you are planning to work on soon. As a team decide whether you prefer to use either t-shirt or story points to try to size them up. We would suggest using something different to your normal method so that you can compare its effectiveness.

The key point of the activity is to try and tease out differences in the team's estimates. You are looking for the outliers – where someone thinks a feature will be much larger or smaller than everyone else. The idea is to explore any differences of opinion over what a feature entails – less to create an accurate estimate than because this is an excellent way to explore design ambiguities.

Are there any features that felt too big to size? What would the team normally do about such stories?



2.3. Decomposition and recomposition

So far, none of the techniques have helped with the crucial investment decision – how much might this cost me and so should I invest in doing it?

To deal with this we tend to do something that's very natural to us. We break the idea down into chunks and ask how long each bit would take us to deliver. Then we add all the bits together. We might choose to be a bit more sophisticated and consider a best case, a most likely case and a worst case scenario for each chunk and then use a formula to try to adjust for our optimism bias and offer a summed 'expected case'.

This is the way we plan a journey. I don't know how long it will take to travel to Paris, but I could work out how long it will take to drive to the airport (depending on best case and worst case traffic scenarios) and I can look up flight times in advance and even calculate how likely my flight is to be delayed and what the average delay might be, using historical data. I can then add these together to get my final estimate. This shows the process of decomposition of tasks and recomposition of the results to reach the estimate.

Why does it work?

We've said that we find it hard to estimate large chunks, but small parts are easier – now let's look at what that does to error margins. For the purposes of illustration we shall use the example of the journey, but in reality, you would break a product down into features.

Example – How long will the journey from New York to Paris take?

Overall estimate

It took about 12 hours door-to-door that time we flew New York to San Francisco and Paris is about the same distance.

Overall estimate = 12 hours

Decomposition estimate

House to airport = 2 hours

Check-in, security, etc. = 4 hours

Flight = 7 hours 22 minutes

Deplaning, luggage collection, etc. = 2 hours

Taxi ride to hotel = 1 hour

Decomposition estimate = 16 hours 22 minutes

The actual journey took 17 hours 7 minutes.

The original estimate was out by 30%.

The decomposed estimate was out by only 4%.



Now let's look at the actual, decomposed results.

Element	Estimate	Actual	Error	Magnitude of relative error
House to airport	2 hours	1 hour 30 minutes	30 minutes less	25%
Check-in, security, etc.	4 hours	4 hours	0 hours	0%
Flight	7 hours 22 minutes	9 hours 22 minutes	2 hours more	21%
Deplaning, luggage collection, etc.	2 hours	1 hour	1 hour less	50%
Taxi ride to hotel	1 hour	1 hour 15 minutes	15 minutes more	20%
Total	16 hours 22 minutes	17 hours 7 minutes	4%	23.2%

This illustrates what's known as the 'Law of Large Numbers'. One estimate will always be either high or low, whereas when creating lots of little estimates, some are high and others were low. These tend to cancel each other out, providing the overall lower error margin, even though the average error % was about the same as for the single big estimate – 30%.

Absolute estimates provide a number of days that you think it would take to do a task. This is normally called 'ideal developer days'. It assumes that if no one interrupted the developer and he could focus on this and nothing else, then it should take this number of days.

The pros

Where we can actually work out the smaller elements of a piece of work, decomposition works well and allows us to offer a good idea of how much effort a piece of work might involve and thus make better decisions.

The cons

Massive up front decomposition to a level of granularity where we can truly be accurate in our estimation tends to be wasteful for all concerned. We can end up doing our design up front, nailing down scope in advance in pursuit of accuracy.

In any case we risk investing in our estimates without really buying ourselves greater certainty! Software projects are often victims of 'forgotten' features, the kind of unseen work that is described as 'non-functional requirements' or is part of releasing a product and that can add a sudden unexpected delay to a product.

While the reaction to this tends to be to plan more, in greater detail and resist change more firmly, this only exacerbates the problem.

Things required	Developers need to have a level of expert judgement and work alongside those who will be breaking an idea down into distinct parts.
Best for	Edge cases where difficult investment decisions require the best possible data available. To be used in conjunction with small bets.

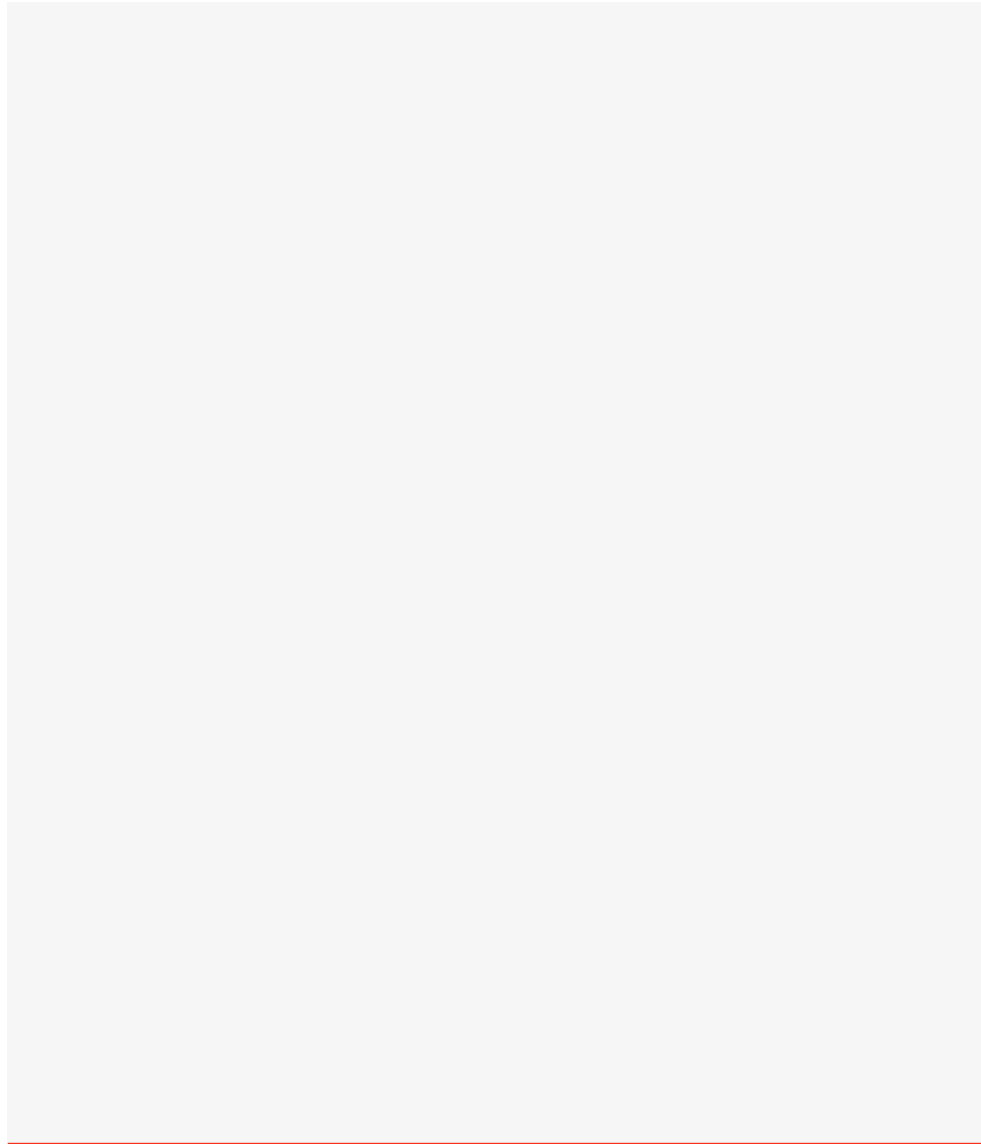
Activity 5: Small and big errors

With the same project used in the last activity, recognise that this activity needs to take place over an extended period of time – probably up to a month.

Take an iteration worth of work or another reasonably sized chunk – perhaps an epic.

Begin by making a guess for how long the whole piece of work will take and assign an absolute number of days to this. Next, break the work down into features and provide an absolute estimate for each of these.

Using a table similar to the one on the previous page begin to record what actually happens and how long every feature takes. Work out the error and order of magnitude. At the end of the iteration or epic, see which was more accurate – the big estimate, or the summed small estimates.



2.4. Wideband Delphi – the Wisdom of Crowds

Wideband Delphi is a process that was first introduced in the 1970s, by Barry Boehm and John Farquhar. It was considered 'wide' because it required a lot of interaction and communication between the people participating.

It has similarities to something known as the Wisdom of Crowds that was made popular by James Surowiecki in the book of the same name.

The book explains how, in days gone by, people could guess the weight of a cow by using the 'Wisdom of the Crowd'. The idea is that a diverse crowd will outperform the individual or expert predictions. The collective smarts will outperform an individual over the long term. Surowiecki suggests four factors to form a wise crowd – Diversity of Opinion, Independence, Decentralisation and Aggregation.

The example cited in the book is about guessing the weight of a cow. We might not know the average weight of a cow, but we do, typically, know our own weight and can extrapolate from there. This simple problem requires us to estimate by comparison individually, and then we can use the collective feedback to take averages. Because of the number of people, and diversity of viewpoint, the average will be close to the correct answer.

As the things we want to estimate become increasingly complex, or exist in a domain that fewer people understand, the crowd are required to have a level of expertise within the domain. Otherwise the guesses are just random. Surowiecki does argue that even in these examples, diversity of opinion is important. But others argue that the more complex the situation and the greater the need for creativity and innovation – the greater the need for expert domain knowledge. For instance, if you are planning on designing and building a new bridge between Copenhagen and Stockholm, it's probably only reasonable for other bridge builders to be involved in the estimation process, but it might be useful to have diverse construction engineers involved. There is no sense asking a Rocket Scientist to estimate, as their domain is very different.

The Wideband Delphi process relies on a similar 'Wisdom of the Crowd', which takes experts within a domain through a set of steps to help people estimate, discuss assumptions, understand variance and further refine estimates. The process is as follows:

1. A coordinator presents each expert with a specification and an estimation form.
2. They then call a group meeting in which experts discuss estimation issues with the coordinator and each other.
3. The experts fill out forms anonymously.
4. The coordinator prepares and distributes a summary of the estimates.
5. The coordinator calls a group meeting; specifically focusing on having the experts discuss points where their estimates vary widely.
6. The experts fill out forms, again anonymously, and steps 4 to 6 are iterated for as many rounds as appropriate.

In this way, the group can avoid too much groupthink because estimates remain anonymous but it still allows people to discuss their underlying assumptions and reasons for their estimates. Each iteration helps people learn more about the problem space and refine their understanding of potential solutions.

The pros

Wideband Delphi is great in environments where there are a number of experts who can debate ways of solving the problem. This can be used in conjunction with all other estimating techniques.

The cons

It works best in domains that have had similar problems in the past. It requires fairly deep expertise in the domain, and can go horribly wrong where we assume all crowds are useful and non-experts are asked to guess.

Wideband Delphi is not as useful in creative or innovative problem spaces.

Things required	A group of experts and time to run the process through a few iterations.
Best for	Deciding how to progress in complicated, but known domains. It is best for estimates that need optimising.

While estimates are an important part of many planning decisions and coordination activities, we need to be wary of spending too much time focused on how long something will take or how much it will cost compared to how valuable it is. While estimates on effort are often out by a factor of 10, estimates on value can be much further out.

Ultimately, we are looking to build up a picture of the world so we can make better decisions, invest our money wisely and create the most compelling products and services.

The biggest issue with all of the techniques so far is that people are extremely sceptical about estimates – the people who make them don't see the value and the people using them are normally just looking for a specific date.

Think back to the original reason why estimates are useful – to make good decisions. Consider whether the techniques so far have helped us to get the best ideas into the hands of our customers and users. Is there something we haven't considered?

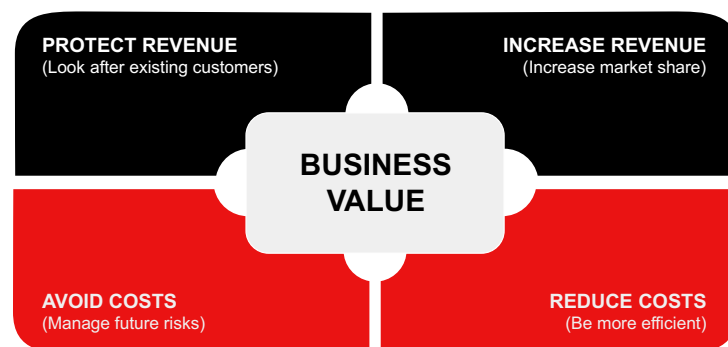
3 ESTIMATING VALUE

All of the estimating techniques we've discussed so far are most typically used to predict cost and effort. But what about value? Surely this is the most important thing to ascertain. If we can find the thing that is going to be most valuable for our business then how much time or money we spend on it becomes less important – that's not to say time and cost don't matter but value is most important.

3.1. Fast benefits assessment

A fast benefits assessment is a quick way of quantifying the business value of doing a particular piece of work. This helps provide comparable data to support informed and quick decisions about what to work on next.

Since 'value' can be notoriously complex to define, it helps to have a simple model that makes benefits easily comparable across work items. Below are the four 'buckets' described in the VFQ Technique Library item, each referring to a type of benefit. New features or work items have benefits, which fall in to one or more of the buckets.



1. **Increase revenue:** Doing the work would either improve profit margin, or increase market share. Examples are:
 - Sell more as a direct or indirect result of new functionality
 - Charge more as a result of new functionality
2. **Protect revenue:** Doing the work would sustain current market share or revenue figures. Examples are:
 - Keep customers through basic or 'maintenance' features
 - Delight customers with features that solve problems they didn't know they had

3. **Reduce costs:** Doing the work would reduce costs that are already being incurred. Examples are:
 - Cut down time needed to perform a task to either reduce or reassign employee time
 - Cut non head-count costs including overhead and equipment costs or decommissioning other applications
4. **Avoid costs:** Doing the work would avoid costs that are not already being incurred but may be soon. Examples are:
 - Additional head-count that might be needed
 - Fines that might be levied
 - Loss of reputation or brand damage

Once you have created your benefits model you can quickly assess the benefits of each new requirement. When a new requirement is raised in your process you can discuss the benefit type(s) and how much value is associated with those:

1. Find out why this requirement matters to the business. If it is not immediately obvious, keep asking 'why?' until you arrive at a point where you can identify one or more of the four benefit types.
2. For each of the benefit types you have identified, state the economic value as a figure (in dollars, pounds, etc.). There are a couple of tactics that can help you get to a figure:
 - Quantify the value of the benefits effects. Consider the example of a requirement to improve the accuracy and clarity of invoicing. The effects of doing this are that customers are more likely to pay the correct amount (protect revenue) without delays (increase revenue) and involve less employee time processing customer inquiries and complaints (reduce costs).
 - Set the value equal to the cost of alternatives. Consider the example of a requirement to automate a process. The value of automating this process is at least equal to the cost of doing it manually (plus the value of doing it faster and without human error).

If you don't have exact figures, then make reasonable assumptions.

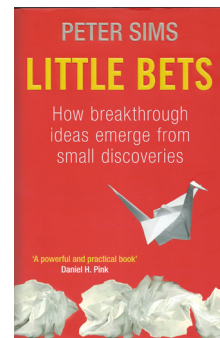
3. Keep a record of the benefits and any assumptions you have made alongside the requirement itself. The aim is to have the benefits (and their underlying assumptions) visible when you prioritise requirements.

3.2. Small bets

Rob Bowley, author of the blog post *Estimation Is At The Root Of Most Software Project Failures*, commented that, 'with software estimation you've only realistically got a choice of 5 minutes, 1 hour, 1-2 days, about a week, and then all bets are off.'

If you cannot reduce all features to these sized chunks, or if high levels of uncertainty mean that several problems are likely to lie in the 'all bets off' category, then decomposition will be a very time-hungry activity and may easily end up as waste. What do we do when we are not yet ready to dive in to the granularity of features, but we still need to make decisions about investing in a project or compare two ideas?

A different way of looking at estimates is to consider what we might invest, in order to learn more. Peter Sims in his book *Little Bets* describes this as risking small amounts in order to test out our assumptions. The idea is that rather than deciding on whether to fund a huge project and thus demand an accurate estimate for how long the whole thing might take us, we should fund just a small element. At this point it should either be small enough that we can decompose and create estimates, or sufficiently small that we can simply use past data and predict how soon we will finish without really needing to estimate at all.



Traditionally, the kind of data that helped make investment decisions came from market research or a willingness to buy other people studies. Today, organisations can make a real small bet by making a few of something or selling a beta version and then learning from the results. This is, really – simply another expression of delivering early and often. The difference is that, like Lean Startup, ideas for products are put in to the market as a way to make forecasts real and decisions grounded in better quality data.

In development terms, small bets often means creating a spike or a working prototype – something that will allow us to explore what is technically possible relatively cheaply. Once we have learned more about the possible solution, we can invest further effort and time in developing it.

CASE STUDY: Parrot cages

Figure 9. An unexpected gap in the market

Forward 3D was an internet media-advertising agency running global paid ad campaigns. Part of their expertise involved building the analysis tools that allowed them to understand billions of interactions and searches to profile demand. As part of that, the company decided to try an experiment. They looked for popular search terms that did not have much paid advertising on Google, reasoning that this would allow them to spot a gap in the market. The one they came up with was pet supplies in general, and in particular – parrot cages.

Since the idea of importing parrot cages was so alien to the company's existing expertise, they decided to test demand before taking any further steps. The company bought a few Google AdWords and set up

a simple e-commerce site through Shopify. The site – JustCages – went live. But there was no ability to actually buy – when anyone pressed on the 'Buy' button, a holding message flashed up. After two weeks, when it became clear that the demand was there. The company began ordering stock from vendors and shipping it directly to customers, meaning that while they paid a much higher price, they held no inventory and kept risk to a minimum.

Once the profitability was established, Forward 3D built their own, more sophisticated website, hired a warehouse and began to hold inventory. Later they expanded to operating ten niche outlets selling cages and aquariums. Eventually, the idea proved so successful that Forward 3D set up an entire business – Just Shops – offering over 4000 niche products, producing £3.2 million in sales.

The pros

Small bets help us make large up-front decisions by an incremental funding model and, combined with past data techniques, they can offer excellent predictability and an ability to safely discover and learn what we need to go further.

The cons

Finding a way to test underlying business assumptions is not always easy. As with Forward 3D, the process is often a creative one – finding unusual ways to measure demand. Where there is no historical data, or companies do not know how to go about something, the emphasis must be on ensuring that the company can afford to risk this money and that learning is prioritised as an outcome.

Things required	Many small, clear decision points. An incremental funding model and the ability to gather, record and use data for feedback.
Best for	Deciding on investment for products with high levels of uncertainty.

3.3. The value of variability

In our quest for estimates, forecasts and predictions, there is a danger that we focus too much on getting our estimates right. We can start optimising for predictability of estimate and not value or payoff. Often, this can mean removing sources of variability in process and ideas, and aiming for something safe that we really understand rather than something truly valuable and unique. The job of the upfront process should be to try and sift through the best ideas quickly, estimate where we need to and try to find the things that will make the biggest difference.

When a product starts producing 'hockey-stick' growth it means you've hit something great that works for your customers, and you've figured out a way to monetise it. What's interesting is that this could be down to just one feature, or it could be a whole new product.

Consider the story of the Short Message Service (SMS). In 2012, SMS celebrated its 20th Birthday. By that time it had 3.6 billion (yes, billion) active users. And, in 2013 it broke the \$150 billion in revenue per annum mark.

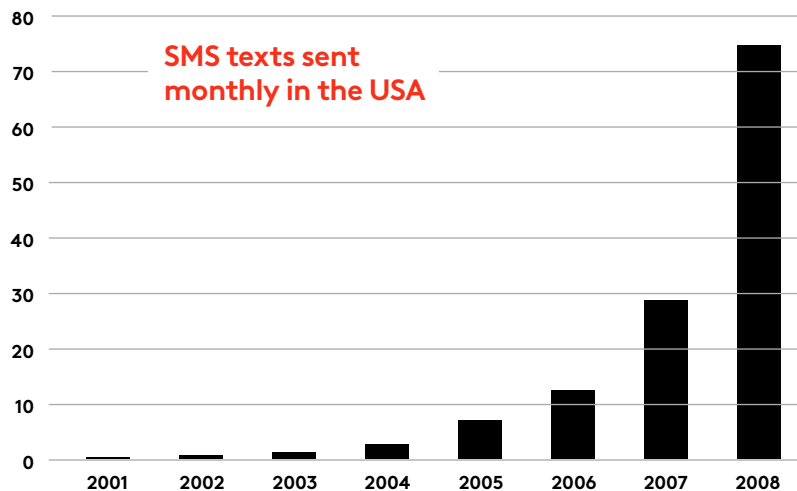


Figure 10. The hockey-stick growth of SMS texts in the USA

The idea was born in the 1980's centred around the idea of controlling the telephone network traffic using a small amount of data – 160 characters. The approach taken meant that every mobile station in the world could take advantage of the control mechanism by a simple software update. It was never intended to be the communication method of choice to share 'Happy New Year' messages with friends or to let them know you're running five minutes late and to go into the restaurant, you'll 'be there soon'. But that's what happened. It would be unfair to say that SMS alone would have been the triumph it was without mobile phones, networks and all of the other features that enabled its success. Nevertheless, that one relatively simple feature turned out to be a billion dollar idea – 150 billion by 2013 to be precise!

In his book *The Principles of Product Development Flow*, Don Reinertsen provides insight into how product development (and other creative processes) can exploit variability to increase our chances of delivering something extremely useful and valuable, but also allow us to effectively manage risk. Without explicitly stating, throughout this book we've skirted around the idea that estimates should really be associated with a confidence level, and any estimate will be subject to the laws of probability. How accurate we will be in terms of effort estimation, cost estimation and value estimation will follow a standard distribution.

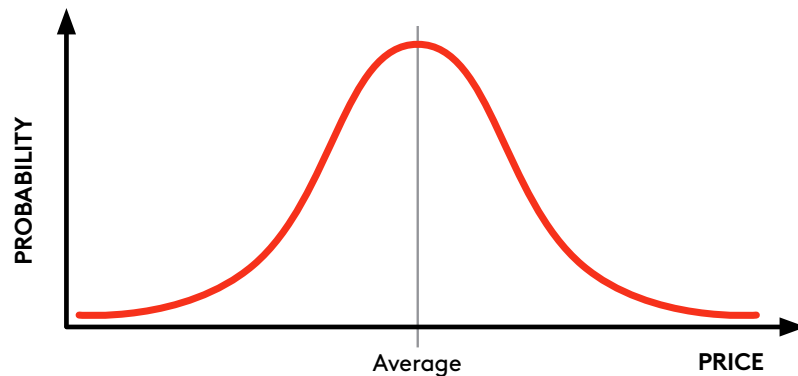


Figure 11. A standard distribution curve

Take a look at the following decision tree. To work out the Expected Monetary Value (EMV) for each path you multiply the payoff by the probability of success and then subtract the outcome of the failed route. So, to calculate the EMV for the first decision you multiply \$1,000,000 (payoff) by 0.5 (50% probability) and subtract \$100,000, which is the cost of the failure. This equals \$400,000.

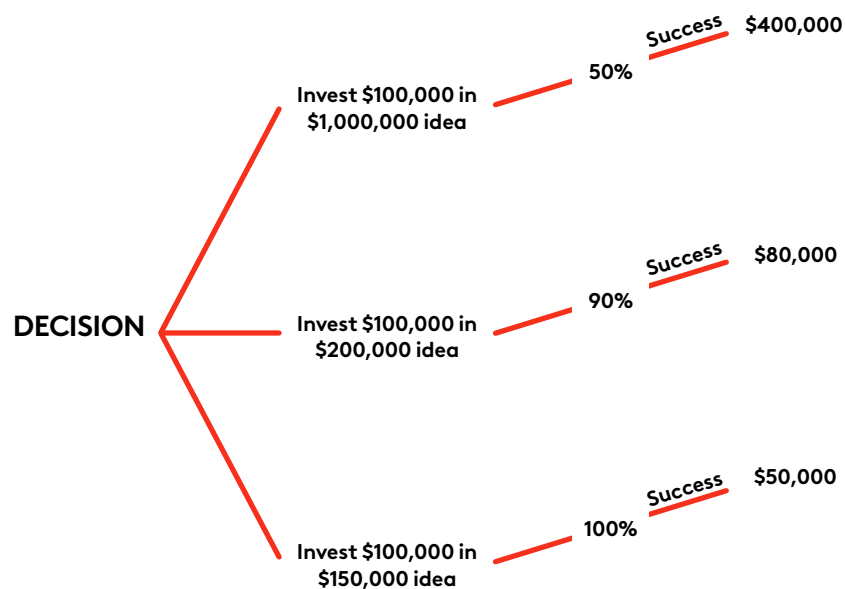


Figure 12. A decision tree

It is very normal to pick the safe, right decision. Nobody gets fired for picking the path that is 100% certain – regardless of the potential difference in payoff. But, we need to consider the most valuable potential outcomes, whilst controlling our risks. This conflict between certainty and increased value is something that needs to be handled and explored to see how we might manage it.

Product development, software development or any other development process is a series of choices and decisions. We don't have to take any single choice. As we discussed in the Planning session, Real Options is a process we can go through to consider when an option might expire. It is a common instrument used in investment strategies. Often, we focus on the likelihood of an event happening or not, but not on the potential value that might be created. We must consider the pay-off function.

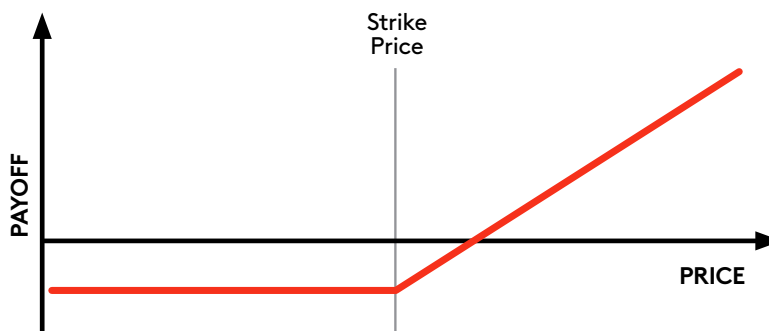


Figure 13. Asymmetric payoff function

In this, the owner of the option has the right, but not the obligation, to buy something at a pre-agreed price – this is known as the 'strike price' that may be exercised. As in Real Options, if the value at the time of call is not sufficient, you don't need to take it. If it is valuable then it makes economic sense.

The chart above shows how options have an ability to create unlimited upside. A situation where we have controlled risk and downside and a potentially unlimited upside is known as asymmetric.

As with all processes that are based on probabilities of success, there lies a distribution of success and failure to consider. When we combine the payoff function (Figure 13) with the probability (Figure 11), we get to a point where we can demonstrate that we can balance risk and reward.

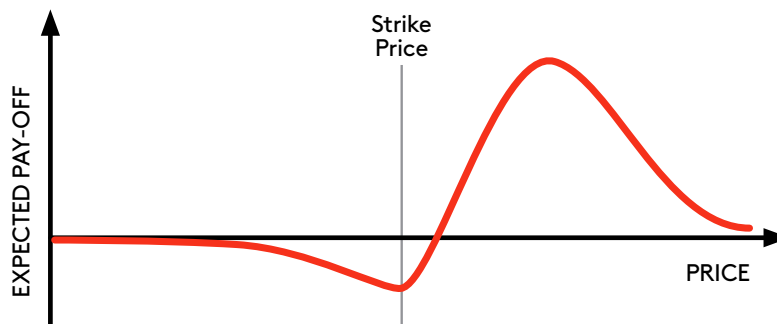


Figure 14. Combined payoff function with probability

Our job in planning and executing our portfolio of ideas is to balance risk and reward, and give us the ability to explore our ideas, gather feedback, and use controlled experiments and incremental delivery processes to learn about the market and our solutions. This gives us the greatest chance of finding gold – regardless of our upfront estimates.

All of this is to explain why a small bets approach remains the most important way of testing both business and technical assumptions in order to know when it is worth investing the up-front work of breaking down ideas, estimating them and creating an environment where we are most likely to significantly increase the output of value. Even if working without estimates and simply using probabilistic forecasting, it is essential that we build in decision points, delivering elements of end-to-end functionality where we can gain feedback and decide what to continue with, what to change and what to stop.

As Joshua Arnold explains on his blog blackswanfarming.com, you can tilt the playing field in development environments by combining smaller batches, a focus on value and a discovery mindset will help find the things that are truly valuable to you and your customers.

4

FORECASTING

Sales is a numbers game. That's a pretty standard response from most sales people. It might be true. But which numbers matter most? Is it the number of opportunities or leads? Is it the win rate? Is it the average time to close the deal? Or is it the average deal size?

According to the work of the TAS Group, there are four primary drivers. They use something called the Sales Velocity Equation to explain their rationale. It goes like this:

$$\# \text{ OPPORTUNITIES} \times \$ \text{ AVERAGE} \times \% \text{ WIN RATE}$$
$$\text{LENGTH OF SALES CYCLE}$$

Once you have information about each lever you can decide on where you need to focus, and you can also generate a lagging measure of performance that you can review periodically. The perceived wisdom is that the more opportunities you put in; the more likely you are to do well in sales. But the studies actually show that the top performing sales people have faster sales cycle, are good at managing the time to first contact and follow-up, and have higher deal sizes. The big question is: do they get that by focusing on those things or are they just good already and that's the result? Cause and effect aren't always immediately understandable. One thing made clear by the study though, is that it isn't quite the numbers game that most people would suggest. The impact of speed is the most powerful lever.

How do we know this? Because there is a lot of sales data in the world – much of it coming from platforms like Salesforce.com and new analytics companies like InsightSquared.

4.1. Pipelines

Sales are a good area to examine when thinking about forecasting. Most sales organisations will keep data about their pipeline, the probabilities of moving through the pipeline stages, win/loss ratios, average deal sizes, time-to-close and many more factors. In fact, project management and product managers could learn a lot about bringing their best ideas to customers using pipeline techniques. A sales process is exactly that – it's the lead-to-customer process. Development projects are about ideas-to-customers. They're very similar.

As we've learned throughout VFQ, tools like cumulative flow diagrams and burndown charts, and understanding how to examine queues, work-in-progress and the impact of batch size allows us to gather more data than we have traditionally been able to in projects. In a sense, the pipeline of projects, requirements, features, defects, work requests and many other work item types, can be treated in exactly the same way and with a little more sophistication than the typical VROM/ROM processes that are often found in project-based organisations.

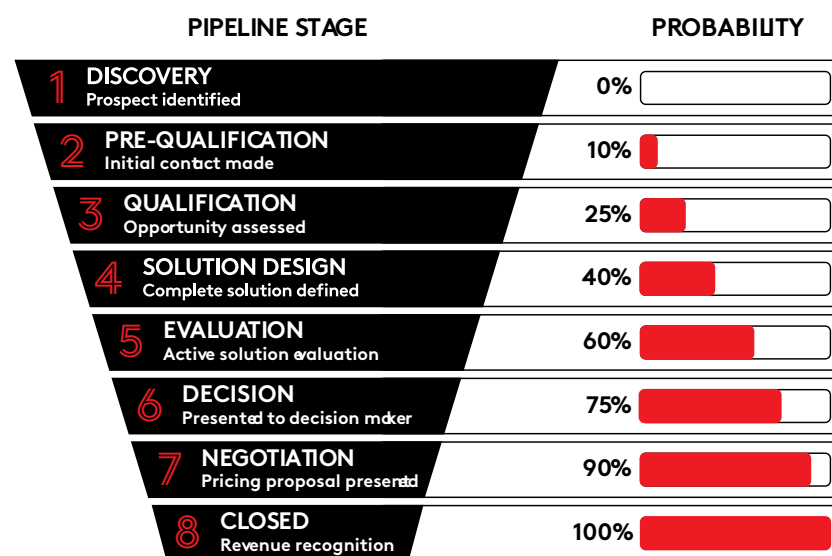


Figure 15. A typical sales process with percentages

4.2. Learning from history

In the estimating section we covered the idea of 'yesterday's weather' using past data. What's interesting in sales is that the aggregation of all the data from every team and sales person allows companies to forecast potential deals and income. This can be dangerous for a new person or team just starting out with a new company, as it will take some time to get up-to-speed. But, this can also be captured and accounted for.

The key thing in a forecast is to understand that statistics are at play. Knowing the likelihood of closing a deal that is only currently sat at 'pre-qualification' in your pipeline and has a probability of 10% means we can weight our pipeline. Also useful is knowing that for deals worth between \$10,000 and \$20,000, our first estimate of close dates are typically two weeks too aggressive.

In his book, *Software Estimation – Demystifying the Black Art*, Steve McConnell talks a lot about probability and likelihood. He argues that most of the time when we have single point estimates (i.e. we'll be done on the 12th December or this feature will cost us \$42,321) it is more likely that a target has been given than an estimate that has really been thought through. If it were an estimate you'd be more likely to see some sensitivity analysis or ranges associated with the forecast. A great estimate is likely to include the probability with it.

In their ebook *The Right Metrics For Your Inside Sales Team* InsightSquared discuss the key pieces of insight required to build up a solid sales forecast. These include items like:

- High probability opportunities ('strike zone')
- Pipeline integrity
- Conversion and win rates
- Metric-driven forecasts

These are all built up using probabilities and analysis. They all use data about how often dates are moved or estimates of value are changed. They look at actual closed dates, and how these relate to our original guesses. They also consider how these might compare to previous quarters, months or years, and how they look against averages. For instance, when examining pipeline integrity, these are the rules used to flag up 'red' items:

- An opportunity's close date has changed five or more times
- An opportunity has lingered in the same stage six times longer than an average deal
- An opportunity is three times larger than an average deal

These are assertions that might indicate a little more risk or reward, and a different approach to managing the situation.

All of this is only possible because data is captured, stored and then analysed. Step One is to capture things that are important and then the actual results. As with all data capture systems, you need to make this as easy as possible.

4.3. Building models

Forecasting is about creating models that will allow us, to some degree, manage and control the resources (time, energy and money) to the greatest affect.

One of the effective forecasting models we know of is for the weather. This is a model that looks at hundreds of variables, and takes into consideration past performance, and a lot of complex equations and statistics to see what we might expect in the near, medium and long-term future. These models get better and better with time, and computational power, and we have reached a stage where things can be fairly accurate. But, it hasn't always been this way.

In 1987, the UK was battered by storms that uprooted trees, demolished buildings and brought some areas to a standstill. It isn't the first time this has happened, and it won't be the last, but it was the most fierce hurricane to hit the UK since 1703. What makes this event stand out for most Brits though is what the Weather Forecaster, Michael Fish, said the night before the storm. He said, categorically, that the storm was 'not going to be too bad' and that there was 'nothing to worry about' – even though others had asked if they should worry. He used definitive language, which was unusual, and he created a sense of certainty that came back to bite him. It has been marked in weather reporting history as a blunder that gets rolled out whenever another storm is forecast.

What we do know is that with the more data we collect, and the creation of assertions and variables that impact our business, we can come up with improved forecasts.

How well are this year's sales progressing?

€3.1m this year versus €2.4m on average

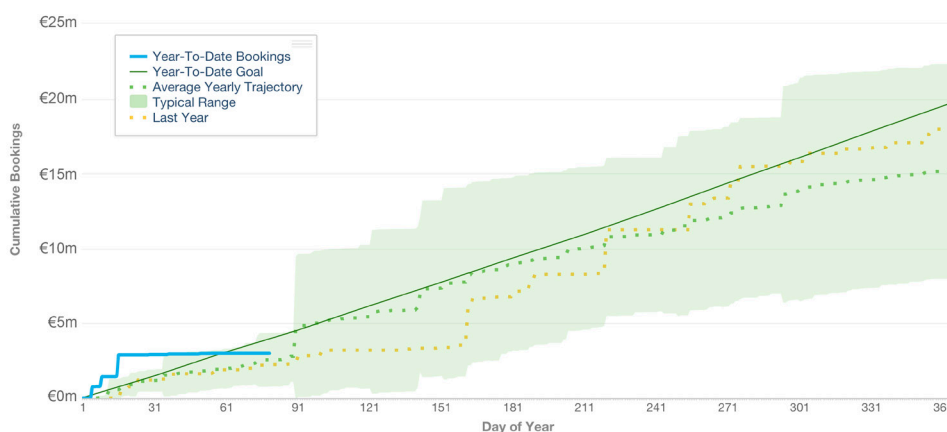


Figure 16. Sales booking trajectory example from InsightSquared

What's interesting about this is that it encapsulates all of the concepts so far – ranges, estimates, predictions, commitments, forecasts and comparison to the past. In the image above, the green shaded area shows the potential range of sales for the year. It clearly has a large degree of uncertainty. The green dashed line shows average yearly trajectory and the yellow dashed line shows last year. This is all judged against a new target (solid line). The actual performance is the blue line.

However great our understanding of past performance is, it doesn't mean our future performance will be predictable or accurate. But, over time models will help us better manage risks. As we discussed in the Planning session though, sometimes we'll be hit by black swan events and they might be good or bad. We just have to deal with the consequences.

Activity 6: Forecasting a value pipeline

A number of companies are starting to use the concept of pipelines to manage their 'idea to market' process. Basically, an idea is considered similar to a 'sales opportunity', but it represents a piece of work that needs to be developed. This idea might be a re-organisation, a new product, a change to an existing service, a piece of regulatory change or even a replacement for an existing IT system. All of these things are ideas we might invest our money in and have the ability to affect business today, tomorrow and into the future.

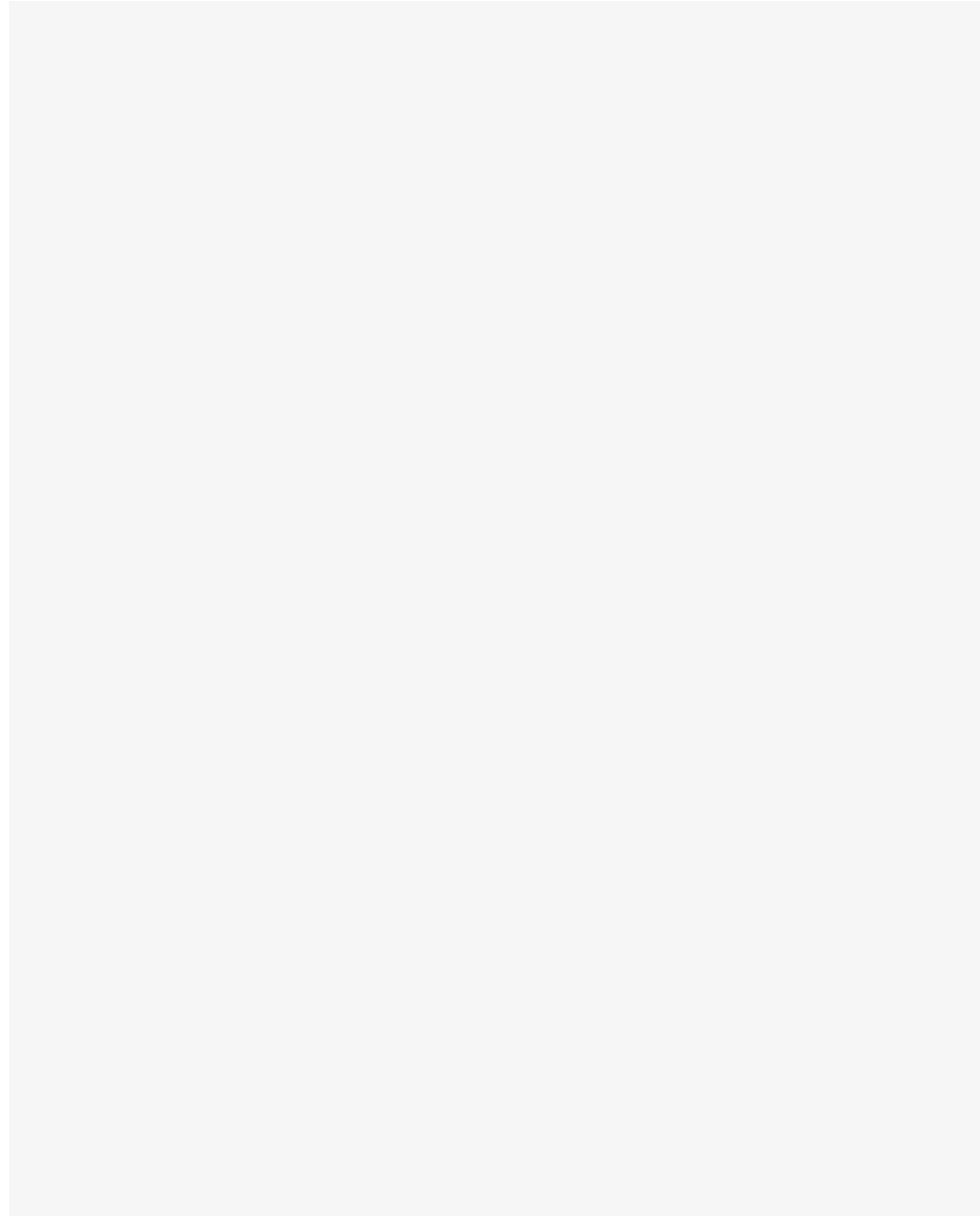
This activity is going to apply the concepts from a sales pipeline to our ideas. In projects, product and software delivery, requirements management, work management and project management tools are normally used. However, even if you plan and execute work using spreadsheets or cards on a wall, you should still be able to start applying the concepts over time. If, however, you use a specialist tool, you may find you already have some analytics, to start. Work will still be required though.

A typical sales opportunity system will capture:

- Customer (account, stakeholders)
- Opportunity name and description
- Value
- Stage
- Probability
- Value
- Project start date

Tasks:

1. Identify the information you have about your ideas/requirements/projects/initiatives. Are there any attributes from the list above that are not captured? Take your list and add the missing variables. If this takes some configuration in your tool, speak to the administrators and discuss ways of capturing the data.
2. Consider how best to collect the data given your current workflow. Tip: The easier and quicker this is, the more likely the data will be captured!
3. Explore in a group how long it might take to get insight for your team, department or organisation. When will you start learning about actuals versus estimates? Will your data collection approach help you spot changes to estimates or forecasts?
4. Come up with a team plan on how to continuously improve your forecasts – share these with your managers.

**Commentary:**

Many organisations struggle with the collection of good project data at the beginning. The value of this data will be seen over time and will continue to improve. This information not only helps build up a picture about a single opportunity at the beginning of its life – which can be used immediately. It can also help us judge the performance throughout its life compared to other items of a similar size, value or timeframe. Or from different teams, product areas or customer segments. Also, because it can be instrumented, assumptions and past history can be used to create better future forecasts.

5

CONCLUSION

Planning and estimating are topics that are inextricably linked. Figuring out what we're going to work on, where the value is and how long things might take are puzzles that plague every organisation. The problem is, many managers focus on the estimation problem a little longer than might strictly be valuable. Every moment we spend estimating is time away from actually developing amazing products or software, or implementing our great ideas. In the back of our minds we should keep thinking about the problem that is trying to be solved. If the goal is to ensure we are working on the things that matter most, then a focus on figuring out what is most valuable seems like the right thing to do. If we really need to give our customers a due date or we need to figure out how to launch our product by a certain time for market reasons then we need to consider time a little more. Our aim is to not only plan to build the right things and make good decisions at the beginning, but also to ensure we are continuing to make the best decisions going forward. Many of the techniques and concepts that are discussed in this session can be coupled to the planning model – things within the knowable and controllable domain probably need a small amount of estimation and forecasting to provide visibility of plans to a team. But when we find ourselves in more uncertain situations, having an approach that allows us to build something small, predict it, and learn and adapt, will provide us with the best ability to create solid forecasts and predictability. Most importantly – we will have the opportunity to obtain the largest returns on our investments.

Learning outcomes

Now that you have completed this session, you will be able to:

Describe why and where estimations and forecasts are useful

- Focus on estimating to figure out the value side of an equation – don't just focus on predicting cost
- Consider the value of the idea or feature before agreeing an amount of time and effort to put in to estimating
- Use iterative and incremental delivery to shrink the cone of uncertainty and validate estimates and forecasts

Understand and appreciate the strengths and weaknesses of different estimating techniques

- Use techniques that predict value as well as cost and effort

Explain why using crowds (groups) and data is useful in improving estimates

- An increased number of people with a diversity of viewpoints will make the average estimate closer to the reality
- Crucially, when estimates are required in a complex domain, the individuals involved must have a level of expertise in that area

Design an approach to improve your ability to forecast future value throughput

- Assess the business value of a particular piece of work using fast benefits analysis
- Trial ideas using small bets testing assumptions before deciding to fund on a large scale
- The value of variability and asymmetric payoffs

Create strategies for collecting, identifying and providing key value and effort information into the planning and execution processes

- Use the Sales Velocity Equation to decide where to focus sales and evaluate sales performance

BIBLIOGRAPHY

- Alleman, G.**, 2009. Plan The Work – Work The Plan. Herding Cats, [blog 11 September]. Available at: <http://herdingcats.typepad.com/my_weblog/2009/09/plan-the-work-work-the-plan.html>. [Accessed 12 November 2012].
- Appelo, J.**, 2010. Management 3.0: Leading Agile Developers, Developing Agile Leaders. Addison Wesley.
- Arnold, J.**, 2014. Product Development Payoff Asymmetry. [online] Available at: <<http://blackswanfarming.com/blog/>>. [Accessed 6 March 2015]
- Bowley, R.**, 2011. Estimation Is At The Root Of Most Software Project Failures. [blog 21 September]. Available at: <<http://blog.robbowley.net/2011/09/21/estimation-is-at-the-root-of-most-software-project-failures/>>. [Accessed 6 March 2015].
- Cohn, M.**, 2005. Agile Estimating and Planning. Prentice Hall.
- Daly, D.**, 2010. There Are 4 Sales Velocity Levers. Are You Pulling Them All? [online] Available at: <<http://www.thetasgroup.com/donal-daly-blog/there-are-4-sales-velocity-levers-are-you-pulling-them-all>>. [Accessed 4 March 2015].
- DeMarco, T., Lister, T.**, 2003. Waltzing With Bears: Managing Risk On Software Projects. Dorset House Publishing.
- Fowler, M.**, 2005. The New Methodology, Predictive Versus Adaptive. [online] Available at: <<http://martinfowler.com/articles/newMethodology.html#PredictiveVersusAdaptive>>. [Accessed 24 October 2013].
- Harvard Business Review**, 2007. A Leader's Framework For Decision Making. [online] Available at: <<http://hbr.org/2007/11/a-leaders-framework-for-decision-making/ar/1>>. [Accessed 12 November 2012].
- Harvard Business Review**, 1984. How Senior Managers Think. [online] Available at: <<http://hbr.org/1984/11/how-senior-managers-think/ar/1>>. [Accessed 13 November 2012].
- Harvard Business Review**, 2012. The True Measures Of Success. [online] Available at: <<http://hbr.org/2012/10/the-true-measures-of-success/ar/1>>. [Accessed 9 November 2012].
- Harvard Business Review**, 1999. Turning Goals Into Results: The Power Of Catalytic Mechanisms. [online] Available at: <<http://hbr.org/1999/07/turning-goals-into-results-the-power-of-catalytic-mechanisms/ar/1>>. [Accessed 24 October 2013].
- Highsmith, J.**, 2009. Agile Project Management: Creative Innovative Products. 2nd Edition. Addison Wesley.
- Hubbard, W. D.**, 2010. How To Measure Anything: Finding The Value Of Intangibles In Business. 2nd Edition. John Wiley & Sons.

InfoQ, 2013. Interview With Eduardo Miranda About Estimating And Planning Agile Projects. [online] Available at: <<http://www.infoq.com/articles/eduardo-miranda-estimating-planning-agile>>. [Accessed 24 October 2013].

InfoQ, 2013. Martin Fowler at GOTO Amsterdam 2013 about Agile Essence and Fluency. [online] Available at: <<http://www.infoq.com/news/2013/06/martin-fowler-agile-essence>>. [Accessed 24 October 2013].

InsightSquared. The Right Metrics For Your Inside Sales Team. [ebook] <http://www.insightsquared.com/wp-content/uploads/2013/11/The_Right_Metrics_for_your_Inside_Sales_Team_v8.pdf>. [Accessed 4 March 2015].

Kelly, A., 2011. Three Plans For Agile. [online] Available at: <http://www.allankelly.net/static/writing/webonly/ThreePlansForAgile_long.pdf>. [Accessed 24 October 2013].

Langr, J., Ottinger, T., 2011. Agile In A Flash: Speed-Learning Agile Software Development. Pragmatic Bookshelf.

Loch, C., DeMeyer, A., Pich, M., 2011. Managing the Unknown: A New Approach to Managing High Uncertainty and Risk in Projects. Wiley.

McConnell, S., 2006. Software Estimation, Demystifying The Black Art. Microsoft Press.

North, D., 2009. The Perils Of Estimation. Dan North & Associates. [blog 1 July]. Available at: <<http://dannorth.net/2009/07/01/the-perils-of-estimation/>>. [Accessed 18 October 2012].

Perkin, N., 2011. Agile Planning (Redux). Only Dead Fish. [blog 14 July] Available at: <http://neilperkin.typepad.com/only_dead_fish/2011/07/agile-planning-redux.html>. [Accessed 24 October 2013].

Pichler, R., 2013. Agile Product Planning: Vision, Strategy, and Tactics. Pichler Consulting [blog] 15 April, Available at: <<http://www.romanpichler.com/blog/product-planning/agile-product-planning-vision-strategy-tactics>>. [Accessed 24 October 2013].

Project Management Institute, 2009. A Guide To The Project Management Body Of Knowledge: PMBOK Guide. 4th Edition. Project Management Institute.

Reinertsen, D., 2009. The Principles of Product Development Flow. Celeritas Publishing.

Sims, P., 2012. Little Bets: How Breakthrough Ideas Emerge From Small Discoveries. Random House Business.

Smith, P., Reinertsen, D., 1997. Developing Products In Half The Time: New Rules, New Tools. 2nd Edition. John Wiley & Sons.

Solon, O., 2010. Ad Firm Becomes Largest UK Parrot Cage Importer. [online] Available at: <<http://www.wired.co.uk/news/archive/2010-07/23/forward3d-paid-search-parrot-cages>>. [Accessed 6 March 2015].

Surowiecki, J., 2005. The Wisdom Of Crowds. Reprint Edition. Anchor.

The Telegraph, 2012. The Great Storm: Michael Fish's 1987 Hurricane Forecast Blooper. [online] Available at: <<http://www.telegraph.co.uk/news/weather/9608533/The-Great-Storm-Michael-Fishs-1987-hurricane-forecast-blooper.htm>>. [Accessed 5 March 2015]

Wheelwright, S., Clark, K., 1992. Revolutionizing Product Development: Quantum Leaps In Speed, Efficiency And Quality. Free Press.

Vasco, D., 2013. A Quick Trip To The Future. [online] Available at: <<http://www.slideshare.net/duartevasco/a-quick-trip-to-the-future-land-of-no-estimates>>. [Accessed 24 October 2013].

Wikipedia. Hofstadter's Law. [online] Available at: <http://en.wikipedia.org/wiki/Hofstadter%27s_law>. [Accessed 7 August 2013].

Wikipedia. Planning Fallacy. [online] Available at: <http://en.wikipedia.org/wiki/Planning_fallacy>. [Accessed 7 August 2013].

Wikipedia. Short Message Service. [online] Available at: <http://en.wikipedia.org/wiki/Short_Message_Service>. [Accessed 6 March 2015]

APPENDIX

Activity 1: Answers - How good is your estimation?

Question	Answer
How many moons are there within our solar system?	Ironically, no one really agrees on an answer because no one can agree on how large a rock orbiting a planet needs to be before it is called a moon! The answer is thus anywhere between 174 and 240.
What is the average life expectancy of a man in Peru?	75
How many books are there in the New Testament of the Bible?	27
What % of the ocean is salt?	3.5%
How many people live in Timbuktu?	54,453 according to the 2009 census
When did the last person who spoke Cornish as their only language die?	1676
How many copies of Mao Zedong's Little Red Book have been sold worldwide?	900 million
How many train stations are there in India?	7,083
How many miles of coastline are there in Alaska?	5,580
What is the average number of correct answers in an estimation quiz?	Average number of correct answers is 2.8

valueflowquality.com

emergn.com