Reliability-Cost Trade-off and Wrong Problem Risk in Requirements Engineering

Ivan Jureta

Fonds de la Recherche Scientifique - FNRS

and Namur Digital Institute, PReCISE Research Center, University of Namur

ivan.jureta@unamur.be

http://ivanjureta.com

https://www.draw.io/#G1HI1BlwwlRHbSJcKvVW2jSwVkAitrtcZs

Abstract

Introduction

Information that you get from a requirements specification is an approximation of information which may be relevant to you when investing effort to understand opportunities and problems to address, and how to design new or change existing systems to do so.

What does it mean that it is an approximation? Requirements exist when there is a need to make a change in present conditions, including thereby presently available systems. Requirements describe desirable properties of a future in which new systems (i.e., those not available at present) are assumed to be up and running.

Therefore, a requirement *involves a prediction of future preferences and of a future environment.*

Let me clarify. If there is a requirement that the system should allow every user to consult data generated through her own use of the system, then this indicates the preference for enabling this, over doing something else, i.e., not doing so. Moreover, there is no system today, so this is

a preference assumed to apply in the future, when that system-to-be is expected to be up and running. It also follows that a requirement involves a prediction of some part of the world in which that system is up and running, and in which those preferences will be held. By "part of the world", I mean that we are going to predict only some things about the world in the future, which are somehow related to the system-to-be.

Importantly, observe that we cannot know today if users (or more broadly, stakeholders) will in fact have this preference in the future. Even if they tell you today that they will have it, they themselves cannot predict the future and cannot know if they will in fact have that preference. They can change their mind in the meantime. This fits results in research on preference prediction, namely prospection research in psychology [GW2007], and projection bias in economics [LOR2003].

The next best thing you can have today, then, is a prediction. You have an assumption about what could happen in the future. Ideally, you would know, i.e., have an empirically validated assumption today, of those future preferences. Since that is elusive, the next best thing to have is an approximation. It is not even clear what it means to validate future preferences today. As I said above, they can change.

Future *is* the future, so to speak, precisely because it has not happened yet. You can therefore only predict it, and you can only do so imperfectly. Worse still, there is no perfect information about the present, so it is illusory to believe that you can have perfect information about the future.

What is missing in these approximations, for them to be perfect information? That is, what is the difference between the approximations *you can have*, and the information *you would ideally have*?

This is really a question of what can be said about unknowns of a specification, of that which remains unaccounted for by a representation of requirements.

This is an interesting question. A requirements specification is a central output of Requirements Engineering [Z1997,ZJ1997,JMF2008]. Having an idea of that which can be missed, is an input

to understanding and living with the risk of dealing with approximations. At the same time, it is an opportunity to consider what we are approximating in the first place, and to what extent we even can access that perfect information (and if it might exist at all - in which case approximations are not approximations, i.e., are not *by definition worse* than something else).

I hope to convince that, besides being interesting, it is also a relevant question. Part of the contribution is the argument that being informed about what may be absent, in general from a specification, can lead to different conclusions from that specification.

Outline

I develop the following argument.

1. I start from a simple premise, that different categories of raw material (including, e.g., elicitation results, parts of specifications, diagrammatic models, simulations, prototypes, etc.), which you make and buy during Requirements Engineering, have different cost and reliability.
2. I distinguish four categories of raw material, M1 to M4, roughly corresponding to data, information, intentional states, and knowledge. I define these categories here so that they are different in terms of cost and reliability.
3. I use reliability to establish an order of preference of M4 through to M1, M4 being more reliable than M3, M3 than M2, and M2 than M1.
4. I use cost to establish another order, of M2 being costlier than M1, M3 to M2, and M4 to M3. That cost order is due to the cost for transforming one to the next, i.e., nothing to M1, M1 to M2, and so on.
5. While I may prefer the specification to convey M4 alone, I argue that this is in general not feasible. Cost is absorbed only as much as available resources and contextual factors allow. Time and cognitive capacity [M1978,GG1996,K2003], for example, are types of limited resources. Contextual factors, such as the sheer unavailability of some data, make it impossible, even if resources increased substantially, to access all relevant knowledge.

6. The above leads me to conclude that a specification can give only an approximation of perfect information. But what is perfect information? I provide a definition of perfect information from a specification, and thus of a perfect representation of requirements. I then argue that such a thing is only an ungrounded abstraction, and that its instances are elusive in practice. Importantly, it is not an ungrounded abstraction because it has an unbearable cost, but because its instances simply cannot exist.
7. Since perfect information is elusive, I argue that when doing Requirements Engineering, you must continually decide between committing more resources to moving from M1 to M4, roughly, from data to knowledge, or not doing so, but instead absorbing a risk that you are wrong, since you decided against investing to move to a more reliable specification. In other words, I highlight that there is a tradeoff that comes from there being a preference for lower cost, which makes M1 preferred to M2, M2 to M3, etc., while there is the preference for relevance for M4 to M3, M3 to M2, etc.
8. Finally, I argue that this Reliability-Cost Trade-Off is resolved continuously during Requirements Engineering, and highlight that its resolution leads to an inevitable accumulation of risk, of solving wrong requirements. This risk is called the Wrong Problem Risk in this paper.

Reliability Cost Premise

The argument developed in this paper starts from this premise: *Not all raw material made and bought in Requirements Engineering is equally reliable or equally costly*. I call this the Reliability Cost Premise (RCP) in the rest of the paper.

Here, reliability means the extent to which you would trust a requirement to accurately reflect actual future preferences. Given a requirement that you read from a specification, reliability is the extent to which you would consider that that requirement reflects preferences which you predict will be true in the future. Note that reliability asks you for two evaluations of a specification. One, it asks you to evaluate the fitness of the prediction conveyed by the specification to your prediction of future preferences. Two, you need to actually have those predictions of your own.

As a side note, I consider that prospections [GW2007] are a kind of prediction, and I see no reason to distinguish them from predictions in general; in any case, I do not see how that distinction would affect the argument presented in this paper.

I say that some raw material is costly, in that it takes, at the very least, time to acquire or make, and then use it to make decisions during Requirements Engineering.

It is trivial to observe that raw material acquisition and production have cost. Even if that raw material may in fact involve representations and thoughts, rather than concrete, metal, glass, and so on. Try, for example, to produce a specification of a system such as a government information portal on tax norms. It will take you some time to get to know who the target audience is, what they may expect such a system to help them with, how they expect this to happen, why they may use it, what would make them ignore it, and so on. Then, you would need, following the usual methodology narrative in Requirements Engineering [DVLF1993,CKM2002], to identify and reduce imprecision, vagueness, ambiguity, in that information, from there understand which requirements the system-to-be should satisfy, represent these requirements, understand their interactions, so as to ensure coherence and consistency. And so on.

The point, in this example, is that this takes time. If you make or buy systems, such time translates to cost, if only of time of people dedicated to work on design and analysis. Other costs exist as well. You commit to a specific set of requirements, invest into the engineering and release of a system to satisfy those, as opposed to others. That is an opportunity cost.

Raw Material Categories

Given that there is cost, I will avoid the traditional language of Requirements Engineering, where one gathers, gets, finds, or obtains that raw material through elicitation [GL1993,ZC2005]. It is important for the argument developed here, to keep in mind that all of these verbs involve the passage of time. Since that time has a cost, I will be saying that I *buy*

data, rather than *get* it. Even if I am the one doing the effort which produces that material, and thus perhaps I am not paying myself, that is simply an accounting matter, which does not alleviate that there is cost.

So what, then, are we buying and making during Requirements Engineering? There are different ways to categorize that, and naming will depend on the properties you want to highlight.

Following the Reliability Cost Premise, I am interested in categories which allow me to highlight two kinds of differences, those of reliability and cost. Given that reliability has to do with trust and accuracy, categories need to differ in how they relate to these. If I reasonably assume that judging accuracy involves interpretation of data, and that judging trustworthiness involves evaluation of available evidence, I need categories which differ in the role of interpretation and evidence. I also want a minimal set of categories which allows me to continue this discussion, rather than a comprehensive account of either interpretation and evidence.

Four categories are enough. They are called M1, M2, M3, and M4. Naming is intentionally such that I want to avoid immediate associations with notoriously plurisemantic notions of data [R2017], information [A2013], intentional states [S2015,R2016], and knowledge [IM2017], despite similarities that I discuss below.

M1

M1 is any *explicit* and *shareable representation*.

Explicit means that it is recorded on something other than one's mind. Text and drawings on paper, paint on cavas, sound recording in a file, on vinyl, tape, etc., images recorded on a memory card, are examples of explicit representations. This includes requirements models, specifications, software code, execution logs, databases, and so on.

A representation is shareable if it can be accessed as-is by someone other than you. It matters that it is shareable as-is. Both you and I can access, roughly speaking, the same file if you email it to me. Evidently, I may be accessing a copy, and so we are in fact not accessing the same instance, but uniqueness does not matter here. Sameness and sharing do. Since this text is focused on costs, note that sharing involves a cost too.

Sharing requires that the representation is explicit. Communication by voice, for example, produces representations of what the speaker is trying to convey. Spoken word is also an explicit representation, accessible to others (provided their presence, and so on).

Sharing matters, because it is necessary for communication about that which is being shared. Once there is communication, we can collaborate on the content of that communication. Sharing and collaboration will be important as I get to M3 and M4 later.

M1 covers, then, two categories. All existing explicit representations you get your hands on, but did not make yourself, are one such category. That category includes all representations you purchased. All representations that you make are in the other category. This includes changes to representations you bought.

For example, when specifying requirements for a freight transportation marketplace, a report on freight rates was purchased. The report, an electronic document, was made by a consultancy specializing in analysis of historical freight transportation rates. The report itself is an instance of M1.

Note the *make versus buy* distinction. This is more important than it may look. Suppose that you are investing in a new business which aims to facilitate clinical workflows in cardiology, specifically focusing on the diagnosis and treatment of hypertension. The team there has to choose if they will do research on current clinical workflows, or if they will rely on a subcontractor to do it. In the former case, they will make M1 themselves. In the latter case, they are buying it. If it is true that there is more information one can have, than information one can render explicit, i.e., we know more than we can communicate, then choosing make over buy in this example has important consequences. Specifically, it means that anything that is known by the subcontractor, but remains outside the specification, will also remain outside the

organization which purchased M1. As I will elaborate below, M1 is an approximation of M2, and while M2 may have remained in the organization if the choice were to make, it will remain outside, with the subcontractor, if the choice was to buy. As a side note, make vs buy is directly related to broader research on transaction costs in economics [W1991,WW1984,KZ1992].

In the rest of the paper, I will write M1 to refer to the class, and m1 to some generic instance thereof. To distinguish two instances, I will name instances by any combination of letters and numbers, and predicate them by the class they instantiate. For example, M1(a) and M1(b) are two different instances of M1. I do not define parthood over M1 instances, as this makes no difference to the discussion in this paper. More generally, I do not propose a language over Mx instances, as I have no use of it here. Importantly, this means that when I write M1(a), it can represent something complicated in practice, such as an entire specification document.

In guise of a summary on M1, remember that all which is available to you and can be available as-is to anyone else is called M1, and that any self-contained unit thereof is a result of a buy vs make decision.

M2

I said in the Introduction that you *get information from* a specification rather than that *information is in* the specification. Since a specification is explicit and shareable, it falls under M1.

M2 is what you understood from accessing M1.

For some instance of M1, the corresponding M2 is your interpretation of M1. Interpretation happens in the mind, so that M2 instances themselves cannot be explicit representations, and cannot be shareable.

Since reading some a|M1(a), gives you some b|M2(b), the notational convention hereafter is to write a/b to capture this relationship.

All that can be shared is M1. It follows that any effort you invest to share your M2, is not going to result in the sharing of that M2, but of representations you make of M2. You will be sharing instances of M1, which you made to convey those instances of M2 that you intended to convey.

An important implication of differences introduced so far, is that your M2, or more specifically all that you have and that falls under M2, must be a function of M1.

Would M2, then, be a function only of M1? Let us assume that it is, that some instance m2 of M2 is a function only of some instance m1 of M1.

When would M2 be only a function of M1? If for you, interpretation equals inference as in formal logic, and your brain were to operate exactly as an inference engine from those rules, and if you could somehow isolate that m1 from everything else you may hold in mind when making inferences, i.e., producing instances of M2, then m2 would be a function of m1, and since inference rules apply to all M1 instances in this case, m2 is a function only of m1.

The problem with this is that all you would have in M2 are logical consequences of M1, so logical consequences of all that you observe.

It seems, however, that creative conclusions are not necessarily logical conclusions, in the sense of being reached through deduction. Creativity, in other words, cannot fully be accounted for by inference rules; otherwise we would have been able to create creative expert systems. To the best of my knowledge, this has not happened.

M2 is not, then, the outcome of some known rule-based transformation of M1. To some extent, at least, M2 depends on M1, but also on such things as attention and memory. Attention, in the sense that in drawing conclusions from some m1, I will only use a subset of M2 instances I can recall. And memory limitations mean that I can recall only some of the M2 instances I had ever held in mind before the present. Attention and memory research shows no established rules required at the level of abstraction needed here [K1973,P1980,PP1990].

There are several important conclusions from the discussion so far:

1. M2 instances cannot be defined as a function only of M1 instances.

2. When you communicate your M2 to others, all they get are M1 instances with which you approximated your M2. These produce their M2 instances when they pass through their cognitive apparatus.
3. You and I will get different M2 instances from same M1 instances. You and I can read differently the same specification. Your M2 is not my M2.
4. You and I cannot agree on M2, but only on some M1. This is not odd at all; contracts, for example, are under M1, and courts are there to enforce your and my agreement on that M1, especially when we disagree as a result of differences of M2.
5. If there is a difference between your M2 instance and mine, both from the same m1, and if we need to jointly commit to m1 as, e.g., a specification of the system-to-be, then we have a cost to agree on m1. We have to bear that cost before we can make that commitment. This is the cost of getting to an agreement that the representation conveys what you and I both need it to convey. In Requirements Engineering, activities performed to reach that agreement are typically called requirement validation.
6. There are three kinds of costs generated by the difference between M1 and M2: cost to access some m1 in order to be able to form m2, cost to interpret m1 and thereby form an m2, cost to agree on some m1 given that My m2 from that m1 is different than your m2 from that same m1.

Keep the notation in mind as we move forward: m2/m1 is for your m2 made from accessing and interpreting m1. A way to think about this notation is that, when you read it backwards, it gives you the origin of the leftmost item. I further want to emphasize that m2 is yours and not mine, so I will write (m2.me)/m1 for my m2 from m1, and (m2.you)/m1.

M3

What can you do with some m1? If you can access it, you can either leave it be, or access it. If you do access it, e.g., you read some part of the specification, this amounts - in the language of this paper - that you are investing effort which produces change in your M2 instances, i.e., you get some (m2.you)/m1.

What do you do, then, with these M2 instances? If we take, for the sake of simplicity, folk psychology as our theory of mind, then your M2 affect your actions through being the content of your intentional states. For example, it will be different if you believe that m2 fits the world (i.e., is true), or if it is not and yet you desire it to be. If you desire it, you may commit to act, to change current conditions so that m2 is no longer only desired, but is the case; this depends on other beliefs and desires you have, other intentional states.

To affect behavior, M2 instances become content of intentional states, which gives us M3 instances.

The key difference between M2 and M3 is that I was non-committal on the veracity of M2, and truth enters the picture in M3. That generates a cost, because committing to some m2/m1 being true involves at the very least some thinking about it being plausible, and more strongly, and costly, empirical validation. Observe that thinking thoughts that you have not empirically validated is clearly different than those you have validated. The former is, again in terms of cost, more modest. I return to this later with M4.

I will take some m3 to be some m2, combined with a truth confidence gamble in the following sense. I may have read in the specification this sentence, an m1: “The system should automatically produce a price estimate of an item that a user of that system posts for sale.” This led me to m2 that the system, once up and running, will be computing a price estimate for each item someone, who uses that system, puts on sale to others on and via that system. Do I think that that will happen (i.e., that the system will in fact do it, and that users will in fact post items for sale, and so on), before I see the system being in fact used? Before I see it, I can only make a prediction that it will happen. Based on what I learn about user behavior, system’s environment, and so on, my confidence in that prediction will be changing. Perhaps I am overconfident early on, and as we do more research on user behavior, I conclude my overconfidence is no longer justified. Or I find out that I was completely wrong, leaving me without confidence in that statement.

Point being that m3 from that m2 is a six-tuple consisting of

1. that m2, which I got from being triggered to think, when accessing some m1,
2. a truth value, conveying the fitness of m2 to my experience of the world;

3. confidence, described by odds, for a bet I would take take that that m2 will have and hold that truth value during a time period,
4. validity of the truth value, which is a potentially unbounded time period during which I assign the truth value to m2,
5. validity of odds, which is also a potentially unbounded time period during which I accept to take the bet,
6. a preference statement, which says if the given truth value for m2 is more desirable to another truth value for m2, or a truth value of some other M2 instance, or if I am indifferent to the truth value of m2.

So I may have accepted a bet yesterday that the system will be used in the way mentioned above, after its first release, but I may no longer accept it today, i.e., I would only accept a different bet on it. Clearly, going from m2 to m3 requires some thinking, so time and effort. Notice that this also suggests that not all in M2 is content of all in M3. I can read (access) something without forming an opinion on its truth. I just read on bbc.com that a Chinese taxi driver, a single mother, takes her toddler every night with her when driving. Reading it did not lead me to form the bet on its truth. But I certainly can invest more thinking to get to that bet.

These odds and bets may seem like an unnecessary complication. I argue, however, that they are needed to convey a subtle dynamic in teams involved in the system design and engineering. Each participant will hold their own prediction of the future, which in turn means that they will make judgments, even if they keep them their own, about the relevance of requirements and corresponding solutions, that is, their fitness with their prediction about how and why the system will be successful or fail once it is up and running. This will affect confidence in the design and engineering decisions made, and can in turn affect their motivation. The odds above are there to emphasize the presence of such judgments, thus a need to keep in mind that there is much more about requirements than the specification can convey. In practical terms, it may be interesting to invest in eliciting odds that team members have, and use this as a signal of the divergence of predictions. I leave those concerns, of which questions a specification should answer, outside the scope of this paper.

What costs are there to go from m2 to m3? You need to choose a truth value, choose odds, choose time periods or conditions for validity, and preference.

More notational conventions are needed for M3. Firstly, I write m3/m2 to say that m3 is about an m2. Intentional states being personal and non-transferrable, I write (m3/m2.me) to say that my m3 comes from my m2. The other five pieces of an m3 are referred to as follows:

- (about.m3/m2.me) for the m2 that m3 is about, shorthand being (m3/m2.me) as m2 is there anyway;
- (truth_value.m3/m2.me) for the truth value;
- (truth_value_confidence.m3/m2.me) for my confidence in the truth value;
- (truth_value_validity.m3/m2.me) for the time period or conditions that need to hold for me to keep my confidence in the truth value;
- (

7. a truth value, conveying the fitness of m2 to my experience of the world;
8. confidence, described by odds, for a bet I would take take that that m2 will have and hold that truth value during a time period,
9. validity of the truth value, which is a potentially unbounded time period during which I assign the truth value to m2,
10. validity of odds, which is also a potentially unbounded time period during which I accept to take the bet,
11. a preference statement, which says if the given truth value for m2 is more desirable to another truth value for m2, or a truth value of some other M2 instance, or if I am indifferent to the truth value of m2.

M4

M4 is M3 whose truth value is obtained through a procedure that others can reproduce.

To go from some m3 to m4, I need a procedure which allows me to establish the truth value. Moreover, that procedure needs to be reproducible by others, so that they can establish the truth value same way I did. The resulting value may differ, though, and there is no need here that M4 instances always reproduce the initially observed truth value. It follows that the procedure needs to be represented as an m1, i.e., needs to be explicit and shareable.

At the same time, however, an m4 cannot be about an m3 or an m2. Because there must be a procedure for reproducibly determining truth, an m4 can only be about an m1. Indeed, M1 are explicit and shareable instances, while those of M2 and M3 are not.

Costs related to M4 are the effort that goes into designing and applying the procedure, and creating a representation of that procedure so that it can be communicated to others.

An m4 is a tuple consisting of

1. m3, which is the intentional state whose M2 content that I want to reproducibly validate the truth of,
2. a truth value, produced through the application of the validation procedure,
3. confidence, described by odds, for a bet I would take take that the next application of the validation procedure will yield the same truth value,
4. validity of the truth value, which is a potentially unbounded time period during which I assign the truth value to m2,
5. validity of odds, which is also a potentially unbounded time period during which I accept to take the bet,
6. representation of the validation procedure, which was applied to obtain the truth value.

Validation does not mean that the resulting truth value is stable. In some laboratory experiments, for some specific statements, reproduction may yield so many times the same truth value, that confidence increases substantially. In practical settings, especially those discussed here, confidence might grow. But is unlikely to ever get close to certainty.

If m1 is the statement “users will post items for sale on the system-to-be”, perhaps they do indeed for a time, and as competition intensifies and they have more choice, confidence in that m1 might start to go down.

The critical implication is that it is important to monitor validity, and therefore repeat validation, update confidence, review the validation procedure when it fails, and so on. All that has a cost. In three of the most recent ventures I was involved in, that validation amounted to having a sample of target users, who initially participated in giving feedback on early passive visual designs, then testing prototypes to give feedback, then, after the initial release, used private

pre-releases of product upgrades, again to provide feedback. In early ventures, more than a decade ago, our practice was different and involved less cost but more risk; we would have no target user feedback before we were far into engineering. Initial system releases at that time were emotionally charged events. We would put a command in, it would let the product live and push ads to high traffic locations,as we watched users get in, and track them to see how many stayed, how long, and what they did. Those initial hours, we would be waiting to see how our requirements and design bets, made far in the past, pay off, if at all. Often, as perhaps in lottery draws, we did not win.

Relationships and Rules

Relationships over the four categories, and rules over these relationships have been mentioned across the text so far. I summarize them as follows.

- Every m1 is a result of an effort to produce that representation in such a format that it can be shared as-is. Going from nothing to an m1 involves a cost, thus every m1 has a cost property, even if its value may be unavailable.
- Every m2 is the result of reading, or otherwise accessing M1 instances, and I used the term interpretation to refer to access and its consequences.
- Going from some M1 instances to an m2 involves an access cost, interpretation cost, and agreement cost.
- Every m2 belongs to a person, to capture that m2's are results of someone's interpretation of explicit and shareable representations.
- Every m3 enriches an m2 with the truth value, confidence in that truth value, validity of the truth value, validity of odds, and a preference statement.
- Each of the parts of m3 can be represented by M1 instances, another way of saying that I can make a representation so as to convey my intentional states. I can make an m1 to represent the m2 that m3 is about, write another m1a to represent the truth value, another m1b to say that m1a is the truth value of m1, and so on, to represent m3.
- Every m4 has six parts mentioned earlier. And the same applies as for m3: there is a relationship between that m4 and its m1, and I can make an m1c to represent the truth

value that m4 assigns to m1, an m1d to relate that truth value to m1, and so on, to represent each part of m4, and represent relationships between them.

Categories, relationships and rules together provide a sketch of an ontology of raw material manipulated in Requirements Engineering. The ontology is designed, as I mentioned earlier, to emphasize differences of relevance and cost. If you are interested in other differences, then, as the discussion continues below, try to think about how these types of raw material can be transposed to an existing requirements ontology. For instance, if your inclination is to call some requirements goals and others tasks, then are all goals in M1? Can some be in M4? I return to this later.

The discussion so far suggests also that a specification can represent more or less of the raw material that was made and bought in the process of making that specification. If there are no m1's which convey truth values, confidence in those truth values, validation procedures, and so on, then that specification does not represent M3 and M4 instances, even if they presumably do exist. More on this in the next two sections.

Notational Conventions

To simplify writing below, I will write m1/m3 when I want to emphasize that m1 represents some (part of an) m3. For example, that m1 may represent the truth value, confidence, validity of confidence odds, and so on. When it matters which part this is, I will write m1/m3/confidence for confidence, for example. This also means that if m1a is a representation of some m3, and that m3 is about some m2, and that m2 come from accessing some m1b, then I will write this as m1/m3/m2/m1b. Same notational convention goes for any m4.

Moreover, I will write M1/M3 for all instances of M1 which represent parts of instances of M3, and will do the same for M4.

Finally, I will write M0 for all M1 instances which are not about parts of M3 or M4.

Reliability Preference

Which would you prefer to have: a specification which has only representations of information, neither of intentional states, nor of reproducibly verifiable assumptions? This is the same as asking if you prefer a specification which is made of M0 only, neither M1/M3, nor M1/M4. Would you prefer to have a representation of a verifiable assumption about some m1a, as opposed to a corresponding non-validated assumption of that m1a? That is, do you prefer, all else being equal, m1b/m4/m2/m1a to m1c/m3/m2/m1a?

Suppose that m0 is part of a specification, and that it is this statement: “user should not have to authenticate to view the balance of all bank accounts associated to their name”. Perhaps this was added by someone who did not think much about security implications, or is simply an error due to something else. But if you do not access it, it remains only an m0 for you, while at the same time it remains part of the specification. You avoid access and interpretation costs, but in fact remain ignorant that it is in the specification in the first place. Consequently, that m0, perhaps together with other parts of the specification that you did not access, are necessarily of lower reliability, *relative* to something thay you did access from that specification. If you accessed it, you at least did some interpretation of it, and it led to an m2. Suppose that you know how authentication should work for that system-to-be. Specifically, you already hold a firm belief (which would be an M3 instance) that authentication does allow one to view account balances, provided she had logged in so recently that her current session has not expired yet. One conclusion you might draw from accessing m0, thus moving it to m2, is a new m3b, a belief that that m0 was imprecise, in that it fails to say under which conditions that requirement applies, one being that user should not need to authenticate *again*, if already authenticated and her request to view balances is made within some specific time window after that last authentication.

The point, in short, is that reliability increases as you move from m0, to m2, to m3, and m4.

A disclaimer is in order. Evidently, it should matter if you did the validation instead of it being done by another. Especially if the stakes are high, and if you have skin in the game. I do not want to complicate this discussion further by these two considerations, stakes and skin, as they look orthogonal to the criteria I am interested in so far, namely cost and relevance; adding those dimension to the mix remains future work. With that caveat in mind, we move forward.

Cost Preference

As I argued so far, moving from m0 all the way to m4 means investing to cover a series of costs. Access and interpretation costs when going from m0 to m2, several costs to form intentions about m2 and thereby get an m3, and even more to design and apply a reproducible validation procedure.

Since these costs are due, at the very least, to the passage of time, and time on its own is scarce by definition, and is further restricted by deadlines and finite budgets, preference should normally go from from less costly to more.

The Cost Preference, then, is a preference order which makes m4 less desirable than m3, m3 less than m2, and a non-accessed m0 the best of all as it means you bore no costs for it. Clearly, if all we wanted was to minimize these costs, regardless of outcomes of pursuing that minimum, there would be no need to make the specification in the first place.

Perfect Specification

Given the discussion so far, what would be a perfect specification? What would you want to have as an ideal specification, as a result of doing Requirements Engineering?

If the perfect specification includes only the most reliable content, then it would be one which disregards cost. If, instead, it is perfect in minimizing cost to increase reliability, and reliability did

not matter at all, that perfect specification would be no specification at all, i.e., the perfect specification would be empty. Both of these are unrealistic.

If, instead, the perfect specification is a specification with perfect information, as that term is used in economics, it would be a specification which includes only and all true statements about the future. That is elusive too, since there are no validated truths about the future, only predictions.

This leads me to the point mentioned in the Introduction, that all specifications are going to be approximations of perfect information. Now, however, you should see that that was incorrect - there is no point approximating something that does not exist.

Instead, let us assume that, just as good scientists would do, that we want to have representations of validated ideas, and that we value more those validations which are reproducible. If a good, rather than perfect specification is one which has more m1/m4's than anything else, then at least one dimension of its quality is measured along the reliability preference order.

Reliability-Cost Trade-off

The problem, again, with increasing reliability is that it increases cost. If someone else adds some m0 to a specification, your cost to reproducibly validate it may involve asking them for a validation procedure, in which case you are assuming m4/m1. In the other case, when there is no reproducible validation procedure, you need to go from that m0 to your own m4.

At the very least, then, each time you access or update the specification, you make a decision that balances reliability and cost. You decided if that remains an m0, if you access it and get m2, and so on.

Wrong Problem Risk

Each time you decide against increasing reliability, you implicitly favor cost, but you also accumulate risk that the reliability of the specification is such that, if the system does get designed accordingly, it will satisfy the wrong requirements. It will be solving the wrong problem.

The extreme case would be one where you are aware that there is a specification, but do not access it at all, and where you put no procedures in place to have it accessed by other stakeholders, and certainly no means to perform reproducible validation of the specification's content. Though this sounds irrational, I consider it to be one of major reasons that led three ventures that I had access to, to fold, after substantial investment in engineering, based on an overconfidence in specifications that had low reliabiliy.

Discussion

Substantial effort is being invested, in both software engineering and requirements engineering, and for decades now, in the creation of representation languages and methods, used to decide and describe what systems are expected to do, and how they should do it, as well as into techniques for the analysis of those representations to ensure they satisfy some generic desirable properties (e.g., logical consistency). This the problem of what a blueprint for a software system looks like, and how to make good blueprints, both of which are far from being resolved. This includes work on, for example, formal specification languages, temporal logics, visual languages, checklists, best practices, software and requirements engineering methods, and so on.

Apart from Boehm's work on software engineering economics [B1981], very little has been done on understanding and conceptualizing the economics of software design. This paper is a small step towards that aim, focusing specifically on a notion of specification reliability, cost of that reliability, and risks that ensue from incremental resolutions of the reliability-cost trade-off.

Conclusions

This paper presents the argument that there exists a so-called Wrong Problem Risk whenever doing Requirements Engineering, and that that risk accumulates as decisions in favor of cost over investment in reliability are made. The argument uses a categorization of raw material made and purchased during Requirements Engineering. The aim of the paper is to highlight this particular kind of risk, so that further work can be done on measuring it, and informing decision-making in presence of the reliability-cost trade-off.

References

[GW2007] Gilbert, Daniel T., and Timothy D. Wilson. "Prospection: Experiencing the future." Science 317.5843 (2007): 1351-1354.

[LOR2003] Loewenstein, George, Ted O'Donoghue, and Matthew Rabin. "Projection bias in predicting future utility." The Quarterly Journal of Economics 118.4 (2003): 1209-1248.

[Z1997] Zave, Pamela. "Classification of research efforts in requirements engineering." ACM Computing Surveys (CSUR)29.4 (1997): 315-321.

[ZJ1997] Zave, Pamela, and Michael Jackson. "Four dark corners of requirements engineering." ACM transactions on Software Engineering and Methodology (TOSEM) 6.1 (1997): 1-30.

[JMF2008] Jureta, Ivan, John Mylopoulos, and Stephane Faulkner. "Revisiting the core ontology and problem in requirements engineering." International Requirements Engineering, 2008. RE'08. 16th IEEE. IEEE, 2008.

[M1978] March, James G. "Bounded rationality, ambiguity, and the engineering of choice." The Bell Journal of Economics (1978): 587-608.

[GG1996] Gigerenzer, Gerd, and Daniel G. Goldstein. "Reasoning the fast and frugal way: models of bounded rationality." Psychological review 103.4 (1996): 650.

[K2003] Kahneman, Daniel. "Maps of bounded rationality: Psychology for behavioral economics." The American economic review 93.5 (2003): 1449-1475.

[DVLF1993] Dardenne, Anne, Axel Van Lamsweerde, and Stephen Fickas. "Goal-directed requirements acquisition." Science of computer programming 20.1-2 (1993): 3-50.

[CKM2002] Castro, Jaelson, Manuel Kolp, and John Mylopoulos. "Towards requirements-driven information systems engineering: the Tropos project." Information systems 27.6 (2002): 365-389.

[GL1993] Goguen, Joseph A., and Charlotte Linde. "Techniques for requirements elicitation." Requirements Engineering, 1993., Proceedings of IEEE International Symposium on. IEEE, 1993.

[ZC2005] Zowghi, Didar, and Chad Coulin. "Requirements elicitation: A survey of techniques, approaches, and tools." Engineering and managing software requirements. Springer Berlin Heidelberg, 2005. 19-46.

[R2017] Romeijn, Jan-Willem, "Philosophy of Statistics", The Stanford Encyclopedia of Philosophy (Spring 2017 Edition), Edward N. Zalta (ed.), URL = <https://plato.stanford.edu/archives/spr2017/entries/statistics/>.

[A2013] Adriaans, Pieter, "Information", The Stanford Encyclopedia of Philosophy (Fall 2013 Edition), Edward N. Zalta (ed.), URL = <https://plato.stanford.edu/archives/fall2013/entries/information/>.

[S2015] Setiya, Kieran, "Intention", The Stanford Encyclopedia of Philosophy (Summer 2015 Edition), Edward N. Zalta (ed.), URL = <https://plato.stanford.edu/archives/sum2015/entries/intention/>.

[R2016] Ravenscroft, Ian, "Folk Psychology as a Theory", The Stanford Encyclopedia of Philosophy (Fall 2016 Edition), Edward N. Zalta (ed.), URL = <https://plato.stanford.edu/archives/fall2016/entries/folkpsych-theory/>.

[IM2017] Ichikawa, Jonathan Jenkins and Steup, Matthias, "The Analysis of Knowledge", The Stanford Encyclopedia of Philosophy (Fall 2017 Edition), Edward N. Zalta (ed.), URL = <https://plato.stanford.edu/archives/fall2017/entries/knowledge-analysis/>.

[W1991] Williamson, Oliver E. "Comparative economic organization: The analysis of discrete structural alternatives." Administrative science quarterly (1991): 269-296.

[WW1984] Walker, Gordon, and David Weber. "A transaction cost approach to make-or-buy decisions." Administrative science quarterly (1984): 373-391.

[KZ1992] Kogut, Bruce, and Udo Zander. "Knowledge of the firm, combinative capabilities, and the replication of technology." Organization science 3.3 (1992): 383-397.

[K1973] Kahneman, Daniel. Attention and effort. Vol. 1063. Englewood Cliffs, NJ: Prentice-Hall, 1973.

[P1980] Posner, Michael I. "Orienting of attention." Quarterly journal of experimental psychology 32.1 (1980): 3-25.

[PP1990] Posner, Michael I., and Steven E. Petersen. "The attention system of the human brain." Annual review of neuroscience 13.1 (1990): 25-42.
APA

[B1981] Boehm, Barry W. Software engineering economics. Vol. 197. Englewood Cliffs (NJ): Prentice-hall, 1981.