Dealing with Information Ambiguity

Lecture 40, v03

John G. Artus

BSEE

MSSE

INCOSE ESEP

Reducing Information Ambiguity

- One of the principle roles of the system architect is to reduce the amount of information ambiguity in the system specification inputs
- Ambiguity can arise from many sources, but comes about mainly from upstream technical and programmatic processes leading to a set of data that informs the development of the system architecture
- The system architect reduces this ambiguity by
  - Defining the boundaries for the development of the system architecture
  - Defining the goals of the system architecture development process
  - Defining the functions of the system
  - Etc
- The architect creates boundaries and concretizes goals by
  - Interpreting corporate and functional strategies
  - Interpreting competitive marketing analyses
  - Listening to users, beneficiaries, customers, or their representatives
  - Considering the competence of the enterprise and its extended supply chain
  - Considering the operations and operational environment of the system
  - Infusing technology where appropriate
  - Interpreting regulatory and legal influences
  - Recommending standards, frameworks, and best practice
  - Developing goals for the system based on the upstream influences

Depending on the maturity of the organization, some of these ambiguity-reducing activities may already have been performed prior to reaching the architecture definition process

Information Ambiguity

- Information ambiguity is composed of two ideas
  - Information Fuzziness
    - Information fuzziness occurs when information is subject to multiple interpretations
    - Example: interpretation of the color or texture of an object
    - Example: what is “good” gas mileage?
    - Fuzziness can also be influenced by cultural context
    - Stakeholder need statements are often rife with such information fuzziness
  - Information Uncertainty
    - Uncertainty occurs when an event’s outcome of an event is unclear or the subject of doubt
    - Example: the outcome of a coin flip is uncertain
    - Example: a new technology may or may not be ready at the date needed to support a new product
- In common usage, ambiguity can also connote incorrect, missing, or conflicting information
- Whatever the source or type of information ambiguity, that ambiguity needs to be driven out of the design specification, as much as practicable
- Sometimes, the system architect has no choice but to accept some ambiguity in the system design
  - Some things just won’t be known until further consultation or design work is performed
  - It is often advisable for the system architect to make educated assumptions when faced with insufficient information
    - These assumptions need to be monitored and eventually resolved with concrete information, or else accepted as the best we can do

Examples of Sources of Ambiguity

- Ambiguous requirements
- Ballooning set of possible architectural solutions
- Uncertainty in the effectiveness or maturity of a technology being depended on
- Insufficient knowledge of an involved technology
- Insufficient knowledge of possible approaches to analysis that could lead to answers
- Insufficient knowledge of how to apply technology tools to develop solutions
- Insufficient knowledge of who to ask for information that might lead to solution answers
- Assumptions for which it is unknown how to resolve them
- Insufficient knowledge of the critical factors needed to make alternative choices
- Insufficient knowledge of the domain of the problem/solution
- Insufficient number of known variables to solve a logical or arithmetic problem

Making Assumptions

- Often, especially during the initial iterations of a design, needed information is simply not available to make concrete decisions
- Still, decisions have to be made for the design teams to move forward
- In such cases, it is better to make an educated assumption to support a decision and then move on with the design, rather than postpone a decision that keeps the rest of the team waiting as well
- The key point in making assumptions is to make the best educated assumption possible
  - The ability of the system architect to make sound engineering assumptions and judgments during the process of design is heavily dependent upon their knowledge and experience
  - Failure by the system architect to fully understand the ramifications of particular assumptions can lead to poor designs, rework, and lost time/profit
- Valid assumptions can be made when the system architect has either
  - Worked through the numbers on enough occasions to know what effect the assumptions will have on the analysis
  - Or the assumptions are so basic in nature that there is no question of their validity
- However, such assumptions need to be carefully recorded and monitored at intervals to determine
  - Whether enough information now exists to either confirm the assumption (and make it a fact of the design)
  - Or to keep the assumption in force and to continue monitoring the situation surrounding the assumption

Establishing Design Constraints from Downstream Needs

- Lack of sufficient information during design can sometimes be resolved by consulting with downstream processes to better understand their needs
  - This situation sometimes comes about due to a lack of sufficient investigation into those downstream needs during the upstream processes (such as in System Requirements Definition)
  - Nonetheless, such lack of information, when it occurs, can be resolved by the system architect though consultation with the downstream processes
    - Example: a system that will contain hazardous chemicals may need special provisions for removing the chemicals during disposal of the system
    - If requirements were not written to address this situation, the system architect can consult with the Disposal Process engineers to determine what will their future need be
    - The collected information can then established as a new system requirements and design constraints can then be established to force the design into compliance with the needs of the downstream process
- Such a situation is not always the fault of insufficient upstream planning
  - Sometimes, the downstream processes have not been sufficiently stood up to be available to address these topics

System Design Risk due to Information Ambiguity

- Risk is derived from incomplete knowledge of some aspects of the system design or operation
  - Such lack of complete knowledge causes us to make errors in the design or in usage of the system in operations
- If we are somehow able to have truly complete knowledge of all aspects of a system design and operation, then risk boils down to those things that are beyond our direct control (such as unforeseen environmental events)
  - However, being able to have such complete knowledge of a system design and operations is both highly costly and highly unlikely
- The best approach to dealing with information ambiguity is to convert “unknowns” (ambiguous information) into “knowns” (unambiguous information)
  - Especially for information ambiguities that are associated with high risk (high likelihood of occurring and/or high impact when it does occur), such as an event that results in a hazardous system anomaly
- Risks are particularly dangerous if we are unaware that they exist
  - We will proceed without any protection built into the system, since we did no work to prevent the occurrence of the risk from developing, or to diminish the impact of the risk when it does occur
- Even if we are aware of risks, we also need to understand how to deal with them (how to mitigate the risk)
  - It is not uncommon on a project to be aware of risks that are difficult to mitigate

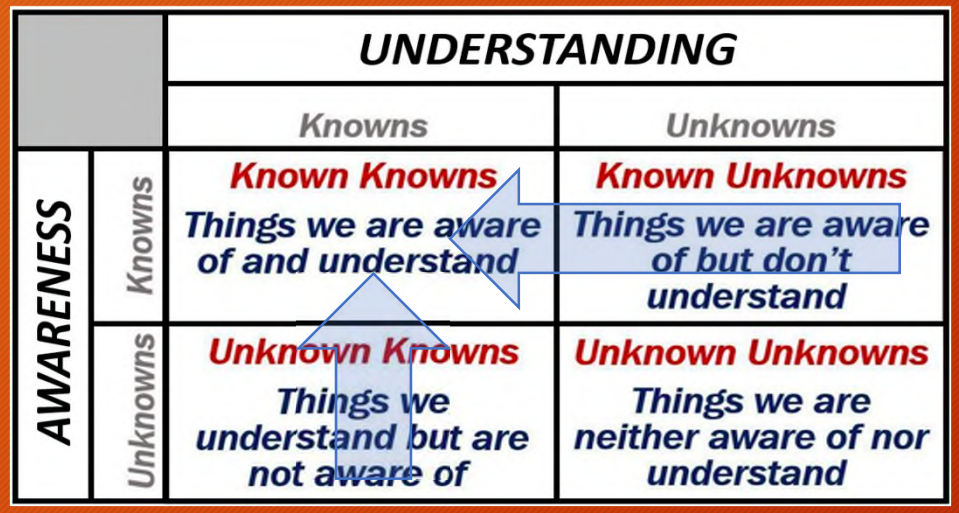
Knowns and Unknowns

- On February 12, 2002, Secretary of Defense Donald H. Rumsfeld introduced to the press, and therefore to the general public, the concepts of knowns and unknowns (and associated risks), saying
  - “Reports that say that something hasn't happened are always interesting to me, because as we know, there are known knowns; there are things we know we know. We also know there are known unknowns; that is to say we know there are some things we do not know. But there are also unknown unknowns -- the ones we don't know we don't know. And if one looks throughout the history of our country and other free countries, it is the latter category that tend to be the difficult ones.”
- These statements generated a good deal of discussion and deliberation about their significance
- However, these concepts were well understood by engineers and scientists long before Rumsfeld mentioned them to the public

Donald H. Rumsfeld

Awareness-Understanding Matrix

- The Awareness-Understanding Matrix characterizes the four possible states of awareness and understanding of system information ambiguity



https://medium.com/the-world-in-the-future/known-knowns-unknown-knowns-and-unknown-unknowns-b35013fb350d

- Awareness addresses whether I am aware that a risk exists of having incomplete or incorrect information
  - Unknowns in awareness are particularly concerning because this implies that the design team simply does not have sufficient skill or experience to be able to recognize that they have incomplete or incorrect information
- Understanding addresses whether I understand the nature of the risk sufficiently well to be in a position to mitigate the risk
- The objective in engineering planning and design is to transition as many Known Unknowns and Unknown Knowns into Known Knowns as practicable
  - Especially those that involve high risk (high likelihood of occurring, and/or high impact when they do occur)

Known Knowns

- Known Knowns – Risks you are aware of, and that you know how to mitigate
  - I am aware that my car will run out of gas if I don't refuel, and I understand how to track how much gas I have left by monitoring the gas gauge
  - The US Pentagon is aware of the importance of knowing how many American troops there are in the Middle East, and understands how to access that information in their database when needed
- Known Knowns are not actually risks, they are facts
  - Better said, they were previous risks that have now been mitigated (through awareness and understanding), and are no longer risks
- With known knowns, there is zero uncertainty – everything is known
- This is the best position to be in (with regard to the design of a system)
- However, getting to this position with every aspect of the system design can be very costly
  - Engineers need to figure out where to place there efforts in reducing uncertainty to get the highest benefit at the least cost

Known Unknowns

- Known Unknowns – Risks we are aware exist, but we do not have sufficient understanding of these things (yet) to successfully mitigate them
  - I am aware that my car will run out of gas if I don't refuel, but I do not understand how much gas I have left because my gas gauge is broken
  - The US Pentagon is aware that there are terrorists in the United States, however, they do not understand who they are nor how many there are
- These are identified unknown facts and are classic risks that Systems Engineers deal with every day
  - Example: we know what kind of system functionality we want the system to demonstrate, but we may not yet know how to get the system to demonstrate that functionality
- With known unknowns, there is uncertainty
  - At least we are aware of what we need to understand
  - From this point, we simply have to develop the needed understanding to mitigate the risk
- This is the second best position to be in (with regard to the design of a system)

Unknown Knowns

- Unknown Knowns – Risks we are not aware exist, but if we were aware, we would find that we have sufficient understanding to mitigate them
  - I am not aware that my car will run out of gas if I don't refuel because nobody ever explained that to me, but if my car stops working I can read the owner's manual to find out that a likely cause is running out of gas, and how to remedy that situation
  - The US Pentagon was unaware that terrorists were planning the 911 attacks, but had they been aware, preventative action could have been taken, since sufficient evidence was there to deal with the situation
- With unknown knowns, there is great uncertainty
  - The worst thing about this is that we are unaware of the existence of the risk (we are blind to it)
  - These are hidden facts
    - If an event occurs, we might realize that we could have resolved the situation, if only we were aware of it
  - This usually occurs with engineering organizations that are poor performers, or don't have sufficient experience to have the needed risk-awareness

Unknown Knowns (continued)

- Can also be viewed as something about the system that is understandable to the engineering team, but its significance is not yet appreciated
- This is the worst position to be in (with regard to the design of a system)
  - You are “flying blind”
- Example of Unknown Knowns
  - The British knew about the Enigma encryption machine before the WWII, as the German designers had filed plans for the device at the UK’s patent office in 1926
  - However, it wasn’t until 1939, when they met their Polish equivalents that the British discovered that it was wired alphabetically, with rotor A wired to the first contact, B to the second contact and so on
  - This was the same set up as in the diagram in the patent application
  - This was known to the British but was so obvious they never even considered it as a possible key to the decryption process
  - https://www.linkedin.com/pulse/unknown-knowns-how-can-hurt-your-project-paul-wiltshire

Unknown Unknowns

- Unknown Unknowns – Risks we are not aware exist, and even if we were aware of them, we would not have sufficient understanding of these things to fix them
  - I am not aware that my car will run out of gas if I don't refuel because nobody ever explained that to me, and if my car stops working due to running out of gas, I wouldn't know what to do about it
  - The US Pentagon is unaware of the intentions of Unidentified Aerial Phenomena (UAPs), but even if they were, they wouldn't have the understanding of how to deal with those UAPs
- Unknown Unknowns are risks that come from situations that are so unexpected that they would not be considered during normal planning and design
- With Unknown Unknowns, there is total uncertainty
  - This is NOT a situation that we can realistically deal with in engineering
  - If such situations arise during development or use of the system, we will just have to deal with those situations the best we can when they occur
  - If we do a thorough job at the planning and design of a system, then we reduce that chance that Unknown Unknowns will pop up unexpectedly
  - If we do further thinking and identify some Unknown Unknowns, then they become Known Unknowns, or Known Knowns, simply because we are now aware of their possible occurrence

Unknown Unknowns (continued)

- Unknown Unknowns do not enter our consciousness when developing design plans for a product (because they are “Unknown” to us) – unless we just “think or work harder”
  - Unfortunately, they sometimes appear during operations despite our best efforts to uncover them during planning and design
  - Unknown Unknowns, when they appear, are generally detrimental
  - Rarely is the appearance of an Unknown Unknown a positive occurrence
- An organization that finds themselves dealing with a lot of Unknown Unknowns should question whether they are in the right line of business to begin with
- One only needs to look back in scientific history to find a break-through point in learning in which someone all of a sudden stumbled on some knowledge that they were totally unaware of prior to the discovery
- Example of Unknown Unknowns
  - In 1895, the German physicist Whilhelm Conrad Rontegen accidentally discovered x-rays while playing with a florescent light bulb
  - By passing electricity through a gas in a low-pressure tube, Rontegen saw that the emitted light would pass through tissue but not through bones or other dense objects essentially giving a picture of the inside
  - https://leslistes.net/top-10-greatest-scientific-discoveries/

Assessing Knowns and Unknowns

- When assessing the position of a particular system design in the Awareness-Understanding Matrix, it is important to specify from what time perspective you are performing the assessment
- There are at least two time perspectives associated with this kind of assessment
- Design Time Assessment
  - This is an assessment that is performed while the system is under development and before the system is put to use
  - This assessment is useful to indicate to the design team where weaknesses exist that need to be addressed in order to improve the likeliness of system success
  - The design team will then have a chance to take actions to convert unknowns to knowns, if possible
- Post-Mortem Assessment
  - This is an assessment that is performed after the system has been put to use and after an anomaly occurs that compels a reassessment of the position of the system design in the Awareness-Understanding Matrix
  - This assessment is useful to indicate to the design team what went wrong with the system during a system anomaly and how their efforts could be improved in the next project (lessons learned)
- Difference
  - The difference between the two assessments, obviously, is that during post-mortem analysis, risks that the design team were unaware of before now have already become known
  - This new awareness might cause the assessor to say that the team “should have been aware” of these risks during the design phase
  - And, had they then been aware of these risks, they perhaps could have gained better understanding of how to deal with the issues when they occur, or prevent them from occurring in the first place
  - In such a case, experience and skill are invaluable to identifying these unknowns during development

Assessing Knowns and Unknowns (continued)

- Consider the previous example:
  - I am not aware that my car will run out of gas if I don't refuel because nobody ever explained that to me, and if my car stops working due to running out of gas, I wouldn't know what to do about it
- From a Post-Mortem assessment, the analyst might say the driver got what they deserved for being so ignorant about how a car works
  - This assessment is unfairly made from the standpoint of someone who already has the facts (awareness and understanding) about how cars work
- However, from a Design Time standpoint, the driver was not fully educated on how cars work, therefore the driver is unaware that gas is a critical resource to the operation of a car, nor understands how to mitigate the risk of running out of gas by monitoring the gas gauge
  - When the car stops working (due to running out of gas), this will come as a complete surprise to the novice driver
  - This is not unlike a design team that takes a prototype out to the field to perform testing and encounters a situation that is completely baffling
  - The team is at least now aware that such a situation can arise, and can now figure out how to deal with the situation in the next design iteration (how to mitigate the new risk)

References

- U.S Department of Defense (2002). *DoD News Briefing - Secretary Rumsfeld and Gen. Myers* https://archive.ph/20180320091111/http://archive.defense.gov/Transcripts/Transcript.aspx?TranscriptID=2636#selection-407.11-407.50
- Wikipedia (2022) *There are known knowns*, https://en.wikipedia.org/wiki/There_are_known_knowns
- Market Business News (2022). *What are unknown unknowns? Definition and examples*, https://marketbusinessnews.com/financial-glossary/financial-glossary-u/unknown-unknowns/
- Dang, Anh (2019). *Known Knowns, Unknown Knowns, and Unknown Unknowns*, https://medium.com/the-world-in-the-future/known-knowns-unknown-knowns-and-unknown-unknowns-b35013fb350d
- Management Yogi (2019). *Risk Classification: Known-Knowns, Known-Unknown, Unknown-Knowns and Unknow-Unknowns*, https://www.managementyogi.com/2019/09/risk-classification-known-knowns-known-unknowns-unknown-knowns-and-unknown-unknowns.html