**Defining Client Deliverables**

**Cornell Cup USA – Arm Enabled**

**Prof. David R. Schneider, Cornell Director of MEng Studies in Systems Engineering, drs44@cornell.edu**

You’ve just been awarded an exciting assignment! -- Maybe it’s to decide whether solar panels should be placed on one of the central buildings of your campus. You’ve been given the semester so you and your team take a week to visit the site and talk with your stakeholders. Out of the stakeholder meeting you create the following agreement which all parties, including yourself, feel well satisfied with:

**Deliverables:** The team shall perform an analysis on whether the college should go forward with the installation of solar panels on the roof of Community Hall. The team shall report on both the estimated payback period and a selection of panel type for purchase. The work shall be delivered in a formal report and presentation.

**Time Frame:** The work shall be completed during the Fall semester and the formal report and presentation will be given during the finals period. 1-2 Mid-reviews will be held throughout the semester to report to the stakeholders on your findings so far.

**Resources:** Your advisor for weekly meetings

Facilities’ Alternative Energy representative by appt.

The next week you find and select a solar calculator (a program commonly used to ascertain the effectiveness of solar panels), and then spend the next week to a week and a half to find and enter all of the calculator’s data. Then in about your 4th / 5th week you compare your results with the cost of solar panels you found on-line and determine a payback period. You’re making progress every week, and it really hasn’t been too hard of a work load. Now you’re feeling really good as you show the stakeholders the answer you already have for them at your first mid-review:

Your team: “…And so as you can see taking the angle of the roof, the geographic latitude, and portion of the roof that is shaded by trees, we get a solar exposure. Now multiplying that exposure for each day of the year by the power production rate of the solar panels we get maximum total production. Then looking at the cost and efficiency of the panels, and the cost of electricity, we get the following payback periods. As you can see, this one has the lowest payback period of 15 years, so we’re done.”

Having never looked into solar panels for a building before, you’re feeling rather proud of yourself and your team for performing this work. …But for some reason your stakeholders aren’t quite as excited as you expected them to be:

Stakeholder: “Does your analysis take into account the number of cloudy days we have here?”

Your team: “Well there wasn’t a place to enter that in the solar calculator.”

Stakeholder: “Well that still is going to affect your results, yes?”

Your team: “Well we could multiply it by another efficiency. I think I read somewhere that you get 40% of the maximum solar radiation on cloudy days…”

Stakeholder: “Is that a number you know or are you guessing? Can you provide the reference and show how that reference applies to our situation?”

Your team: “Well I’m not sure how they came up with that number but…”

Stakeholder: “Well how does it connect into the building’s power system? Is there an efficiency rating for that as well?”

Your team: “I’m not sure.”

Stakeholder: “Well naturally it has to hook in to at least the building’s power system if not the local grid. What is the estimated cost associated with that?”

Your team: “We haven’t looked into that.”

Stakeholder: “In your payback period, did you take into account that the cost for power from the power company fluctuates during the day and night? Have you considered at all any ways to store excess energy as well?”

Your team: “No we didn’t look into cost fluctuations. But that’s a good idea about the storage. We should look into that.”

Stakeholder: “Well bring up your estimated daily power consumption / generation chart.”

Your team: “We didn’t create that….”

Stakeholder: “Well that’s part of a standard solar analysis. You need to do that in your final report. And when you look into the energy storage remember that will, of course, cost more too. You should look at government incentive programs as well. Perhaps there’s a way to subsidize the cost.”

Your team: “Um yeah, we hadn’t thought of that.”

Stakeholder: “I didn’t see anything in your presentation about how often the panels need to be maintained? Does snow need to be cleared off as well?”

Your team: “I don’t think snow is an issue as it slides off right?”

Stakeholder: “Well we can’t just hope so, because if we need to pay someone to go up there every time it snows that’s going to add into the costs. And then there’s safety issues with them being up on the roof. And we need to know the reliability of these panels and any associated components.”

Your Team: “We’ll look into that too.”

Stakeholder: “Can you at least show us a rendering of the building with the solar panels on it?”

Your Team: “We don’t have the skills to do a rendering.”

Stakeholder: “You need to get buy-in from the building manager and the architectural review committee for both the look of the building after installation and to ensure your design meets building codes.”

Stakeholder: “By the way, there are also plans to build another building close by. Is that going to affect the shading on your panels? And the roof has to be re-done in 10 years. Anything with a payback period longer than that isn’t worth it.”

In the end, everyone is disappointed.

Your team’s final impression: “But we did everything that they asked us to. If we had known they wanted those other things, we would have done it.”

Stakeholders’ final impression: “These students just don’t do good work.”

But it didn’t have to go this way and it all starts with defining your deliverables. Far more than helping you just manage expectations, this guide will use this solar panel example to highlight some of the techniques and tricks you can use to explore what needs to be accomplished in any kind of project in order to have that project be a success.

One good tool to formalize this is called a Deliverable Tree (a.k.a. Question Tree, Deliverable Graph). This guide first introduces a summary of the key steps to construct the tree so you can get an idea of how all these steps work together. Although the approach is presented in a linear step by step fashion you will most likely repeat various steps in a very iterative, non-linear fashion as you build your deliverable tree. After this step by step reference overview, this guide then goes into more detail with key Tips on how to perform each step well. Once the Steps and Tips for creating the standard deliverable tree are explored in detail, the guide concludes by offering a number of optional ways to enhance/utilize the tree further.

Additionally, a sample deliverable tree for this solar panel example is also provided in a separate document “BuildingTheDeliverableTree.pptx” which shows how one can be constructed in an iterative fashion and continually improved upon with every iteration. The sample also contains comments for the reader on its construction to make it easier to follow and to highlight important aspects. Before beginning to create your own tree, read through the details of Step 0 below and then continue onto the Deliverable Tree Summary section. – Please also note that people use software other than PowerPoint, including the very popular free on-line software MindMup, to create their trees. PowerPoint is easier to share and common to more people and hence it is used for this guides examples.

**Step 0.** Don’t expect to be told everything you have to do. Figuring out what you need to do, and yes even what are the expected deliverables, is an essential part of being any kind of designer or problem solver. It may seem counterintuitive because if they know what they want, why don’t they just say it?

The problem is that they are trying to tell you; it’s just that humans aren’t always the best at communicating in a way that can be understood from someone else’s perspective. A common source of this problem is that stakeholders assume that the terms they use, which may have a special meaning in the context of their work, aren’t confusing or unfamiliar to you (you’ve probably run into professors before who knew the material so well they had a hard time explaining it). Stakeholders also tend to want to tell you information as quickly as possible as they want you to do the work, not them. Similarly, as you don’t want to have to take any more of their time than necessary (and don’t want to look “stupid”) so what tends to happen is that everyone naturally avoids the details and more easily accepts ideas that sounds like everyone can reasonably agree on, i.e. “we want it to look good – you know what we mean, right?”

It’s also possible that the stakeholder simply doesn’t know how to describe what they want -- but they’ll know it when they see it. In many cases, the stakeholder is not an expert in what they’re asking for (that may be why they’re asking for it, instead of doing it themselves). So they may not be able to accurately express their needs in their initial request. As a general rule, you need to be smarter than your customer in that if they express that they want something called “A”, what is the real need that “A” addresses and can you perhaps develop something other than “A” that could meet that need even better. For example, if someone asks for a car but never heard of an airplane, they wouldn’t know to ask for an airplane even if it meets their needs even better than a car would.

The deliverable tree method can help you determine if this latter case is happening as well. If it does, the deliverable tree may also help you flesh out how “they’ll know it when they see it”. If this is the case though, you should probably at least use the Define your System guide and quite possibly parts of the Delving into the Stakeholder’s Needs guide as well.

**Step 0 is Complete When:** You recognize that you’re going to have delve into “what is being asked for?” in order to determine what is really needed to successfully complete your project.

**Deliverable Tree Summary**

The deliverable tree itself can be described as a graph of primary tangible deliverables (i.e. the deliverables that will actually be turned into the client) and the intermediate tangible deliverables and data that are necessary to generate the deliverables for the client (i.e. the deliverables that you and your team must create in the process of making the final deliverables). It can take some practice to get used to thinking about how to break up goals into sets of deliverables and tasks. So to help, this guide reviews the deliverable tree 3 times:

* 1st just as a quick overview of the general form and procedure of creating one
* 2nd showing the construction of a more complete example, but just focusing on how the example fits the general form while highlighting a few key practices that go into using the tree for real.
* 3rd using the construction example to demonstrate a dozen tips on how to make best use of the deliverable tree to ensure you meet the needs of your clients and as a planning tool for your team.

Three walkthoughs may seem excessive but like any tool, the deliverable tree can be used well or one can simply go through the motions of creating the tree without offering any real value – we obviously want to help you create as successful a project as possible and hence we’ve taken the time to explore it and we think you and your team will be rewarded to do the same.

For the 1st introduction, if you haven’t already, open the general form example “DeliverableTreeGeneralFormExample.pptx”. Then read through the blue text of Steps 1, 2, 3, and 5 below to get a brief overview of how the deliverable tree works. Then once you’ve reviewed those you can move onto the 2nd walkthrough, opening the example construction of a deliverable tree in the PowerPoint file “BuildingTheDeliverableTree.pptx” and review all of the Steps 1-7. In the 2nd walkthough it’s still not important to look at the specific details of the construction example but just recognize how the general form takes shape in implementing a real tree.

Both of these example files will be referred to throughout this guide for . To begin with some basic terminology, each “box” in the tree will be referred to generically as a “node” and the box(es) at the top of the tree is referred to as the “root node”. Similarly any series of connected nodes is referred to as a “branch” and the nodes at the end of a branch are referred to as “leaf nodes” or simply “leaves”. In this way, the resulting graph can kind of resemble a tree that is upside down (root at the top, leaves at the bottom ends, connected through branches in between)

**Step 1:** To begin, start off with your main known deliverables. These are shown in example as the first row of nodes coming off of the root node in slide 1. Many of these initial nodes may be in the customer’s terms and hence may still be rather vague, but their meaning will be better defined as you continue to grow the tree.

**Step 2:** For each deliverable, determine what tasks need to be completed or what other deliverable results need to be obtained to achieve that initial deliverable. Make branches for each of these tasks and/or deliverable results. (i.e. in order to achieve that initial deliverable, what else is needed and what else must be done.)

**Step 2a:**For each newly identified task, compile a list of the kind of data and resources needed to complete it. Make additional branches for these sets of data/resources. Then add additional tasks immediately off of these sets of data/resources that focus on how to obtain that data as needed. These data/resource nodes are marked in example slides in green.

Recognition of the data necessary, or even an early estimate of what data is required, is a way to recognize what your first steps in your project work should be (i.e. the collection of that data) . This data collection can help your team be more productive more quickly in the beginning of a project.

**Step 2b:**For each deliverable and its supporting tasks, determine what criteria will used to assess its performance quality. (i.e. ask yourself and your clients to define what is the difference between doing something and doing something well?). Add branches to record all the ways the deliverables and tasks will be assessed. These nodes are marked in the slides in yellow.

Quite often this can be one of the most valuable steps and largest points of discussion with the client. For example in slide 13 of the construction example, recognition that the client is expecting estimates of time, required expertise, safety issues, and some estimate of “complexity” to be included as part of their cost estimate analysis is something that you might have missed had you not taken the time to explore this further. It also helps recognize that “complexity” is something that is not well defined in this context and steps should be taken early on to formalize this – otherwise how else are you going to know that you’ve completed your work well?

**Step 3:** Re-arrange the nodes as necessary, potentially splitting the tree’s branches onto different pages so the tree is easy to read. – If this is your first time seeing a deliverable tree and are reviewing the general form example, you may skip onto Step 5. Please come back to review all the steps when you are ready to look at the “BuildingTheDeliverableTree.pptx” construction example file.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Large trees are quite common and hence it is also common to split the job of delving into different branches amongst the team and then ask the team to reassemble their work and in doing so, review each other’s work via Step 4 below.

Advancing from slides 6 to 7 in the construction sample is an example of a tree re-arrangement that helps with the expansion of the Cost Estimates branch started all the way back in slide 1. Note that slide 7 is labeled with a breadcrumb title in the upper left hand corner to indicate where this figure ties back into the rest of the tree. There is another example of this between slides 9 and 10 via the block “Analysis.3Panels.CostEst.GovtIncent”

Note that the wording in the construction example may still be imperfect. Some deliverables are written fairly tangibly and well on their own such as “Power Consumption / Generation Chart” as it is pretty clear what needs to be given to the client here. Others such as “Installation Cost Estimates” are still perhaps too vague on their own, but when combined with the nodes farther down that branch, the data needed and especially the ways it will be assessed, a more complete picture of what is expected in that deliverable becomes clear.

Do not be afraid to have large box names as in many cases these larger names are far more beneficial. For example, even your first iteration of your tree, you should at least recognize the difference between saying something like “Develop an understanding of…” vs. “Creation of an example problem that demonstrates how the system will operate and at what performance under all identified use case scenarios” and understanding that the latter is a far more tangible deliverable that you can clearly tell when it has been sufficiently achieved.

**Step 4a:** Examine the branches of the tree for repeated tasks/deliverables. This can help reduce re-work if it is understood how all the ways the results of a task need to be used. It can also help determine the priority of achieving the task if many later tasks/deliverables depend upon it.

**Step 4b:** Examine the branches of the tree for common data & resource needs. Common data nodes indicate that that data influences multiple deliverables and hence may be quite important to the overall outcomes. This in turn may suggest that trade-offs may need to be made between the deliverables’ performance.

An easy example of this is that the maintenance of the panels is highly linked to the layout of the panels but so are the maximum output and efficiency of the entire system. The tree therefore helps you recognize this trade-off early on before the panel layout was optimized for maximum output and efficiency only to later realize that maintenance issues need to be accounted for which in turn would cause the layout to have to be redone.

**Step 4c:** Examine the branches of the tree for common performance criteria. Recognizing that multiple deliverables may influence the same performance criteria can help you, in turn, recognize various ways to achieve the same kind of value for the client and thereby enable you to optimize your work and your time to do so.

In the slide 15 construction example, determining a metric for complexity is identified as a task in 2 separate branches. Although in this case each situation may call for complexity to be measured in different ways, the same team members who work on one metric may want to work on the other or at least compare findings to develop a more cohesive and complete understanding.

Recognizing common performance criteria can also impose unforeseen constraints -- while a certain amount of complexity may be tolerable in one situation, it may be impractical to handle in the other – and it’s important to recognize this before you design to the more lax constraint and then have to redesign later to meet the more restrictive constraint.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Step 5:** Repeat steps 2-4 until your bottom nodes are tasks that cannot be reasonably split up further, deliverables are clear, and the team has what is necessary to begin working on them. Nodes that cannot be split up further are also called “atomic nodes” as they are your basic “atomic” building blocks to your project.

In other words, keep iterating and drilling down further until for each atomic task, you:

1. Know how to perform it correctly
2. Know how to assess the quality of your work/solution from both your own and your stakeholders’ perspectives.
3. Know how to get all of the needed information accurately and in a timely manner.

In focusing down one subset of branches, you may wind up creating tasks that were already created in other branches. That’s ok! And actually beneficial at this stage as it can emphasize for you how a task or a piece of data, etc. is needed for multiple aspects of your work.

You may stop here and open the construction example file and then review all of the Steps for your second review of the deliverable tree.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Step 5a:**If the team does not know how to begin working on a task, end the branch with a set of questions that need to be answered and a set of tasks designed to them find the answer instead -- such as looking up information or talking with an expert or stakeholder further. It can be worthwhile to add performance criteria to the questions as well, i.e. what should a good answer tell you, or what might a good answer look like. Sometimes questions are represented with a different shape box. Diamonds or parallelograms are common shapes used to help make questions standout.

**Step 5b:** Take a break. It can take some real lateral thinking to be able to delve down into all of your tree’s branches. If they aren’t all perfectly explored yet, that’s okay. Continue only when you have been able to put your deliverable tree out of our mind for a little while and are now ready to review it again.

**Step 6:** Review at least the deliverables and performance criteria as being accurate with the client before starting the work. This step can be carried out and repeated at any point / iteration of creating the tree but a final “sign off” from the client that the work outlined in the tree will meet their needs can greatly help manage expectations and ensure a meaningful end result.

**Step 7:** Refer to the deliverable tree regularly (*at least weekly*), particularly the ways the deliverables will be assessed, to ensure that the team is performing tasks and prioritizing efforts well. The tree can often be used as a precursor for the development of a detailed timeline.

Following the Steps above the team from the initial presentation example could have created a small questions tree that only covers what the team did. Like any tool, the deliverable tree can be used well …or used not so well, e.g. lots of people will dance but not everyone dances at the level of a professional. Assuming you want to be a professional designer, here is an initial list of tips & approaches you can use to flush out your deliverable tree and delve into what is really being asked for.

Many of these approaches can and should be explored more using the other guides that are a part of this set, but to what extent they are used will be determined largely by how much you are being asked to design something new vs. solve a specific problem. The solar example this guide offers is more of a specific problem but even if you are designing something new, for example designing a brand new kind of solar array, you will most likely have to solve some specific problems along the way and these tips can still be quite helpful.

**Tip 0:** **For each deliverable / task node, ask the questions, “If I have to complete this deliverable…**

* …what do I have to do?”
* …what do I have to know?”
* …how do I know this deliverable is done?”
* …how do I know this deliverable is done well?”

For your answers to those questions, create new nodes. But do notice the difference between being “done” and “done well”. “Done” means you have met the requirements of the task, i.e. “it works”. “Done well” means that you have some way of objectively assessing the quality of your work and thereby can prove its value. The qualities that your work will be assessed on are called *performance criteria*. And having an objective way of measuring your performance criteria is referred to as having *performance metrics*. In creating your tree, you want to at least be able to identify any performance criteria, which can then be flushed out further with the help of both “Performance Criteria Tips for Deliverables & Decision Matrices” and the “Decision Matrix” guides.

“What do I have to know?” is most commonly answered with sets of data that are needed. It may also be used to specify a special technique someone on your team may need to know how to perform. Particularly when you’re starting your project, answers to this question may also indicate that you don’t know something at all and need to find out more information before you could answer any of the other questions. Some examples of this in the example tree are the “Building Renderings with Solar Panels Installed for min 1 Top Panel Choice” node on slide 24 or the “Architectural Code Requirements” node on slide 16, where the team doesn’t know anything about renderings or architectural code, but only that they will have to show that the architectural code is properly addressed, i.e. they just don’t know yet what has to be done or what they even have to show in order to prove that code is met. So the team will have to create additional tasks to learn more about this area, or obtain additional resources / help from outside experts who may be able to help with these needs.

**Tip1: Explore what’s meant by any generic names, or terms that may be professionally specific:** An analysis, a report, a decision, a recommendation, an understanding are all common terms as deliverables but are rarely specific enough on their own. As shown in the example, there were a lot of elements the stakeholders expected in the “final report” that weren’t specifically called out. Commonly, stakeholders may associate a generic term with a certain standard that they somehow assume everyone will naturally be familiar with. This can be commonly problematic if you’re doing outside consulting work for another group.

When you encounter any generic terms, ask questions like the following:

* What do they need to see in that analysis or report specifically?
* Are there specific calculations they are expecting? Are there additional constraints/regulations that need to be specifically addressed?
* What information do they need to have included to make that decision?
* How do you know when an understanding has been achieved or an analysis is considered completed?
* Most of all: What makes for a well performed analysis/report/etc. versus a poorly performed one?

These will all help the conversation get started but expect to continually have to delve and repeatedly ask variations of these questions (like you were building a deliverable tree with your stakeholder). Remember, to your stakeholder what they’ve asked for is perfectly clear in their mind. They may even be a bit uncertain as to why you’re asking so many questions, but keep a positive attitude, don’t worry about looking foolish (you’ll look more foolish later if you don’t) and keep digging until you find the information.

Remember these comments from the solar example Stakeholder: “Well bring up your estimated daily power consumption / generation chart.”... “Well that’s part of a standard solar analysis”... : “Can you at least show us a rendering of the building with the solar panels on it?”

**Tip 2: Watch out for Incomplete Responses.** In conversations with even the best intentioned stakeholder, it is not uncommon to still fall into the trap of agreeing too quickly that everything has been specified. For example:

Your Team: “What do you need to see in that analysis specifically?

Stakeholder: “Good question. We’re looking for solar power production vs. estimated building usage shown over the course of the year, the cost & payback over the lifetime of the panels, integration into the building, you know, that kind of thing.”

Your Team: “Great! We’ll look into that.”

Both your team and stakeholder leave feeling that this has been a worthwhile discussion and everyone has a good understanding of what needs to be done.

However, even among the things mentioned, it still is lacking specificity. What does “integration into the building” entail? Even if you have significant past experience in solar panel work like this, the example shows significant DANGER. Any statements like “you know, that kind of thing” are a huge red flag. This implies not only may there be more to discuss but that the stakeholder feels that it is okay to gloss over the details. Do not let statements like this slide by. If needed also see Tip 11: How to Handle Running out of Stakeholder Meeting Time.

Unlike in a class situation where you just need to get a hint from the TA to be pointed in the right direction, in your professional career you have to think of situations where you are defining the deliverables more as developing a contract. You need to have all of the deliverables spelled out specifically and objectively in a way that all parties have agreed to; see the Performance Criteria Tips for Deliverables & Decision Matrices guide for more information on deliverable assessment and contracts.

**Tip 3: Ask for Samples of Past Similar Work.** Try to get this, or even any part of this, as soon as possible. And then review it as soon as possible. Look to see if there is anything in there that is unexpected or you are unsure of, or unsure of even why it’s included. If effort was spent in a particular way before, there usually is a reason, and it’s best to find out why.

After you have reviewed any samples yourself, it can be quite valuable to review the samples along with any questions you have with the stakeholder as you may be surprised why a certain calculation / performed technique / etc. is so important to them.

If samples aren’t available, all the more reason to make sure you spell out your deliverables very clearly.

**Tip 4: Find out who are all of the Stakeholders:** “You need to get buy-in from the building manager and the architectural review committee for both the look and meeting building codes.” Just because you’re working with one main point of contact, doesn’t mean there aren’t others that need to be satisfied.

* Are there others that we need to get approvals from?
* Who else will these deliverables affect?
* Who will be the primary (& secondary…) users of our deliverables?
* What are the needs, priorities, deadlines, etc. of any of these other stakeholders?

Sometimes it can be hard to pin down exactly who all the key stakeholders are but you will want to repeat many of the suggested questions in this section with each of your stakeholders.

**Tip 5: Explore what your deliverables must work with.** Very few things stand completely alone. In the example, “Well how does it connect into the building’s power system? Is there an efficiency rating for that as well?” The key here is to determine what part of the overall system you’re working with that you have control over, what parts could you possibly influence, and what parts are you just stuck with.

Trying to develop a *Context Matrix* (see Defining Your System Part I) can be a good way to formally explore these boundaries to your problem. Some problem’s deliverables can be fairly narrow, i.e. designing a specific gear box that works with a motor. Others can have multiple and quite varied interfaces but it’s your job to determine all of them and the requirements they impose upon your System.

**Tip 6: Explore what are ALL of the situations that need to be addressed.** “Does your analysis take into account the number of cloudy days we have here?” is an example of determining the situations that your deliverables should be able to address. Your stakeholders may or may not be savvy enough to help you with this, but regardless, this is something you need to explore.

* What are some of the common situations you would expect?
* What are some of the unusual/special situations you would expect?
* How important are each one of these situations?
* What would you be concerned about in each one of these situations?
* What are some of the ways the deliverable will be used after you’ve completed your work?

These can help get the conversation started and should at least inform you as to how savvy your stakeholder is about what they’re asking for. If your stakeholder isn’t savvy enough to answer these questions (which may be part of the reason why they’re hiring you to solve their problem) you still need to find resources that can inform you of these. Also try not to limit yourself to one source: “Well there wasn’t a place to enter that in the solar calculator.”

If you need help in this area, start by trying to create a list of *Use Cases*, as is specified in Defining Your System Part I. Depending on how that goes you may want to explore other guides such as Defining Your System Part II and/or Delving Into Your System’s Needs. This is actually essential if you are designing something new; whether that’s a net zero building, the latest cell phone, a disaster management plan, etc., you will have to explore the needs that this new something is trying to meet.

Why are they asking for you to build a cell phone? Why do they want a cell phone, or rather why do they *think* they want a cell phone? What needs are they trying to meet? By exploring their needs you may find that they don’t want a cell phone after all, that’s just closest thing the perhaps non-technically savvy stakeholder could think of that might meet their needs. Exploring needs are essential for innovation, after all, how do you think tablets came to be popular? What set of needs and performance metrics could lead someone to the conclusion that people would be interested in paying more for a smart phone without the phone? Exploring the needs can lead to new opportunities that might otherwise never even be considered.

**Tip 7: Clearly defining what is the scope of your work** Once you’ve determined many of the ways your system will have to interface with other elements outside of your design boundaries (Context Matrix) and all of the situations you’ll have to handle (Uses Cases) you can begin to determine the possible scope of your work. In your conversations with your stakeholder, you can use things like your Context Matrix, Uses Cases, and of course your Deliverable Tree to explain the possible scope of your work as you see it. You can then decide together what should be within the scope of your current project. Things to keep in mind with regards to scope are brought up in many of the following Tips but a few key ones are:

* Hours / week that can be allocated
* Resources available, including outside expertise, and expertise in your team
* Testing or final verification required
* Timeliness of when other projects’ work/decisions need to happen
* The current starting point of the team (including any team training that may have to occur)
* Priority as compared to other projects’ team commitments

Your stakeholder may naturally want you to do everything under the sun; they want as much good work from you as possible. Stakeholder: “Have you looked at all into ways to store excess energy?” Your team: "… that’s a good idea about the storage. We should look into that.” There is also a natural desire to want to please your stakeholder but be careful what you commit to, even if it is a very good idea. At the very least it is important to prioritize which aspects of the deliverables should be addressed first.

Depending on how good of a working relationship you have with your stakeholders they may also be okay with you making a list of potential additional deliverables that could be added on later and perhaps decided on at a mid-project review meeting. This, of course, places a lot of trust in you that you will work very hard to meet your goals and that these add-on/stretch goals are something you really will shoot for. But the more “new” to you and/or your stakeholders the product / service / etc. that your developing is, the more likely there may be unexpected issues that may arise, and hence the more careful you should be in what you guarantee. After all, being able to reach your stretch goals is far better than not meeting all of the goals you agreed to.

Overall though, the Deliverable Tree can help your stakeholders understand better how much work is required to achieve various goals. Therefore they may be more willing to listen to your perspective on what should be the scope of a project as well.

**Tip 8: What are the ways your deliverables will be assessed?** Figuring out how you are going to be assessed is a really good way to determine what work needs to be done, as well as where your priorities should be. In the solar example, criteria such as maintainability, aesthetics, and available incentives were all aspects that the stakeholders would judge the quality of their deliverables but none of these were specifically mentioned in the original document. Read the Performance Criteria Tips for Deliverables & Decision Matrices guide for information to help you define how your work will be assessed and determining new deliverables as a result.

**Tip 9: What resources do you have available?** Resources can be financial, equipment, software, experts, available labor assistance, etc. If a resource is being offered, it must be for a reason. Inquiring about a resource or how stakeholders expect a resource to be used can be a great indicator about the deliverables. If you feel there is a deliverable that you cannot meet because one of these things is lacking, it is important to make that clear

* Who will be the main point of contact for that information?
* What sources of information do you find reliable enough to use in your deliverables?
* Who is in charge of that equipment? Can we get trained on that equipment?
* We don’t have experience in that area. Is there someone from your team who we can work with? And will they be held responsible for being available to us?
* Do you have all of the resources available for us to complete this project?

After you’ve reviewed your deliverables in more detail, you may find that there are insufficient resources. It is important to make your stakeholder aware of this as soon as is practical. However, you may want to come to that meeting with alternative options on the deliverables, i.e. you can’t do ‘X’ but given just a little bit extra you could do ‘Y’ instead. You should also try to come with cost and availability information on the needed resources. This will help the stakeholder be able to make a more informed decision with you as to whether the additional expenditures are worth it or perhaps it’s best to set this deliverable outside of the scope of this phase of the work.

Hold off on fully committing to the deliverables until you have confirmed that resources are available. Or at least, say that a deliverable is contingent upon resource availability and make sure the stakeholder is aware of and agreeable to that condition. Sometimes a dashed line is used for deliverables in the original deliverable tree but have later been cut out of the scope. These out-of-scope deliverables may be still included in the tree though so that people are aware that this was consciously cut and not simply forgotten. It can also serve as a reminder as to what you may want to include in future stages of the project should more resources become available.

**Tip 10: Think about how your system could fail.** Your team: “I don’t think snow is an issue as snow slides off right?”, Stakeholder: “Well we can’t just hope so, because if we need to pay someone to go up there every time it snows that’s going to add into the costs. And then there’s safety issues with them being up on the roof. And we need to know the reliability of these panels and any associated components.”

Thinking about the ways things may happen less than ideally can be a great way of figuring out what you have to do in order to ensure they don’t fail, or at least lessen the failure probability or severity. Granted, failure can be a “scary” word at first but it’s far better to recognize the potential of failure when you still have time to adjust your plans, rather than discover the failure potential first hand later on. The Brainstorming Risk guide offers ways to systematically think about potential issues and how to resolve them. Some additional important points to consider when discussing potential failures:

* Determine tests that must be passed for deliverable qualifications
* Reliability requirements can be good to base performance metrics on.
* Determining what situations your deliverables are able to address and which they can’t helps to set the scope of your work; as a simple example: if you were designing a watch, perhaps you can make it shock resistant at this phase, but will leave water resistance for another phase.

**Tip 11: How to Handle Running out of Stakeholder Meeting Time:** It’s not uncommon to be in the situation where your stakeholder doesn’t have the time to go into more detail on the deliverables but you know that there needs to be more specification. In these cases, end the meeting by telling the stakeholder that you will draft up a specific deliverables list with performance criteria. You must also tell them that they must review the list and criteria with you for final approval before any work can commence – hence they will certainly make themselves available for that future meeting.

**Tip 12: Review your Deliverable Tree Deliverables with Your Stakeholder:** Especially if this is the first time working on a project of a certain type or it’s the first time you’re working with a new stakeholder, this deliverable tree process can be a surprisingly helpful communication tool. If you’re just starting out, you’re encouraged to even show the example deliverable tree provided here with your stakeholder so they can understand how they can best help you, and what level of detail you need in order to be successful.

Obviously the person giving you the task wants you to be successful; they have a need and they are asking you to meet it. In the end, whatever tree and/or set of deliverables you do come up with, it is wise to review that with your stakeholder. That way, if they do come back with additional requests, you can show that you did fulfill your work as they had agreed upon.

An important note on student based projects or any projects in your career where you will have to do some learning to accomplish the deliverables: Stakeholders need to be aware of your background and skill level. The best scenario is to help them recognize that they need to take ownership of the deliverable definition as well. The first time (even the second, or third time…) you aren’t going to know everything that should go into an “analysis”. Stakeholders may forget that what is second nature to them is not second nature to everyone they work with and when everyone takes some responsibility in the outcomes, everybody wins.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

The remainder of this guide provides some optional Steps that you can follow if you’d like to take your deliverable tree further. It’s probably best to be comfortable with the Steps listed above and using your tree effectively before attempting to incorporate these additional steps.

**Step 8: (optional) Color and rearrange the nodes for visual clarity.** This step can be done as you go, or left towards the end; whichever works best for you. Many people, especially if in their first few times doing deliverable trees, find that worrying about what to color what nodes as you go can actually be too burdensome and take away from the “flow” of the tool.

Regardless of which works best for you, a common color / shape scheme based on the questions of Step 2 is shown below.

* Actions/Tasks (light blue, rounded corner rectangle): what we have to do/perform
* Tangible Deliverables (blue, rectangle): what we need to produce or give to the stakeholder
* Data we know (green, rectangle with a cut-off corner): what we need to know and do know
* Questions we need answered (orange, parallelogram) : what we need to know and don’t know
* Performance Criteria (yellow, hexagon): how the deliverables’ quality will be assessed
* Links to other branches of the tree that are on other slides/pages (red, oval)

The distinction between Actions/Tasks and Tangible Deliverables does not necessarily need to be made but can be used to emphasize what actually needs to be handed over to your stakeholders/clients. There is not a right or wrong way to color your tree however but some other common uses of different colors/shapes are to show:

* where the answer to the question might be coming from, i.e. what stakeholder, test, etc.
* what teammate is responsible for a task / finding an answer to a question
* what time frame which tasks/questions should be addressed
* the importance of each task/question; for example, coloring something bright yellow to indicate that, depending how this question is answered, it could greatly change how you perform other tasks or even what is or is not a valid solution.

Sometimes these 2 lists of shape & color ideas can be combined. One might following the standard way in the first list for the shape, and fill in the color based upon ideas offered in the second list.

**Step 8 is Complete When:** You have agreed upon a common way to color/shape code your deliverable tree amongst your team and potentially with your stakeholders as well. You have also implemented this scheme for your color tree and arranged the nodes in a way that is easy for another reader to follow.

**Step 9: (optional) Transforming Deliverable Trees into a Responsibilities Chart or Timeline:** Now is a great time to translate all of your deliverable tree slides into a timeline, so check out the Timeline Guide for a more traditional approach and for some guidance on project planning. You may likely find a few more tasks / deliverables, performance criteria, needed data and questions in creating the timeline as you review your deliverable tree work as well. Due to overlap in the themes and steps in the Timeline Guide with what is discussed in this guide, you may also be able to replace some timeline Steps (such as determining subtasks) with the work that you have already done here.

Alternatively if you’ve created a very in-depth deliverable tree though, you can also add boxes that represent you and your teammates at the bottom of the tree. Then connect teammates to the questions, deliverables and/or atomic tasks they are responsible for (some people also assign a color to each teammate and color the deliverables they are responsible for as mentioned in Step 8). Along each teammate deliverable connecting line you can also assign either a due date or the amount of time it takes to complete the task.

Continue this process of adding due dates / tasks’-required-time-to-complete back up the branches of the tree until all edges coming into a task/deliverable have a due date or a required time to complete. You may also color these edges and/or tasks/deliverables to signify who is responsible for these each task/deliverables. The assignment does not need to be continuous however, i.e. a teammate can jump across different branches as needed. Once this is done you now have an estimate as to how much each teammate has to accomplish and an order in which to do these things in. Your deliverable tree has now essentially become a timeline.

For the initiated, if you labeled your edges with the tasks’/deliverables’ required time to complete, you may also find it easy to treat your transformed deliverable tree as a PERT Chart. You can then apply many of the PERT chart techniques to your tree, such as determining your project’s critical path. That is outside the scope of this guide but information on PERT charts can be found in many project management texts.

**Use Case & Functionality Variations on the Deliverable Tree:** As you are delving into the deliverables and tasks, it is not always clear what actually must be done particularly if you are designing something new and you’re not quite sure even what that something new has to be or do entirely. i.e. You only know that it has to do ‘A’ but you’re uncertain how to further break ‘A’ down into say B, C, and D.

In this case, instead of ending your branches with atomic tasks, it is common to end them with functionalities that must be achieved. Take designing a new cell phone for example; you may not be certain how your system is going to receive calls but you don’t know what tasks it will take to enable your new device to receive calls. So for the time being “receiving calls” is the end of your branch… well, sort of.

Receiving calls could also easily be considered a use case by itself. So to be an even better designer, you could actually create a use case behavioral diagram that helps to spell out the functional requirements that must be achieved in the receiving calls use case. This is a more advanced technique but breaking use cases into use case behavioral diagrams is a common staple of professional design. So for each use case functional leaf to your deliverable tree, you may want to consider creating a use case behavioral diagram. For more information on use case behavioral diagrams and functional requirements, check out Defining Your System Part II.