# System Interfaces and Change Control/Tracking Interface Matrix Guide

Interfaces between components, subsystems, or even your system and exterior systems (i.e. systems that you did not create but your system must work with) are one of the most important aspects of a design to analyze & track, yet are often one of the most frequently ignored design aspects. Take, for example, the common case where two people are building two different subsystems, say perhaps one of the example pairs shown below:

| Example Subsystem 1 | Associated Example Subsystem 2 |
|---|---|
| Motor drive system | Power system |
| Communications system | Sensor I/O hardware system |
| Chassis design of a robotic rover | All other subsystems that have to fit within the chassis |

By themselves each subsystem may be very well thought out and work very well. But it is not hard to imagine that the motor system may cause power spikes that the power system design did not anticipate and can't safely handle. Or for the communication system to be very reliable for the test data messages the communications team created, only to find out that this message format can't handle the kind of data that is generated by the sensor I/O system. Or that the chassis was designed to hold only so much weight and although all the other subsystems are individually well below that weight limit, combined they exceed the limit significantly.

These kinds of issues are quite common and hence it is often said that if a system is going to fail, it's going to fail at the interfaces. This is often because the people designing a subsystem may understand the needs and trade-offs of their subsystem but are not as familiar with the needs of other subsystems. Then in optimizing their own subsystem, one subsystem team may inadvertently and even more scarily, unknowingly, significantly hindered another subsystem.

This document provides you with a tool to help prevent these kinds of issues from causing you significant problems and better understand the needs of the entire system by creating and maintaining an Interface Matrix.

---

## How to Create an Interface Matrix

An interface matrix is a list of information that might be needed across various subsystems. It includes which subsystems are responsible for what pieces of information and when. It also helps provide a means for updating designers of other subsystems of any changes and even provides a mechanism for ensuring that these changes are discussed with all of those influenced before the change is implemented.

There are many fancy software packages that can be used to assist with this but for many projects a simple spreadsheet can do just as well. The steps below discuss a means of creating an Interface Matrix in Excel. A sample Interface Matrix is also provided in the file InterfaceMatrixBasicSample.xlsx. It is worthwhile to read through all of the steps first as it is often useful to do some of them at the same time.

**Step 0:** You can begin starting to create an interface matrix when you have divided the team / system design responsibilities into various subsystems, dependent upon the needs of the project, the skill sets of the team, and the resources available.

For the example InterfaceMatrixBasicSample.xlsx file, it will only be considering 3 subsystems although a real project may have quite a few more. Hence the sample file should only be considered as a sample as additional subsystems and interfaces would most likely need to be added for many real systems (some are purposefully left out in this example to aid in the discussion below.)

**Step 0 is Complete When:** The goals and perhaps even deliverables for your various subsystems have been defined.

---

**Step 1:** Set up the Interface Matrix by giving each subsystem its own sheet in the Excel Workbook. Begin by list the names of all of the subsystems along the top of the spreadsheet, each in their own column. The sample InterfaceMatrixBasicSample.xlsx file only shows 3 subsystems, but more can easily be added.

Create a copy of this sheet for each subsystem in the Excel workbook and name each sheet after one of the subsystems. Add a title to the top of each sheet for the associated subsystem. Each sheet will now represent the perspective of this subsystem and its relation to the rest of your system.

**Step 1 is Complete When:** Each subsystem has its own sheet in the Excel workbook for your overall Interface Matrix.

---

**IMPORTANT NOTE:**
The rest of the steps will discuss only one sheet in the Excel workbook but the steps should be mirrored for all subsystems' sheets in the Excel workbook. The subsystem whose sheet we are working on will be referred to as Subsystem A.

---

**Step 2:** In the column after all of the subsystems names, make a list of all of the aspects of this subsystem (Subsystem A) that may influence other subsystems. This is best done first by having the teammates working on Subsystem A take their first guess as to what other subsystems may need. After all they probably understand their own subsystem best.

In the InterfaceMatrixBasicSample.xlsx file, a motor selection subteam may begin by adding the following information about their motor that the other teams may need to know: motor voltage, motor current, power rating, maximum duration usage, weight, cost, dimensions, etc. (Please see the ModBot "How to Spec a Motor Guide" for information that is pertinent). Notice that the information may be listed in a kind of hierarchical format to make it easier to read and find the info you're looking for quickly.

**Step 2 is Complete When:** a list of all of the information that Subsystem A has control over, and that other subsystems might need to know, has been added to the Subsystem A spreadsheet. The values of the information, i.e. column R in the sample file, do not need to be filled in yet. You only need to list that this information will eventually be provided.

---

**Step 3:** For each piece of information listed about Subsystem A, determine which other subsystems may need this information. Mark the cell in the Subsystem A information's row and corresponding needing subsystem's column with the works "provide to".

This indicates that the people working on Subsystem A will provide these other subsystems with this information i.e. row 8 reads "The Motor Drive Subsystem (Subsystem A) will provide the motor power rating to the Power Subsystem". It also serves as a reminder that when any changes are made to this value, the subsystems marked "provided to" should at least be informed of the change.

**Step 3 is Complete When:** Each piece of information listed is marked as being "provided to" to at least one other subsystem. An examples of acceptable exceptions to this rule is in the Motor Drive subsystem sheet of the InterfaceMatrixBasicSample.xlsx file rows 5, 11, and 14 which are essentially header rows for the rows immediately below them.

The wording "provided to" can also be abbreviated or changed if it is more convenient. The important thing is that the idea is made clear and that abbreviation or change is made consistently throughout the entire interface matrix (Excel workbook).

---

**Step 4:** Discuss with the people developing other subsystems whether the information you (Subsystem A) are providing will meet their needs and whether they can provide the information that Subsystem A needs.

Quite often it is discovered in this step that the information you are providing may not be complete enough or in the form must useful to another subsystem or vice versa. This discussion is crucial for both helping to better define the kind of information that needs to be shared but to better determine the needs of other subsystems and their relative importance.

**Step 4 is Complete When:** All people working on all related subsystems will be receiving the information they need for their own subsystem and it Is clearly listed in the interface matrix with the proper "provided to" cells filled in.

---

**Example of the Importance of Step 4: Incomplete Interface Specifications:**
Discussing what kind of information will be provided ***in detail*** is very important as assumptions on how general needs will be met is often the cause interface failures. Take for example the motor drive subsystem and the power subsystem. The motor drive system may be worked on by primarily mechanical engineers and the power subsystem worked on by electrical engineers. If they weren't using an interface matrix, the power subsystem might simply say "We need the power information about the motors".

Having some basic understanding of electronics the motor drive subsystem mechanical engineers interpret this as meaning "The power subsystem needs to know the motor's operating voltage, and normal operating current" and in return ask the power subsystem what ranges they should shoot for, which the power subsystem happily supplies.

At this point everyone *feels* that they have either received or supplied all of the information they will need to move forward. The motor subsystem then goes about their decision making process

selecting a motor that meets their own needs while having their normal operating voltage and current point being within the range given. The motors are then ordered.

Weeks later and significant dollars invested, the motors arrive. Testing begins only to find that once the motors are connected to the power system, the power subsystem board is almost instantly fried. What happened? The team discovers that during start-up the motor's stall current causes a significant current spike that well exceeds the power subsystem's current limits and the following argument ensues:

Power subsystem:
"What the heck? The motor output current spiked at a value that way exceeded the specs!"

Motor drive subsystem:
"No. That motor should definitely work. You said we just needed to give you the motor's power specs. So we did and they were within the range you gave"

Power subsystem:
"Yes, and we gave the max current and voltage the power system could handle. You must have spec'd only the normal operating current. You have to worry about the start-up current too."

Motor drive subsystem:
"You never asked for that."

Power subsystem:
"We said we needed motor's power information. Start-up spikes are obviously a part of the power information"

Motor drive subsystem:
"Hey when we asked you for a range, you gave us the voltage and current and that's what it says this motor is within right here on the motor spec sheet"

Power subsystem:
"You have to look at *stall* current value on the spec sheet too."

Motor drive subsystem:
"Well you never said *stall* current. I don't know what that is. So we thought we could just ignore it. I mean you didn't ask for it. You just said these are the ranges and that's what we spec'd to"

Power subsystem:
"Well it should be obvious."

Motor drive subsystem:
"Well the cost for the new motors is coming out of your budget."

Power subsystem:
"No you're the one who picked the wrong motors! And who's going to pay for a new power board?"

Motor drive subsystem:
"Well it obviously didn't work anyway."

Power subsystem:
"It worked fine on its own! You just don't know how to pick a motor!"

Motor drive subsystem:
"Dude this motor will work perfect for our stuff! You just didn't tell us what you really needed! And now it will take forever to get new motors in. That's it. You just need to make these motors work…"

Power subsystem:
"Says who?"

Motor drive subsystem:
"Says the budget. Says the timeline…"

Power subsystem:
"Well we'll have to recreate our entire board, so how does that enter into the budget and timeline?!"

Motor drive subsystem:
"Fine! We'll settle this with a Halo match. First one to 15 kills wins."

Power subsystem:
"Oh, it's on noob."

And in the end both say:
"Why do I have to work with such idiots…"

Of course the answer to who's at fault is…. Everyone. The power subsystem should have exactly specified all of the information they needed and not assumed that another group would be able to ascertain what all of their needs are. The motor subsystem should have checked with the power subsystem that the motor they intended to order met the power subsystem needs as well. Even if the power subsystem didn't specify the stall current, or perhaps weren't even aware that motors had a stall current, and the power subsystem still thought they were very thorough, the motor drive subsystem should not simply ignore any found information, such as information on a spec sheet, that they believe might be significant to another group. If you do bring other information to another group's attention, they still might tell you it's superfluous, but the few minutes investment was worth it for everyone.

The end of this example is also meant to be a bit comical but to also make the point. When other subsystem teams do not understand the importance and needs of various subsystems, it often leads to trade-off decisions being made in rather "uninformed" manner. More on a means for helping to track changes and make decisions regarding interfaces will be discussed in Step 7.

---

**Step 5:** Enter in the values of the information provided by your subsystem. It is usually best to write the value across two columns, one for the numerical value and one for the units (columns F & G in the InterfaceMatrixBasicSample.xlsx file) which helps to emphasize the units and prevent conversion errors.

In some cases, a single cell will not suffice for the kind of information that needs to be communicated. Some examples might be that the motor drive subsystem needs to provide installation instructions, or the power subsystem needs to provide a start-up procedure, or a communication subsystem needs to provide a protocol.

In these cases, instead of entering a numerical value, the location of where the information can be found should be listed instead, say perhaps the file name of a Word doc located on a team accessible repository, or the section and page number of a manual. With files in particular, if the location of the file is not absolutely obvious is it good to list where the information can be found.

This can greatly increase the value of the Interface Matrix as a communication tool for your team. If perhaps someone from the power system subsystem team is not around, and yet the rest of the team needs to be able to start-up the system in order to continue to do work, the rest of the team now knows where to look for the most up to date information.

**Step 5 Is Complete When:** All of the current known values are entered in the Interface Matrix, including both the unit for numerical values, and the locations for files.

---

**Step 6:** For all of the unknown pieces of information, write in dates that those values will be determined (column O). This should relate to your timeline and is important to help other people working on different subsystems know when information will be available. Therefore it is important to discuss with the teammates of the various involved subsystems as to what dates may be acceptable.

Many times these discussions may lead to adjustments in the timeline. As discussed in the timeline guide, it can also be worthwhile to provide estimates that other subsystems can use to work from for the time being. Hence there is also a column to indicate the next estimate date (column N) and a column to indicate whether the value is an estimate or final actual value (column H).

**Step 6 Is Complete When:** All unknown pieces of information have due dates, and possibly estimate dates. The team has agreed to these dates and the team's timeline has been adjusted to ensure these due dates will be met.

---

**Step 7:** Establish an "interface champion" for each piece of information. Although the name "interface champion" may seem a bit corny, this is the professional name commonly given to the person who must give their okay any time a change is made to an interface value. This is a very important role as it is their responsibility to check how any change could influence the subsystems requiring that information and then decide whether to permit the change or not.

Granted sometimes changes are unavoidable (for example, a part that was needed is no longer being made by the supplier), but in most cases it's a question as to whether the "extra" work incurred by a change is worth it and it is the interface champion who can hold the final word as to whether it is or not.

These decisions are commonly based upon how will the change effect the overall performance of the system, the timeline, and the available resources. The important thing to remember though is that the interface champion must okay the change ***before*** the change is made.

Interface champions are not always a project lead but are typically one of the people either working on the subsystem providing the information (supplier driven information), or on the subsystem receiving the information (client driven information). If the information being supplied can have significant effects on the receiving subsystem it is common to have the interface champion be on the receiving (client) side to ensure that subsystem's needs are being represented fairly.

**Step 7 is Complete When:** All pieces of information have been assigned an interface champion.

---

**Example of Incorporating Interface Champions: The Working and Testing in Isolation Issue**

Interface champions are also often involved in the testing of subsystems, or at least in the signing off that a test has been passed successfully. As an example, consider the communications subsystem and the sensor I/O subsystem example from the beginning. Here the communications subsystem team developed a message protocol and test messages that they had created themselves. The first issue that can result if they didn't follow an interface matrix technique, is that the test messages may work for the protocol they design but that protocol design may not work for the kinds of messages the sensor I/O subsystem needs to send. Furthermore, even if the communication subsystem followed all of the requirements given to them by the sensor I/O subsystem team, it is still possible that the communications subsystem tests they created did not accurately reflect the way it was planned on being used by the sensor I/O subsystem.

By having an interface champion involved with the testing, and in this case even in verifying that the devised testing procedure would meet everyone's needs, the problems could be averted, or at least reduced. This doesn't mean that the interface champion has to be involved with every step of the test development but at least having the interface champion take a quick look at the test procedure is again a wise investment.

---

**Step 8:** Add "last updated date" and "last updated by" columns to the interface matrix. This is valuable as a log to see when changes were made but also who the change came from, and therefore who should be asked about the change if there are any questions farther down the line.

This becomes particularly valuable for large teams or when several people are working on the same subsystem.

**Step 8 is Complete When:** All pieces of information with values also have a lasted updated date and a last updated by teammate name listed.

---

**Advanced Techniques**
At this point the Interface Matrix is complete enough to use. Below are 3 additional steps that you can take to enhance your interface matrix further. You can see these additional steps applied in the InterfaceMatrixAdvSample.xlsx file.

---

**Step 9: (Optional)** Add a column called “True Budget Total”. This column is used to provide the sum (or other combination) of values from multiple pieces of information. In the example in the beginning, the chassis design system is responsible for providing a total weight limit for the overall system. Therefore each subsystem must provide the chassis design subsystem with their individual weights. As the requirement is based upon the totals of these amounts, column E is added and the individual cells within this column should be filled in with equations to represent this sum.

This approach can also sometimes be used to keep track of some performance metrics for the overall system.

Individual limits are often set for each subsystem as well. This is often referred to as “allocating” the resource budget to various subsystems. Often it is wise to include at least an allocation estimate for each of the subsystems to work with early on in the design, in order to help identify potential violations early on.

**Step 9 is Complete When:** The equations for any budget totals or other combinations have been entered into the interface matrix and any needed allocation estimates have been entered as information provided to the other subsystems.

---

**Step 10: (Optional, To be Combined with Step 11)** Add to each subsystems list of information, i.e. column R, any pieces of information you expect Subsystem A would need from other subsystems. It is best to signify somehow that this information is not coming from Subsystem A but must be given to Subsystem A. In the example interface matrix, information being received from another subsystem is listed in *italics*. You may enter this information via Excel equations that refer back to the name written in the information supplying subsystems so that you do not need to rewrite the name and if the name is later changed, all sheets will be automatically updated.

In each newly added piece of information’s row, mark the corresponding cell in the column of the subsystem that will be providing that piece of information with the phrase “received from”.

The advantage of this step is to make it easier to see how different interface values relate to each other; for example, the power system can easily check what are the power requirements/usage levels of all components in the system. Likewise the chassis design subsystem can easily see where all of the cost budget and weight budget is being spent.

**Step 10 is Complete When:** a list of all of the information that Subsystem A needs from other subsystems has been added to the Subsystem A spreadsheet and the corresponding cells have been marked “received from. The values of the information, i.e. columns F & G in the sample file, will be filled in in Step 11.
**Step 11: (Optional, To be Combined with Step 10)** Enter the values of the information being received from other subsystems, by using Excel equations to refer to sheets, i.e. in the Power

System Subsystem sheet, cell F6 where it says the motor voltage, is actually an equation that relates back to the Motor Drive Subsystem sheet, because the Motor Drive Subsystem is responsible for providing the motor voltage information.

**Step 11 is Complete When:** Not only the numerical value but the values in columns E thru O for all newly added rows for information received from other subsystems filled in with equations that refer back to the supplying subsystems Excel sheet.