The IoT Development and Testing

Part 2: Dev, Ops, and a Little Testing

Jon Duncan Hagar

2017

Publish - LeanPub

Tweet this eBook

This is an online eBook. We do not plan a published hardcopy. It is part of a series where we are keeping costs low to make it affordable to everyone. Early versions will change frequently, and later we will do updates as needed. So, we are always looking for comments and inputs. You can email or tweet these to us. We also hope you will post on social media about it.

Tweet: jonduncanhagar

Linkedin name: jonduncanhagar

Email: jon.d.hagar@gmail

Release Notes

This version is 80+% complete.

Preface

We offer this series of eBooks to help teams moving into IoT development and testing. The focus is on testing, but in IoT we believe teams should not separate development and testing efforts to far from each other. Our experience is that IoT teams must consider the hardware, software, system, and operations to be a great success. Further, teams coming from a particular engineering environment will tend to focus on the familiar and have larger learning curves in other areas.

We have structured this series of eBooks to address these considerations and be cost effective. So rather that one big expensive book, we opted to create small eBooks, follow an Agile approach, keep costs low, put materials online (eBooks), and seek user feedback. To these ends, readers can buy the full bundle or opt for just one or two eBooks. The selection of eBooks depends on the reader's context.

Keeping with the Agile philosophy, we plan ongoing updates, so please contact with ideas, improvements, and what you like. We also hope to have online or traditional training sessions using the eBooks. The exact format of the training is yet to be determined, but again, our hope is to keep costs and time minimized. We are looking to help and learn about IoT as this part of the industry grows.

The IoT Test eBook have the following structure to help readers:

- Part 1: what is IoT, what are testing and development along with top level concepts

- Part 2: IoT Development and Test – A general high level introduction for teams starting into IoT development and test, containing references to more details.

- Part 3: Iot test planning and strategy – A basic introduction and starting point for test planning covering from the informal to formal levels. This is a start for beginners and contains tidbits for the experienced test manager.

- Part 4: IoT test design and security – An eBook that contains test design and implementation details specific to the IoT environment. Test levels move from white box to black box testing as well as non-functional types of test. Part 4 is a guide for teams doing tests in the small start-ups to the larger scale high risk IoT systems.

- Part 5: IoT environments and tools – This part is a necessary addition to the test design of part 4. Part 5 addresses test labs, automation, domains, and support tools.

Finally, we expect that readers will have some knowledge about development and testing. No effective book can address all aspects of technology. We will provide reference, both traditional hard copy books, and online resources. To grow in knowledge and skill, readers must have a large library of reference materials. We have such large libraries ourselves, and this is one reason why some people regard us as experts. We do not know everything, but we know when and how to go about looking materials up. A definition of expert that we like is, "An expert is a person who knows what they don't know as well as how to go about learning the unknown". To this, readers should regard what is said in the eBook, and for that matter many other references, with degrees of skeptics and seek confirmation by experience and testing. Skepticism is should be second nature to development and test engineers. A saying comes to mind “trust but verify” (ref https://en.wikipedia.org/wiki/Trust,_but_verify).

Acknowledgements

We would like to thank and recognize our many mentors, colleagues, and teachers, who have taught us so much through the years. There are many ideas in this book from nearly all of them, which we try to correctly cite throughout the book. We are all standing on each other’s shoulders. Our largest acknowledgement goes to our reviewers and Jon’s wife, Laura—a software–systems geek and head reviewer. Without them, this book would not be possible. With all of these people, the references in this book and hopefully, the book itself, I hope that the IoT device world becomes safer, more secure and fun to use.

We particularly would like to thank the following people who over the years have help and made us think:

Cem Kaner

James Bach

Lisa Crispin

Becky Fielder

Laura Hagar

About the Author:

Jon Hagar is a senior tester with 35 years in software development and testing currently working with Grand Software Testing, LLC. He has supported software product design, integrity, integration, reliability, measurement, verification, validation, and testing on a variety of projects and software domains (environments). He has an M.S. degree in Computer Science with specialization in Software Engineering and Testing from Colorado State University and a B.S. Degree in Math with specialization in Civil Engineering and Software from Metropolitan State University of Denver, Colorado. Jon has worked in business analysis, systems, and software engineering areas, specializing in testing, verification and validation. Projects, he has supported include the domains of embedded, mobile devices, IoT, and PC/IT systems as well as test lab and tool development.

Introduction – What is IoT?

Each eBook in this series starts with an introduction to the environment of Internet of Things (IoT). Some people also refer to this as the internet of everything, but I will stick with IoT.

Why the internet of things?

Well in IoT the basic idea is to take every day physical objects, devices, systems, and systems of systems and connect them with software as well as communications to the internet. The obvious starting point has been existing electronic devices since these objects already had power, which is necessary to support computers and communication. However, many people have realized that other physical devices can have power and a computer added offering "new" smart devices.

Gartner's Nick Jones, vice president and distinguished analyst "The IoT demands an extensive range of new technologies and skills that many organizations have yet to master," he added "A recurring theme in the IoT space is the immaturity of technologies and services and of the vendors providing them. Architecting for this immaturity and managing the risk it creates will be a key challenge for organizations exploiting the IoT. In many technology areas, lack of skills will also pose significant challenges."

Today there is a lot of excitement and hype about IoT. Weather I see a few billion or billions of billions of devices, IoT will happen.

https://media.licdn.com/mpr/mpr/p/7/005/081/13a/317cb86.jpg

Link 1: Tech Growth Curve

We will have IoT smart devices in our home, cities, offices, cars, governments, world and about everywhere one can image (see eBook 1 for more details). I will hear about smart homes, smarter cars that drive themselves, cyborg humans, work places where most physical objects are monitored, and intelligent city governments (an oxymoron if ever I heard one).

We will not debate the many moral, ethical, legal, and human concerns of IoT. There are many, but what this series of eBooks offers is engineering considerations to help developers and particularly testers create smart objects "better".

What This Book Addresses

This book provides is starting point development which include testing and operations information on IoT. To avoid being water fall based, the order is "nonstandard". It is not a detailed text, but includes:

- Planning
- Quality, Verification, Validation, and Testing
- Operations (Ops)
- The traditional life cycles
    - System considerations
    - Hardware design (very brief)
    - Software design (even more brief)

        - Communication (Comm) and integration (only key points)
- Product life cycle considerations
- Summary

Many readers will be familiar with what is called “dev-ops” organizations. This book plays on the Dev-Ops approach to systems, but per se this is not a book strictly on Dev-Ops. I believe that due to nature of IoT many organization will follow a Dev-Ops (ref) approach. But pure Dev-Ops, if there is such a thing, is not strictly necessary for IoT.

The reason I think Dev-Ops will play a role in many IoT projects is the nature of IoT where I have “smart” physical object which feed data via communication channels during their usage life. So, I must think about both development and operations when building IoT systems. And for us, testing is a major success factor in development to assure viable operations. Further, operations will need feedback into testing and development. This feedback cycle using large amounts of real world physical IoT data is a separating difference between traditional physical devices and IT systems.

Readers organizations will need to address these differences when developing and operating IoT. Many hardware organization are used to putting devices out into the wild while not having a lot of feedback from the device. IoT will change this. Many IT software organization are used to system which operate primarily in cyber space with limited physical impacts. IoT hardware Ops will change this. Companies and organizations that adapt will do well. I hope this eBook opens up thinking and conversation within organization moving into IoT.

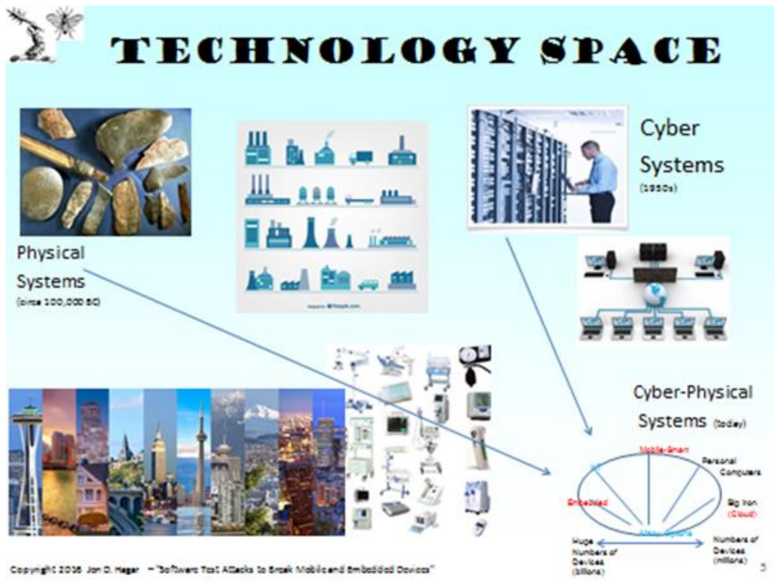


Figure: Picture of physical-cyber evolution
Reference: Jon Hagar Built figure for IoT Classes 2016

The book is intended to be used with the eBooks series as well as other books on assessment and development. I recommend that the reader have several of the books in the references section as no single book can address all aspects of development (dev), operations (ops), testing, Verification and Validation (V&V) for IoT. Hence much of this book consists of basic ideals and pointers to the details. A reader not familiar with an ideal should consult the detailed references.

Audience

Dev-Ops-Testing are large subjects and no single book or reference can be complete. With over 70 years' experience between us, I am still learning new testing skills. Further, readers should keep in mind that there is no best or one path to do dev-ops-testing. There are many options; each has positives, negatives, as well as cost and schedule impacts.

This eBook is written for organizations and people that are new to IoT Development (Dev) and/or Dev-Ops. The IoT world is expected to be valued in the trillions of US dollars over the next 5 years and tens (100s?) of billions of devices (ref Garnter report). Almost every traditional company will enter IoT and there will hundreds of new start-ups. I hope these following kinds of organizations and people will find useful information in this eBook, including:

- Companies, who have never developed software, but want to expand into IoT within their existing product lines. These might include: industrial/medical companies (e.g., medical devices, health monitoring systems, heating control systems, transportation systems, etc.) and consumer companies (e.g., wearables, clothing, home entertainment, etc.).

- Companies who have experience in electronics and limited aspects of software but are looking to expand their software footprint while lacking networked software testing background (e.g., television, audiovisual devices, automotive, etc.).

- Startup companies looking for a good beginning reference into IoT development and testing.

- People looking to learn more about IoT Software Dev-Ops-testing to enhance their careers.

- Groups and government officials looking to define IoT regulations.

- Anyone that is interested in IoT but lacks basic understanding.

How to use this eBook

The eBook can be skimmed or read quickly end–to-end. After an initial reading, sections that were not clear at first can be read in detail. I do not intend the book be read in detail from cover to cover, but by jumping around to topics of interest in IoT

The eBook itself is in some sense written in a front-end-reverse order. What I mean I start with general planning and funding since if you don't have money and time everything else is less important. However, to avoid looking like a water fall, which most books and standards do, I then jump to

V&V/testing, then work "backwards" through a lifecycle ending at architecture, but if you are skipping around, this won't bother you.

The table of contents can also be used to index in to topics that are of interest to readers. I suggest the reader assess if they understand basic concepts found in Chapter 1 and 2. If a reader determines they understand the basics then use the index or table of contents to find specific topic of interest. I would suggest as testing of an IoT device and system progress that testers refer back to the book. I also request if a reader finds things missing or lacking that you contact us, as I plan revisions to the eBook rapidly following Agile concepts.

Planning and Funding IoT Projects

This is an area where I have mixed levels of experience since it is a wide topic. I have helped plan many projects. Funding a project is different than planning and something I have partially done. To be complete, I do have pointers on how to do a key aspect of funding which includes budget estimates, but there are aspects such as obtaining capital for start-ups where I have limited experience. I think many IoT people and projects would like what information I do have on planning, budgets and funding. I certainly recommend following some of the links and looking for additional information to get development (Dev) started.

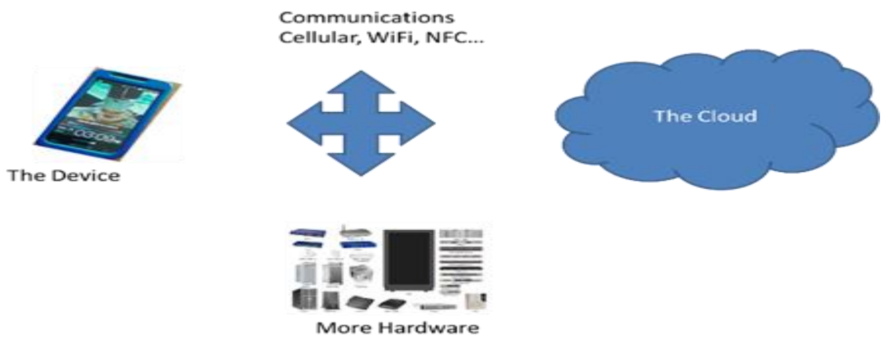


Figure a typical IoT dev and test landscape – go beyond just the single IoT device
Reference: Jon Hagar Built figure for IoT Classes 2016 redo pictures

We will consider IoT efforts from the small size, which is what many people first think, to larger IoT efforts. I cannot cover all aspects, options, and sizes as the space is infinite, but hope to expand as time goes on.

Note: this planning section repeats some of the info of book 1, but address some items in more depth.

Creating a small, or a smallish "critical", device IoT system (for the money)?

Many IoT devices will be "small" at first. Teams will be trying to create them on minimal budgets and planning. This aspect of IoT can be likened to the "gold rush" of the wild-wild west. People want to strike it rich with making a small IoT device that they can sell and make money. Examples IoT devices are shown in part 1 of the eBook series.

Many IoT efforts will not hit gold, but some will. Small IoT devices may not take huge amounts of capital (see below) to go into development but creating any set of hardware and software will take effort. In this sub section, I list the items to include in your considerations in table 1. Medium size devices will take more time, effort and money. Likewise, larger IoT systems will be bigger (see next section).

Table: Example Funding Areas to Consider

Device Parts	**Sub item**	**Timing**	**Cost**
	Hardware Development - a	Early	$ to $$$$
	Software Development - b	Early	$$
	Supporting Software Procurement	Early	$
	Interface items (cloud, network, etc)	Mid	$
	Production (manufacturing)	Mid	$ to $$$$
Supporting Items			
	Shipping (to/from a country like China) including customs	as needed	$$
	Copyright, Patents and legal	Early	$$
	Customer shipping	Late	$
	Marketing	Late	$ to $$$
	Management	constant	$$
	Travel	constant	$$
	Deployment (funders, online stores, and ?)	late	$
	Support functions- c	constant	varies

Table: Example Funding Areas to Consider – tbd remove first

Special notes for the table include:

- Hardware include the computer, sensors, and controllers. Be as generic as possible, but some customization will like happen. Computer hardware come with issues of size, speed, memory, power A2D and D2A (ref my book?

- Software makes IoT devices “smart” and separate IoT from other devices. The device IoT software may be small. It will be 1000s of lines of code. The software for App will likely be a mix of custom and off the shelf. It will be larger 100,000s or more of lines of code. The software area that may be tempting to forget is what I label as “interface” software. You may or may not be reasonable for this software in total. This is software that glues the device and the App together with the rest of internet. Big data and data analytics will be a big part of IoT. I forget this part of IoT at risk of a device failure. Finally, software may also need to handle the communications. This can be third part, but your device needs to work with this software. The total volume of software grows and become bigger over time.

- Support functions are dependent on the device and organization. The functions can be small to large

Many small start-ups will be under pressure to produce a “sample first” device. How much hardware and software do you need for this first version?

Well this depends on what you are doing. I recommend the KISS rule (keep it simple stupid). Get simple success and then expand if you get more funding to live another day. This first version may go to the funding sources such as the cloud or venture capitalists. For this first version is better to hit what is important, undersell what is important, remember things always take longer than you plan and cost more than you want.

Finally imagine you have some first version hardware-software (some software on a prototype wire rap version of hardware) and enough features to impress investors to the point where you can you think you can move stage two in product life. You are done and a success, right?

Will maybe. Remember undersell version 1.

We know their items than I list on the table, but these are the ones I have had to work in the past and/or have had experience with. It is not enough to just create a cool IoT device, even a small one that is “perfect”. You have to create a cool IoT device that work just “well enough” to continue to the next lifecycle stages and revision developments.

We are talking a device made up of hardware and software. This is different than a web page, or IT software people can load easily. You must produce some amounts of hardware. Say you had a Chinese production facility. You should not share your software with the production facility. They will “steal” it. Worse you may need more than one production facility in china to avoid “stealing” of the hardware. This means you will need some “production” (glue the parts and software together) in the USA. At a minimum, you will need to load the software you own onto the hardware IoT device. If you have the “some assembly required” configuration in the USA, this protects you but requires work and cost here.

Great, got it. I’m done, right? Well not totally.

If you have the support App software, which most IoT will have and/or integration-analytics software, you may need to have these available for the first users. This may mean you may need to meet Apple iTunes and Android App store rules. This takes more time and money. It does not happen overnight. You may need to travel to talk with the lawyers about copyright and pay them more money. Even worse where is your integration-analytics software going to be hosted and run? Who owns and pays for that.

Finally, even for a small IoT device-system, all of this may need some level of testing before you ship to any users or investors. You want the device-system to work just "good enough"

Scared?

It will be a challenge, but some creators will raise to the challenge.

Note: if you have standards, e.g. IEEE, Auto, MISRE, ISO, FAA, FDA, etc. with which you must comply, the funding profile must include addressing these (see appendix TBD for partial list). In this book, I consider and include some of these, but many of these standards are historic and "dated". It possible to move IoT forward to be "state of the art" without standards. No standard is state of art, and so they all look backwards, but a project still may need to factor them in the funding efforts. IoT is moving fast, so I must in some cases have both the back-ground history (the standards) and state of art when doing the development. This is a challenge to engineering.

In the Beginning – A Inspired Spark of an Idea for smallish start ups

What are the actions a small IoT team may want to plan and do?

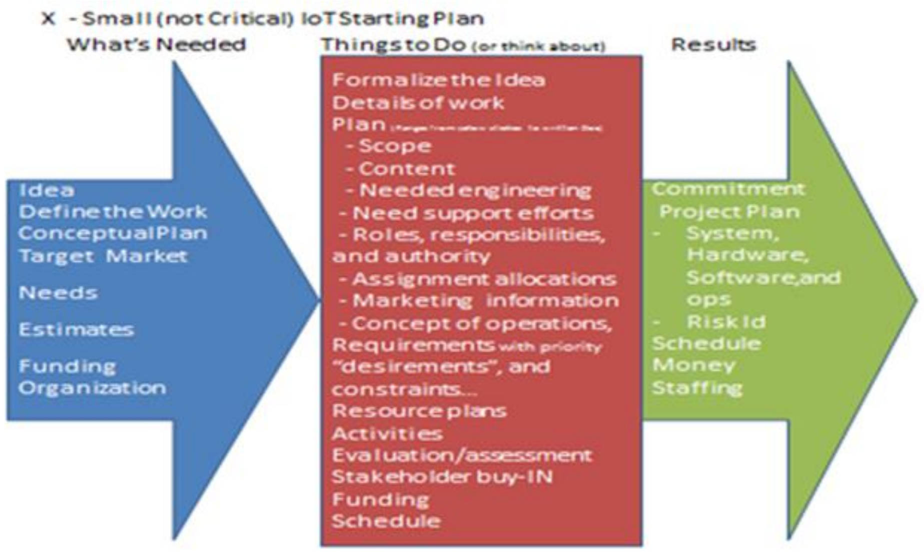


Figure: Planning Smaller IoT
Reference: Jon Hagar Built figure for IoT Classes 2016

Note: I hope these figures are more useful that a lot of text, so I don't explain every item in the figure when I think they don't need it. You can always drop us a line with a question.

There are several things in the figure that may raise questions to the reader.

First Inputs:

a) The idea – needs to be good and something you can "sell" to the funding organization. Along with the idea is the "need", which you can think of as the justification of why this idea will "sell" to an identified target market.

b) Defined work and concept plan – you need some basic idea of effort and what needs to be done to explain this to the funding organization.

c) Funding organization – where are you going to go to get the money (see below)

Results:

a) The project needs commitment of the funding, the team, stakeholders, staff, and schedule. If people don't believe it can be done, it likely will not get done

b) The plan needs to be at least in part written down. This may be on a wall of yellow stickies, a mind map, or a working document, but the act of writing things down and getting others to review it solidifies our thinking

c) Requirements, which may be contained in the plan or separate information, need to be sufficient to support the project commitment and planning, but they will not be perfect. Expect them to evolve. Also at the beginning cycle, the team should under commit to the requirements. Commit the minimum. Finally, requirements may be formal shall statements, stories, function points, pictures on a wall, or other notations, but again, writing things down helps the thought process. See Section Defining needs (stories or requirements) for more information.

Are you responsible for the larger IoT landscape?

The first example above assumed a small start-up team. I suspect many more IoT devices will be created and produced within larger company organization. Here there will be "corporate" goals and constraints to consider. I will have "internal" start up efforts or evolution of existing product lines into the IoT space.

We already see this in partnerships between tech companies and traditional companies, e.g. IT giants and automotive companies. I know of companies producing stereo loud speakers moving into software, the apps, and then IoT. Pretty much every company that has electricity in the products will move into IoT sooner or later. Further companies that are producing "traditional" no electric products will move into IoT either directly or by buying a small start-up. I see this already in IoT shoes, clothing, household items and you name it.

So, the list of the above table and section above applies here too. However, to the table, I would add many items including:

- Companies are doing partnering with other companies because each realizes the IoT has elements they are "weak" in. These organization may need less of this book, but can send us ideas for reversion 2

- Companies are “hiring” talent who claim to know area the companies believe they are weak in. However traditionally in “high tech” good talent is hard to find and there are a lot of people who can’t deliver

- Companies are “growing” high tech talent because there is a “skills” gap in the work place. This people may find these eBooks of some use

- Companies are hiring contract workers who come with the skills needed.

- Companies are hiring foreign H1B workers because American workers do not have the needed skills.

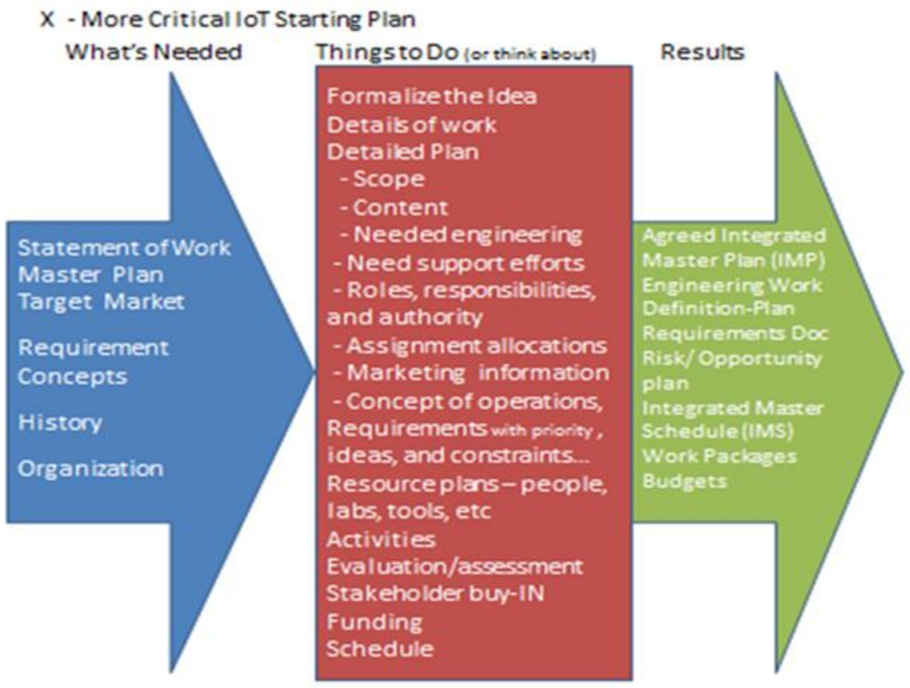
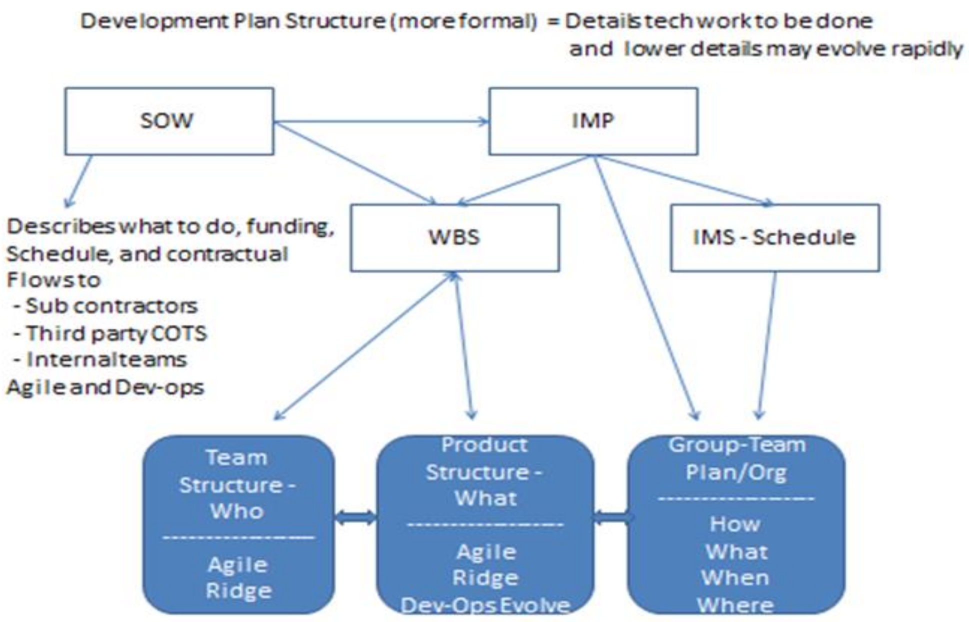


Figure: Planning Bigger IoT Device Efforts
Reference: Jon Hagar Built figure for IoT Classes 2016

Most established companies have business practices. The IoT team will need to fit within this structure. Figure represent the most basic set of needs and results. The practice will likely be a little more formal with statements of work, plans, requirements (written), and organizational practice based in history.

The large company IoT team will likely face more “justification” for resources (company money, time, staff, facilities, etc.). Our experience some organization are very risk adverse (meaning they are not likely to take on new IoT efforts without a lot of justification) while others are very open minded (meaning if you have a good story with ideas like in the small start-ups and meet enough of the organizational expectations you’ll get a “go”).

The result will typically be things such an IMP, Statements of work (SOW), Integrated Master Schedule (IMS), work breakdown schedule (WBS), funding, plans, and more detailed requirements.   These are shown in figure Standard Big Company Planning-Funding Profile. All of these are likely to change during the life cycle in any case. The names and levels of formality of these will be organizationally dependent, but the basic ideas are likely to be there, so become familiar with them. For more information see TBD link.



Development Plan Structure (more formal) = Details tech work to be done and lower details may evolve rapidly
SOW
IMP
WBS
IMS - Schedule
Describes what to do, funding, Schedule, and contractual Flows to
- Sub contractors
- Third party COTS
- Internal teams
Agile and Dev-ops
Team Structure - Who
Agile Ridge
Product Structure - What
Agile Ridge Dev-Ops Evolve
Group-Team Plan/Org
How What When Where

F*igure: Standard Big Company Planning-Funding Profile*
Reference: Jon Hagar Built figure for IoT Classes 2016

*Fu*nding-Planning Options include:

- Planning poker

- Yellow sickies

- Project planning tools, e.g. PERT, Gant, etc.

- Contracts

Defining needs for the IoT device (AKA stories or requirements)

There are three classic approaches you can use to elicit user needs or requirements. These are as follows:

- Interviewing – ask people what they want. People can include users, experts, stakeholders, completing products that you think you can do better, etc.

- Remember to ask the right questions because garbage in = garbage out.
- Definition from experts (this may be you) with history and/or experience
- When dealing with complete products, it is fair to get the device and "use it", but be careful with doing too much reverse engineering as the may trade/copyright issues.
- Modeling – model the new or existing system to determine its uses. Include changes to show the stakeholder how things might be improved.
- Observation or shadowing – watching current users and what they are doing, not doing, what can be better, and what is missing. This works well when moving existing products into the IoT space.

Needs can be stories if you are Agile. Needs can be shall statements, if you are more traditional. Most of us will mix stories and shall statements to get our needs written and defined.

Once you have some needs defined, it is best if I start some work (see later sections). I learn more by creating and using things the spending all our time in thinking exercises.

Agile = prioritize and work on important needs first (see agile principles)

Defining the "Idea" and top-level goal. The product must have the top need-goal to be marketed. The IoT device needs to fulfil a customer need. This idea must be conceptualized. I call these things concepts of operations, stories, use cases, and other names. It is best to have a picture (for example see mission picture). It is also good to have an Concept of operations (Con-Ops), and depending on the criticality of your device you may want some modeling, market studies (interviews), and even demos obtained by shadowing users.

TBD – Define more with pointers the difference between classic requirements and Agile stories – TBD refs

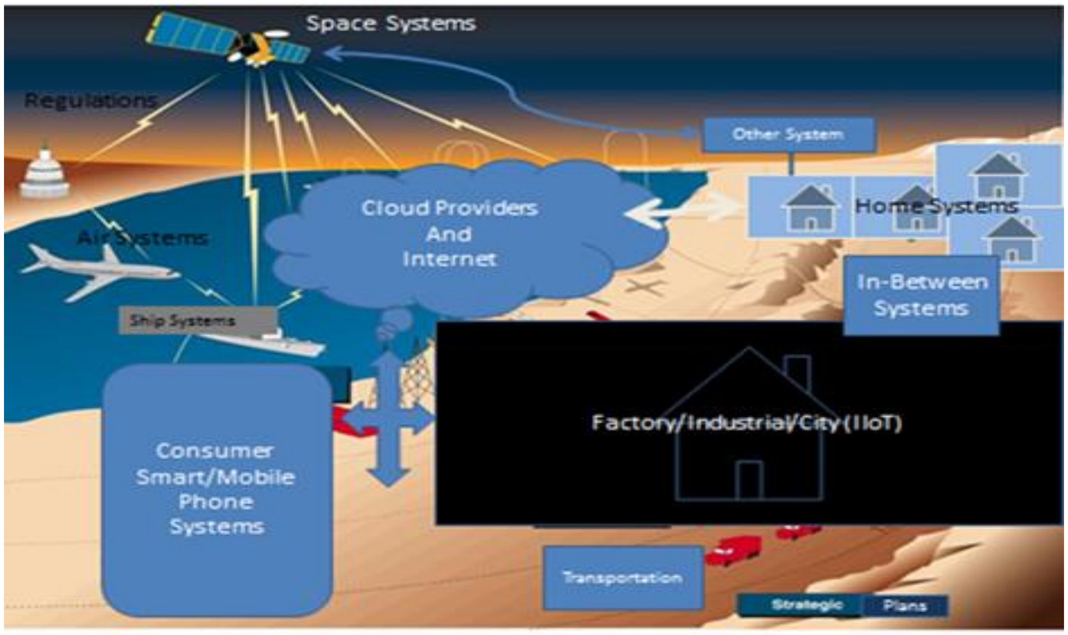


Figure the IoT-Mobile-Comm-Cloud landscape
Reference: Jon Hagar Built figure for IoT Classes 2016

So, what is the role for V&V/tester at this early stage maybe before I even have the "okay" to proceed ahead? I well consider this story.

A Story:

A project was producing a "one of a kind" prototype. There was hardware. There were operations. There was software with communications to the ops and hardware. I create the prototype of a shoestring budget. It worked some of the time. Developers were able to show the product the stakeholders and potential customer who said "cool" you have hardware and software. However, the team using the test information I had from running the prototype knew I had timing, communication, and performance issues that would require new "faster" hardware, comm, and optimized software. Knowing this during the "selling" stage became important once I won, because I know the con-ops I had to "fix" first.

Stories and/or Con – Ops (concept of operations)

The best way to understand the "needs" of a new device is to have stories, also known as Con-Ops (if you are a traditionalist). Areas to address step-by-step activities include:

- Use or Operation of the device, network, and/or system by many different kinds of users, both human and non-human
- Manufacture, production, and distribution (hardware and software)
- Test including development, V&V, independence, integrity, risk of software, hardware, and/or system
- Installation or De-installation of the hardware, software, and/or system
- User actions to learn, obtain help, and/or training
- Storage in the device, local, Fog, and Cloud
- Security and privacy usages
- Maintenance cycles for hardware and software including dev, regression, and ops
- Modification or Upgrade options
- Security impact
- Disposal at end of life (hardware concern for some)
- Normal and Abnormal Use Cases and modeling for Dev and testing (most people think normal first, but non-normal uses usually end up out numbers normal)

Each of these will be one and usually more stories or Con-Ops. I can build these into models. I can build these into formal documents (if you like such things). I will start small (a few stories). I will expand and break them into smaller elements. They will evolve the more I develop devices and learn by Dev-ops-testing.

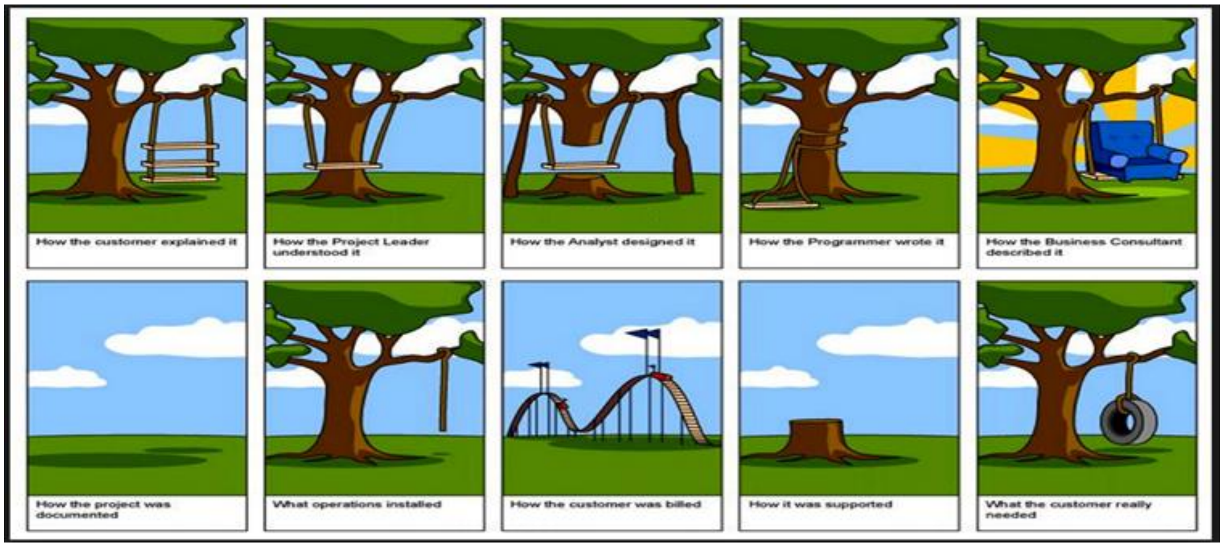

How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Figure- Requirements (stories, needs, etc.) can be viewed many ways
Reference: https://www.phase2technology.com/wp-content/uploads/2013/11/requirements1.png

There are two major classification of needs: Functional and Non-Functional.

Functional and Nonfunctional quality needs (requirements, stories and/or con-ops)

When I define a system-device, I should include functional and non-functional needs (for example see table above). The industry refers to needs as: requirements, stories, con-cops, use-cases, and desires. There are differences in the literature between these and I will not address those in this short book, but readers ultimately should be aware of the difference. Here is our list of IoT “need” refinements:

- Functional - necessary actions for the hardware, software, and system
- Physical – definition/constraints on dimensions, weight, temperature, voltage, color, etc.
- Logical – definition/constraints of actions, reactions, inputs, outputs, processing, etc.
- Interface – interactions between systems, users, environment, software, integration, etc.
- Modes & States – definition/constraints on operational conditions of system in normal and abnormal scenarios
- Non-Functional
- Performance – time, speed, usage, traffic, etc.
- Reliability
- Production – how is the hardware, software, and/or system to be produced or acquired
- Verification and Validation – how is the software, hardware, and/or system to be tested, verified and validated, particular to regulatory and/or legal considerations

Now the impact of these needs to the V&V/tester may be obvious to most. These are or will be the things I V&V/test. Many IoT groups will say “gee there is nothing to test now, so why do I need any testers”. Well time for another story.

A Story:

So, this project was producing needs (shall statements) and con-ops use case models. They had hardware, software, operations, managers, and customers all reviewing the needs. It was tempting to say “gee…”, but I did. I had one lone V&V/test experienced person. This person reviewed each need for:

- Testability
- Producibility
- Clarity
- Conflict (with other needs)

The team found about 30% of the needs/con-ops had major issues during the review. I found another 25% of added “new” needs which had to be added by the tester asking about “off normal” system usage. And I found many other minor changes which made better products and improved testing later. Now the team followed Agile, so the needs were expected to change over the life cycle, and by the end almost every need had been changed or reworked. However, the device worked perfectly the first time it was used by a stakeholder and everyone was happy.

Funding Capital (how to find money)

Show me the money

Whether you are a small start-up or working for a large company, to do IoT will take some capital. Figure Funding shows some of the major items need to get funding and results of doing a funding cycle. Some IoT device will die before this stage is done.

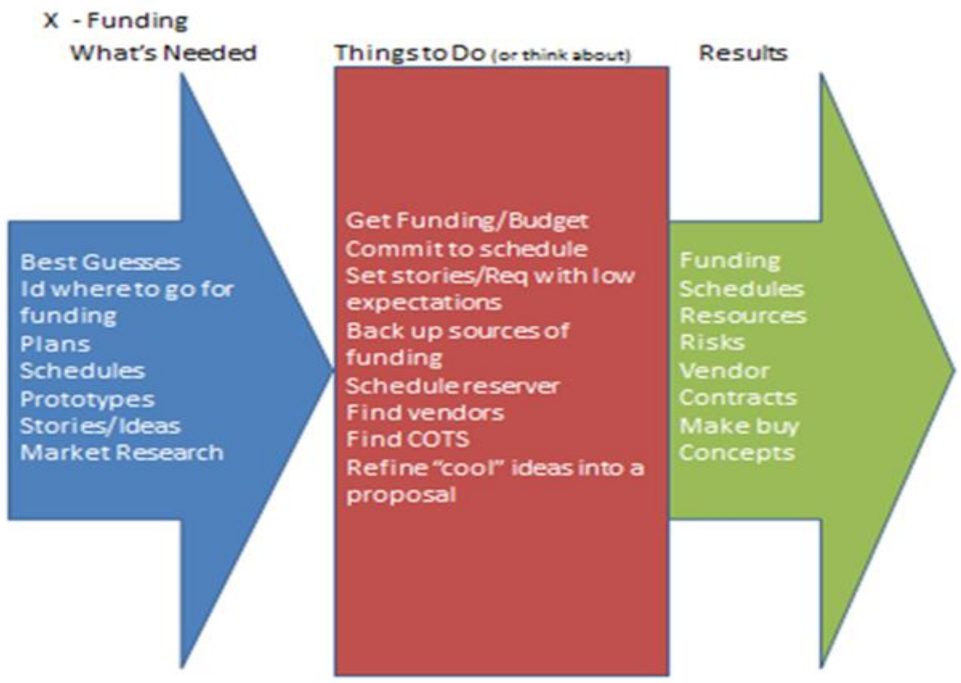


Figure: Getting Funding
Reference: Jon Hagar Built figure for IoT Classes 2016

Key Note on "What's Needed – Things to Do – Results" Figure: In eBook 2 I use this figure format as a short hand space savings. In bigger hard copy books an author would have text explaining the items in the figure in repeated detail. This is an Agile eBook. I provide minimal text for these figures, so if you have a question on the text in the figures, email me.

Editor's Note: Besides giving information only by figures, my mind works by having lists. Many editors and publication believe lists are not acceptable, but then their minds work in long text narratives, and mine does NOT. Be Agile, and think in lists or, again, email me a question: Jon.d.hagar@gmail.com

Testers need to make sure some funds are allocated to V&V/testing. How much will be from a small amount (say under 5% of total funding) to a lot (say over 50% if you are dealing with a life critical IoT device). The allocation is determined by risk tolerance, integrity level, and things such are regulations or company rules. There is not one answer.

Remember:

- Under sell first version features

- Have reserve of money (and time)

A Story

One of the authors build a first of a kind experimental house. The house had needs, design features, and "automation". The budget started at x dollars in the planning stage. Then the vendors submitted proposals to the plans. The budget grew to 2x. I started selling assets and worrying the experiment might fail. Building started. I fired 2 different vendors and took over management. The funding profile grew to 2.5x. Finally, in the last stages of production, the actual expenditure grew to 3x and I had to cut back on "need statements". I had to take out loans to finish, but I did finish. The experiment basically worked, even while I reduced features. Bottom line was the grow of funding while painful, should have been expected.

Link: Picture of Hagar House – https://hagartirebales.wordpress.com/

Here are other items to consider in funding:

- You will have more cool ideas. Consider having project back logs and debt to "place" hold ideas.\
- Remember Agile ideas of building what is most needed first, but may be not the easiest. If you get a main idea working, then you can live another day.
- Vendors and off the shelf items will need funding and they will have issues. Plan and fund redelivery. It may take longer to solve a vendor issue due to communication and distance.
- Vendors will not have understood the problem you were trying to solve. If the needs (requirements) you provide them are not totally clear, there will be problems that take funding to solve.
- Less funding is more because the more money you get the more product you will need to deliver, but if you under sell and keep funding low, you may be able achieve this easier.
- Don't plan on getting rich any time soon. A lot of funding will be spent to get a working rev 1 version or prototype, then later version will be created, and then a device that can be sold. Many start-ups are years of production before they see a "profit".
- Even if you are working for a big company who provide funding. They expect results too. The thoughts above apply to a big company.
- Remember the funding stakeholders now have become your first customer. The customer may not always be "right", but they must be kept "just happy enough by good enough" to let you continue.
- Don't invest more personally, than you can afford to lose

A Story: Robot

One of us built an IoT robot. The robot was both a physical and logical cyber device. As in most IoT devices most of the hardware was off the shelf with some customization. A major of the software was vendor provided, but there was customized software. To start, during manufacturing many of the instructions were less than clear and communication with vendors occurred. Next one of the hardware components (the Wi-Fi device) had a failure. This took three weeks of span time and 4x the funding to fix than was planned. New hardware had to be delivered from the vendor and some software fixes were needed. Phase 1 of the robot was completed but much later than wanted. In talking with other IoT teams, this story and impact to funding is very common.

Figure: IoT Robot
Reference: http://celebratescience.eu/wp-content/uploads/2014/09/Lego-Robot.gif

Suppliers and Acquisition – You are NOT doing all the work yourself

Almost (dare I say nobody) nobody builds their own hardware and software totally from scratch. You will be being buying off the shelf hardware. You likely will be having other companies doing aspects of manufacturing. You will have some software from other vendors. You will build some hardware and software yourself while doing your own system integration (as the story above indicates). You may contract for support work in the operations areas, e.g. the cloud, analytics, help desk, etc.

An IoT device will be an integration of many vendors' products and some customization from your organization. The customization may be too hardware, software or both.

The bottom line is you will need to be dealing with vendors and subcontractors. This involves legal, schedule, and funding concerns.

Story: Going to the Movies

Do you look at the credits in modern movie? There is the cast you see, directors, producers, writers, camera-ops, editors, best boy (what is that?), stunt people, and a list of credits that goes on for many minutes. So, in modern systems (a movie is a system), there will be many people and companies involved. IoT devices and system will be more like a movie than a simple program or web page.

Your organization will need to manage select, acquire, and manage suppliers. This should scare you even if you've done it before.

The impact to V&V/testers during supply and procurement can include:

- Supporting trade studies to select hardware and software.\
- Evaluation (testing) of hardware and software during a life cycle stages (even planning).
- Assessing product and vendor risks.
- Modeling products for testing and integration support
- Creation of test systems, harnesses, frameworks, and tools to work with standalone or integrated components.
- Providing feedback information to development, management on components.

Supplier trade studies and selection activities

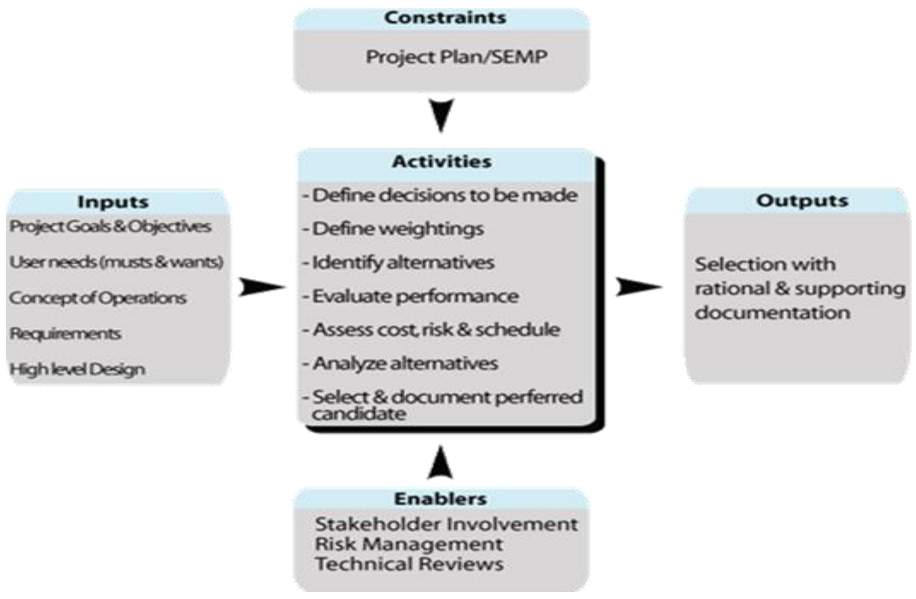


Figure: Trade study practice
Ref (https://www.fhwa.dot.gov/cadiv/segb/views/document/sections/section3/48_files/image001.gif0

In IoT much of the work and elements will be done by other organizations. You will like have third part hardware and software that is commercial off the shelf (COTS) solutions. Your many need to have customization done by a vendor or internally to your effort on vendor supplied components.

Our experience on various small and large-scale efforts is that other organization, even with the best of intention, will funding, schedules and plans (as the Robot Story relates).

If you are customizing hardware and have an outside party, say in China, doing the manufacturing of piece parts, this will be another source of time and effort. I have the following recommendations:

- Don't give the whole baby to vendor (say the Chinese).
- Conduct trade studies (tbd ref) to select the "best" vendor.
- Track and monitor what your selected vendors do (trust by verify).
- Have several vendors working on things separately.
- Your team "glues" and integrates things together.
- You will have parts that don't work, have a process to fix and substitute them.
- You will have software that is buggy and needs fixes, some level of testing should be considers.
- You will have to figure work arounds for hardware and software problems.
- The above things will cost you time and money, so plan it.
- Have a recovery plan "B" and stay agile

At list above implies I can plan to work vendors and suppliers. Further, some development teams do set based design (Ref). In set based design, I will produce 2 or more versions of a component by different vendors. If one fails, I dump it and go with the one that works.

A Story: Toyota system –

Set based design may cost more money up front, but when one vendor fails, you have a back-up. The Toyota story indicates that large organization may save time and money by this approach. However, this must be part of IMP, budgets and IMS with agreement by stakeholders. I have seen this avoid problems from high risk components.

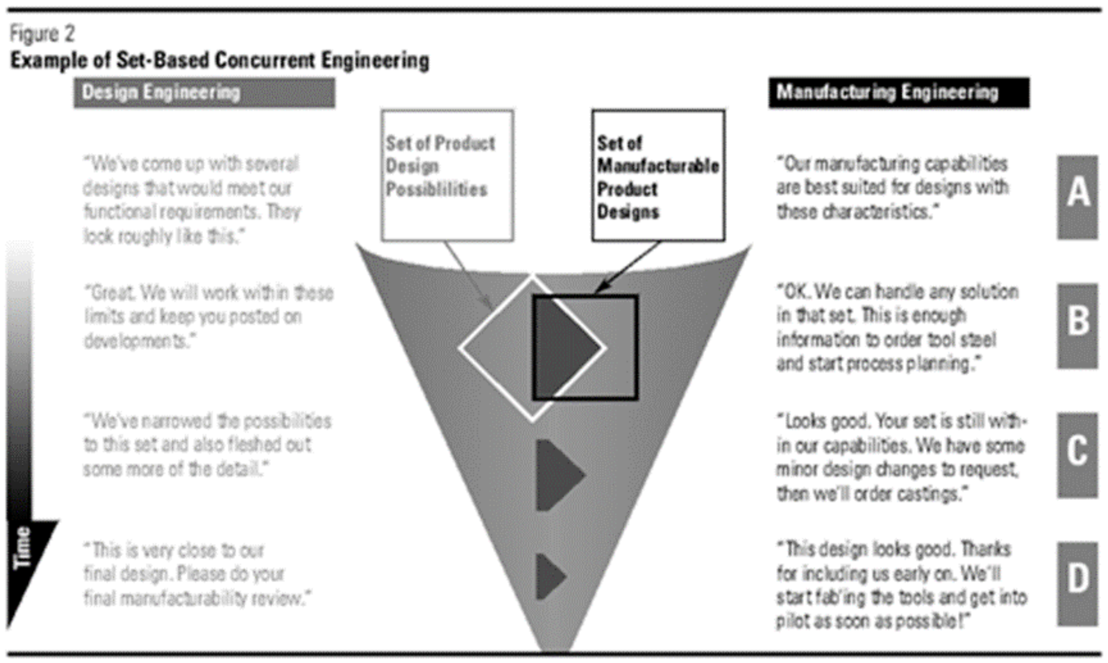


Figure 2
Example of Set-Based Concurrent Engineering
Design Engineering
Manufacturing Engineering
Set of Product Design Possibilities
Set of Manufacturable Product Designs
"We've come up with several designs that would meet our functional requirements. They look roughly like this."
"Great. We will work within these limits and keep you posted on developments."
"We've narrowed the possibilities to this set and also fleshed out some more of the detail."
"This is very close to our final design. Please do your final manufacturability review."
"Our manufacturing capabilities are best suited for designs with these characteristics."
"OK. We can handle any solution in that set. This is enough information to order tool steel and start process planning."
"Looks good. Your set is still within our capabilities. We have some minor design changes to request, then we'll order castings."
"This design looks good. Thanks for including us early on. We'll start fab'ing the tools and get into pilot as soon as possible!"
A
B
C
D
Time

Figure: set based design ideals
http://sloanreview.mit.edu/files/2008/12/4025-ex2-lo7.pn)

In selecting and working with vendors consider:

- Know the key features of a component
- Assure the vendors focus the key features first
- Remember that selecting the cheapest vendor is not always best and most cost effective in the long run
- Have scrap and rework plans for vendor items
- Assure your IP protected
- Have clear and defined interfaces
- Have budget, schedule and staff to support on site vendor activities
- How to work with vendors:

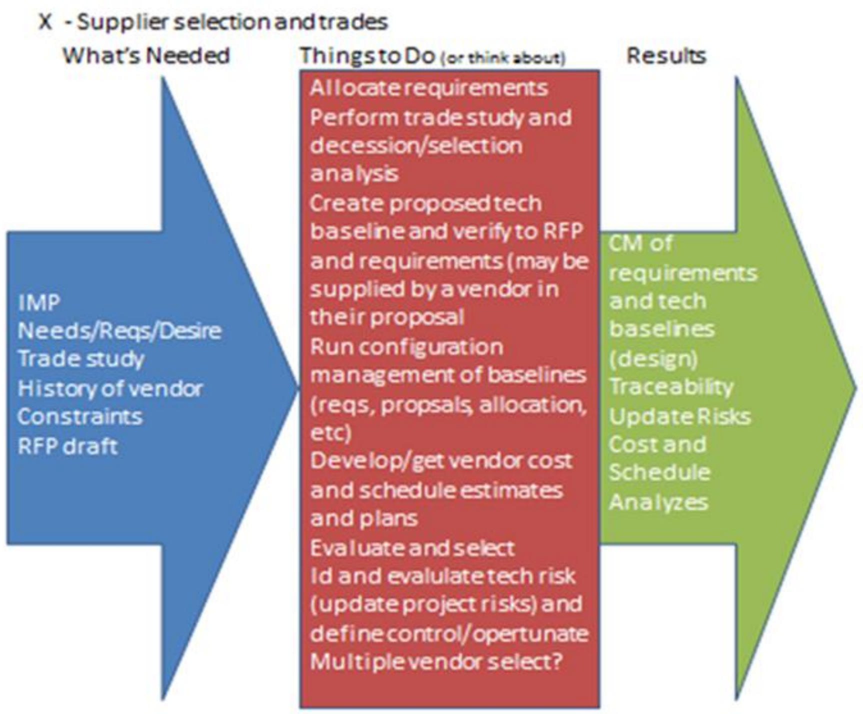

X - Supplier selection and trades
What's Needed
Things to Do (or think about)
Results
IMP
Needs/Reqs/Desire
Trade study
History of vendor
Constraints
RFP draft
Allocate requirements
Perform trade study and decession/selection analysis
Create proposed tech baseline and verify to RFP and requirements (may be supplied by a vendor in their proposal
Run configuration management of baselines (reqs, propsals, allocation, etc)
Develop/get vendor cost and schedule estimates and plans
Evaluate and select
Id and evalulate tech risk (update project risks) and define control/opertunate
Multiple vendor select?
CM of requirements and tech baselines (design)
Traceability
Update Risks
Cost and Schedule
Analyzes

Figure: Working with vendors
Reference: Jon Hagar Built figure for IoT Classes 2016

The impact of working with vendors to V&V/testing can include:

- The V&V/test life cycle looks different because you may get products earlier, e.g. in planning during a trade study
- Increases in the automate because changes in the life cycle, e.g. more regression and smoke test
- Impact from hardware which is completed, but has "issues" which end up being fixed in software
- Development organization which assess internal testing/V&V by the vendor on their products

Managing and assessing a third-party vendor

During Dev-Testing activities in IoT development there will be management of third party vendors. The vendors provide things like hardware, software, tooling, support, and other items that are more efficiently sub contracted. Sub contract management is important and is an area many IoT development teams have issues on. To aid in management and the associated assessment of vendors, I provide a list of question to ask yourself or more likely your vendor.

Depending on the answers to these questions, more actions and work may be needed on your part. These are only as start. If you are unsure about this area and starter questions, you should probably find books and references on contract management as there many in the business management world.

- Does the vendor have a positive contract performance history?
- Is the vendor performing work for your company or now?
- What are the allocated requirements/needs/stories/features (traceability to vendor)? Are these clearly defined to the sub (you only get what you ask for)?
- Is team doing configuration management of the allocated requirements (and by whom)?
- Can the vendor explain back these allocations (do they really understand needs)?
- Prime Contractor V&V/Test/QA actions on the vendor(s):
- Do they have testing on site and is the testing defined on contract?
- Can your team visit and check the planned V&V/QA/testing on site?
- Once a product is delivered, does your organization have acceptance/receiving V&V/testing when the shipment arrives (you must do this for first of kind)?
- What are I (the prime contractor) doing to ensure subcontractors gets the job done right?
- Be on site and assess activities closely
- Negotiate (contracts, story, requirements, needs, design) and be Agile
- Is it defined what is to happens if a product fails in V&V/testing?
  Remember you get what you pay for and if you ask for more, they will charge for it.
- Can you get prototypes to do early V&V/test?
- What property, patent and copyrights do you have in place (remember Intellectual property is what keeps you in business)?
- Remember everything costs?
- Good quality costs, and bad quality may cost more?

There is much more to managing subcontractors, but this list gets your team started.

Estimation

Some believe I should not estimate IoT, but mostly you will find you must have some estimates to get support and funding, unless you are self-funding and then figure 3 times the money and schedule you would like. In any case estimation is hard, likely wrong in predictions, and a big topic. This should section will just get you thinking. If you really must do estimates much research, reading, and practice is needed.

The reason people don't like estimates is that all estimates are guesses about the future and therefore usually wrong, but people treat an estimate as if it should not be wrong. To get funding and budgets teams will likely need to produce estimates, even with the knowledge they are "wrong". Here are some thoughts to help in starting.

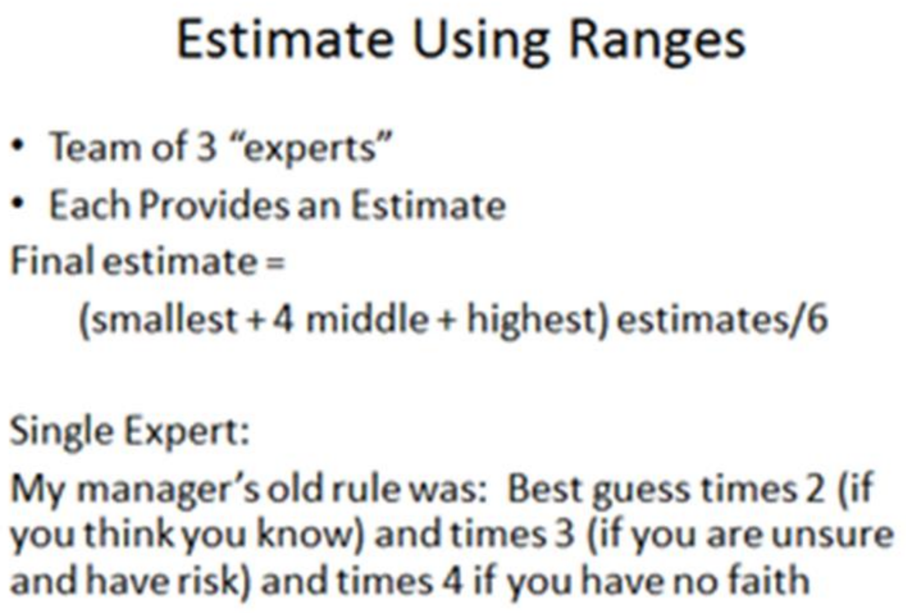
Agile estimates are used as a "not to exceed" budget while working on what is most important first. By doing the most important things first, along with continuous integration and production, I am always very close to having a "working" product, even if it is only a small percentage of totally complete device. This way when the money runs out you have a partial product to deliver.

More traditional teams may have fixed budgets, times and materials. This has risk but can work if I follow some of the agile ideals of getting to creating a working product ASAP. Some traditionalists rely on redoing the contract on change orders or law suites to make a working product. However, I do not believe this will work well in IoT

Estimation is a guessing game where you try to predict the future. Some of us call it "Guestimation". You need something to help get the funding capital. Nobody is likely to write a blank check of unlimited funding and schedule for an IoT project. And there is a "rule" that you always expand your features to fill the budget and schedule you have. You should try to keep budget and schedule in check.

So, remember the "Keep it simple and small (KISS rules)" rule. This increases the likelihood of IoT success

Top down or bottom up Estimation



Estimate Using Ranges

- Team of 3 "experts"
- Each Provides an Estimate

Final estimate =
(smallest + 4 middle + highest) estimates/6

Single Expert:
My manager's old rule was: Best guess times 2 (if you think you know) and times 3 (if you are unsure and have risk) and times 4 if you have no faith

Figure: Estimates a "formula" 3 Oracles of Delphi Method and Jon's old Manager
Reference: Jon Hagar Built figure for IoT Classes 2016

The figure above list some very rough rules for estimating cost ranges. This is known by some as a top down estimate.

I much prefer doing a very detailed line by line, item by item, task by task spread sheet estimate. To do such an estimate you must have some detailed plans and ideas. This is a bottom up estimate. When this is not possible, then the figure above may be a start.

Variables Affecting Estimates

Here is list of factors that may impact estimates. These should be kept in mind.

- Complexity
- Start-of-Art vs known state devices
- Team knowledge and skill
- Team size and
- Team location (co located vs worldwide)
- Security concerns
- Safety issues
- Integrity level (see IEEE 1012)
- Customer and users (usability)
- Stakeholders
- Management
- Regulations and law
- Reliability and other qualities

Venture capital estimates

If you are lucky enough to have somebody willing to provide funds via start-up venture capital, they will likely want some estimate. Hopefully they will spell out if they want top down or bottom up and the associated level of detail. Venture capital people usually want some kind of history or precedent before they will hand funds over. This leads to the idea of Crowd-source funding.

Crowd-source funding

You can google on crowd-source funding sites and organization. I have no recommendation for these organization, just like I don't recommend particular software test tools. I have talked with many IoT start-up who have told me "stories" of crowd-source funding, so here is what I have learned:

- Under promise features on the first dev cycle
- Do not ask for too much money on the first dev cycle
- Get just enough functionality to "meet" the promised first features (KISS)
- Repeat and expand

Estimation over a life cycle

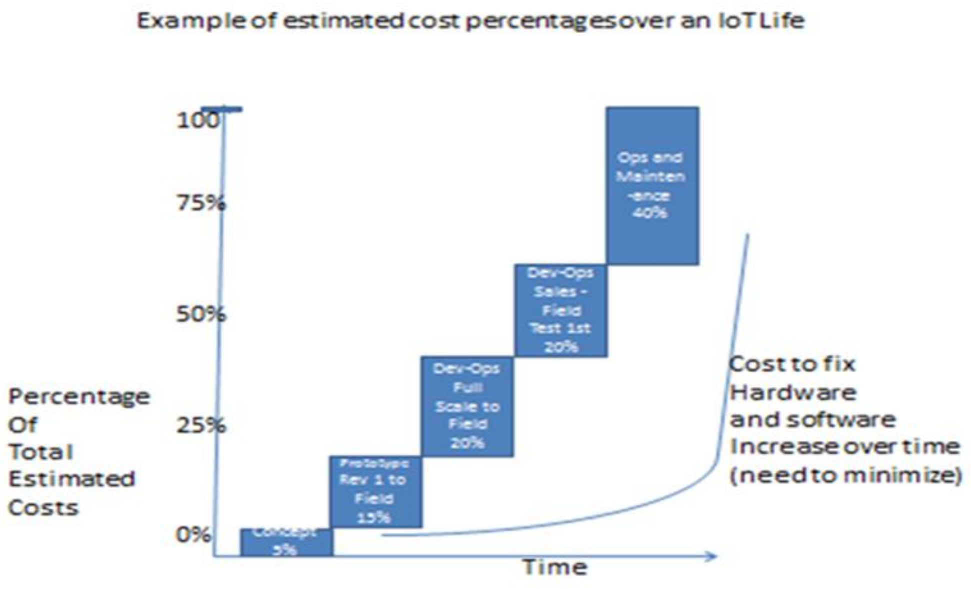


Figure: Estimated costs over an example IoT Lifecycle
Reference: Jon Hagar Built figure for IoT Classes 2016

The Figure is an example from defense industry circa 9/93. It represents an allocation of a total budget over a larger life cycle by percentages. Most IoT will be different. Each phase of "costing" is notional and not absolute. It is possible that cost may be lower for example during testing than classic curves because of Dev-Ops metrics feedback. Here are some thought examples on each phase:

- Concept phase 1 – first estimate – Here a project is just trying to get a prototype to stay alive to the next cycle.
- Phase 2 - Prototype rev 1 is given to crowd investors as a demo while dev continues to work on rev 2. This assumes some funding for phase 2 was received. In phase 2 the project will focus on get more needs (requirements) working to the point where maybe some real sales can be made.
- Phase 3 - First sales and deployment have started but the IoT device may still not be 100% complete. For example, security feature may be lacking and some failure correction functions may not have been finished, but the device is "good enough" to start income which can feed more dev and paybacks.
- Phase 4 – Hopefully at this phase the product is "complete". There may be enhancement, new features, and upgrades in the future, but sales can really take off here. The functions and

quality characteristics are really mature at the end of this phase. V&V and testing have also been fully completed.

- Phase 5 - O&M is the last phase and hopefully it lasts for years. However, cost curve can get out of control if quality is not good enough and the reader will note that 40 of total budget may be consumed during O&M. On many project, the 40% is low. On projects that go on for years (and this is desired), numbers as high as 60 and 70 percent have been seen. Do not underestimate O&M.

Estimate schedules (goes with most capital expense estimates) - Outline

Schedule – big picture – follow Agile for details (likely)

Man power – Get the right skills, small team or big, in house or out

Materials

- Prototype
- Throw away
- First production

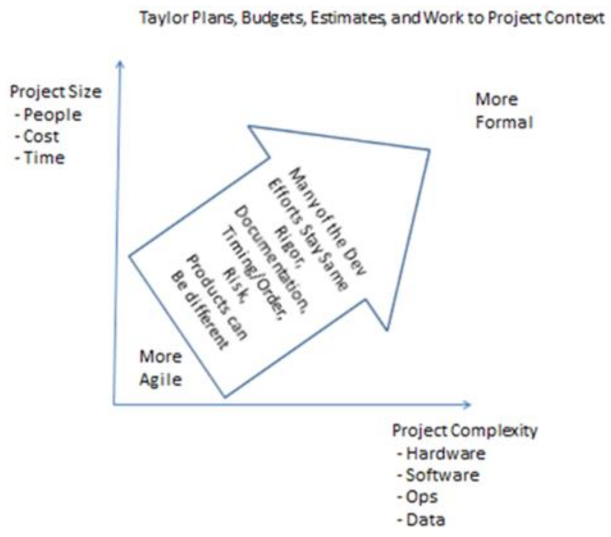

Taylor Plans, Budgets, Estimates, and Work to Project Context
Project Size
- People
- Cost
- Time
More Formal
Many of the Dev Efforts Stay Same
Rigor, Documentation, Timing/Order, Risk, Products can Be different
More Agile
Project Complexity
- Hardware
- Software
- Ops
- Data

Figure: Tailoring and relate more to testing

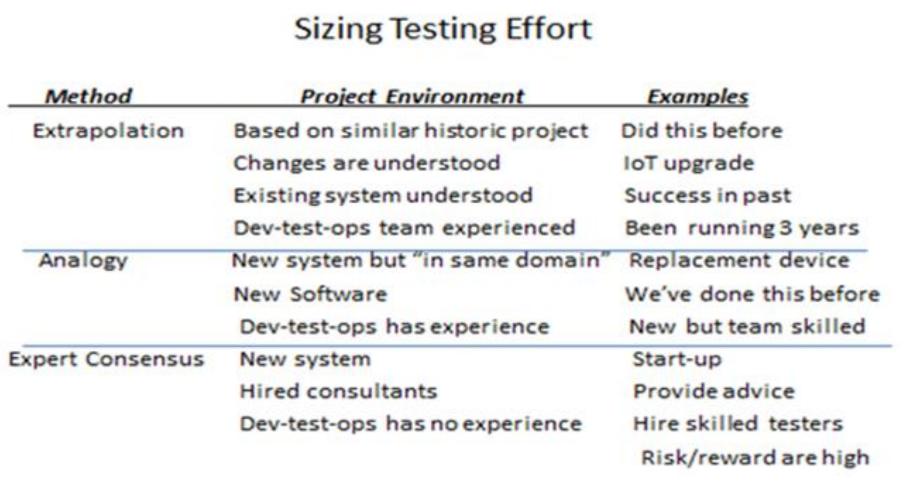
Reference: Jon Hagar Built figure for IoT Classes 2016

TBD pic needs words

Test Estimation Sizing - Outline

Use above but based in testing

Approaches – turn below into table and add Agile planning game and?



Sizing Testing Effort

Method	*Project Environment*	*Examples*
Extrapolation	Based on similar historic project	Did this before
	Changes are understood	IoT upgrade
	Existing system understood	Success in past
	Dev-test-ops team experienced	Been running 3 years
Analogy	New system but "in same domain"	Replacement device
	New Software	We've done this before
	Dev-test-ops has experience	New but team skilled
Expert Consensus	New system	Start-up
	Hired consultants	Provide advice
	Dev-test-ops has no experience	Hire skilled testers
		Risk/reward are high

Figure Sizing Method/Environment/Example- TBD where did this come from and mod level
Ref: Class by Jon Hagar

TBD

Figure: above Taylor to testing – add risk and TBD 29119– redo figure

TBD pic needs words

More on WBS and use in Est

Really need to get the other refs and use if you are doing serious tests

Quality, Verification, Validation, and Testing

We put V&V/Testing after the money because I did not want the book to assume a waterfall appearance and I are testers first. I understand most startup and early project will not think test or quality, but maybe they should have just enough thought about them to assure success as the IoT device evolves.

For details on V&V/test planning, design, execution and environments, the reader should refer to books 3, 4, and 5.

Test and Quality

Some authors have announced testing is dead. This needs some context. Zombie testers should be dead (or are they already dead?). There is need for IoT information that V&V/testers provide.

"Good Enough" concept from software may mean rev 0 of an IoT device has no or minimal formal V&V/Test, but you must have some "checking". The team needs to determine the product is good enough to be released in the wild. Now what "released" means can be very different

Picture of good enough?

As the IoT product evolves so will testing. I plan to use things such as:

- Automation
- Analytics
- Exploration testing
- Others

To get better and better products. The nature of the IoT product will vary thus determining the V&V/testing. See book 3 and 4 for more information.

Verification and Validation

In the next few pages I basic over of V&V, mostly using figures.

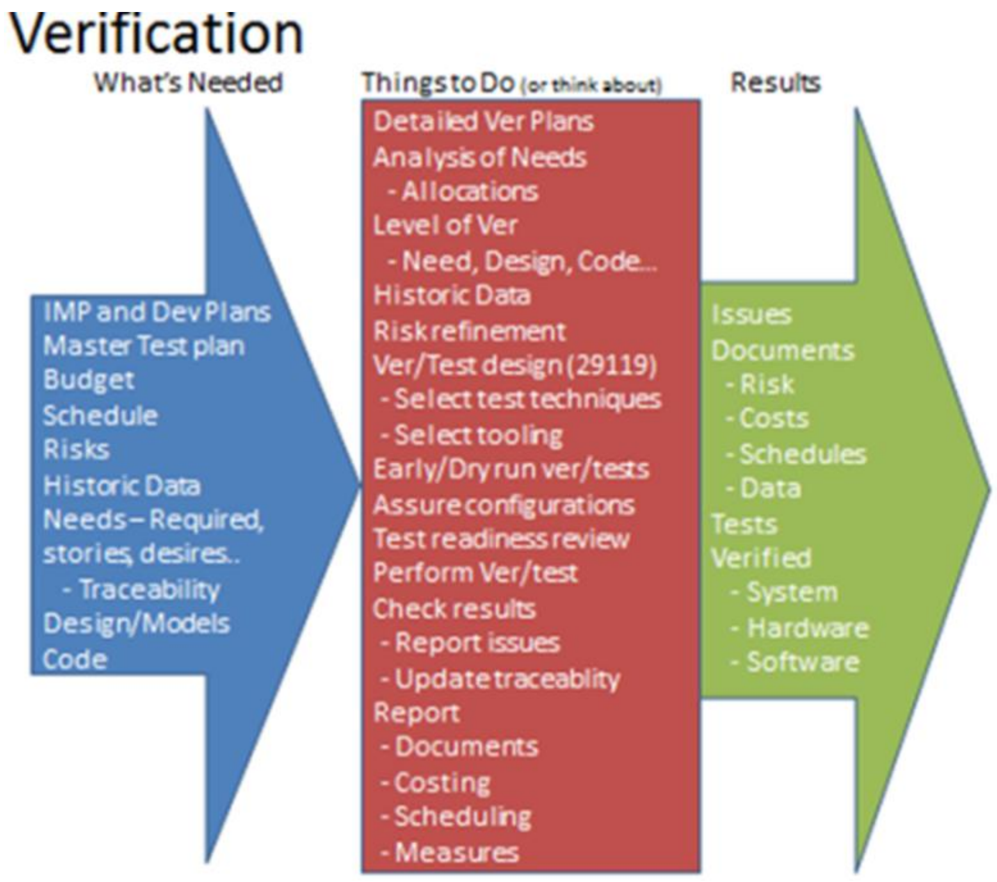


Figure: Verification Activities
Reference: Jon Hagar Built figure for IoT Classes 2016

Verification tries to assess “did I build the product right”. I have some “truth” information such as requirements, con-ops, or stories that I “check” against.

V&V approaches include:

- Inspection – tester visually looks at the item
- Analysis- tester uses tools such as modeling, math, simulation, similarity, or logic
- Test – See early definitions of testing
- Demonstration – Tester does a test with actual system (unmodified) and no special equipment
- Checking and automation – Tester uses tools and/or automation to assess the software (minimal thinking)

Validation is focused on “did I build the right product”. In validation I assume everything, including requirements or stories can be wrong. It is much harder to do that verification, but both are needed for a good IoT system.

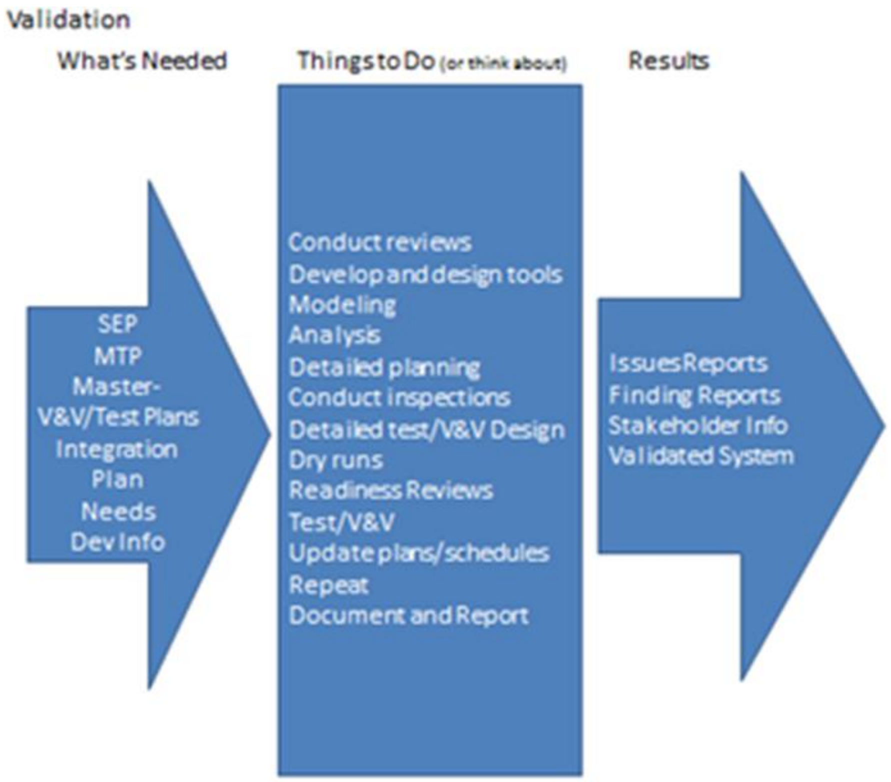


Figure: Validation Activities
Reference: Jon Hagar Built figure for IoT Classes 2016

Minimum activities needed to release an IoT into the wild

We get this question: “what is the minimum I need to do”. The answer is it depends on

- Context (customer, product, regulations, risks, etc.)
- What can you afford to use (money, business, life, etc.)

There is no one size fits all answer but the answer takes some engineering and management work to find.

There is more information on V&V/testing in books 3, 4, and 5.

Operations (Ops)

In IoT operations will be important. There will be user operations, data center ops, communication networks, and device/sensor/controller ops, to name a few.

As developers and testers, I must think “ops”. In some sense, it is a chick and egg problem. Do I think Dev, test, or ops first? Yes

The Ops efforts need to address:

How is your system going to be use?

What data is use, owned by whom, and kept where?

What operations reliability?

How is maintenance of hardware and software going to be done?

What about security and privacy?

Does testing stop after first delivery or do I continue test maybe using the user as a tester?

How does Ops change over time?

Yet again, there are no single “best” answer to the question and further the answer will change over time.

System Maintenance, Security and Retirement

The big 3 in Ops are likely to be maintenance, security and product retirement.

IoT security/privacy is largely agree as a top “concern” for IoT. User are becoming aware of this. Laws are being passed around the world. Your project can not assume anything on security/privacy. It may be best if you develop and test with an open mind on security. eBook part 4 address security testing in more depth.

Whatever you decide on IoT security/privacy is likely to change over the next 10 years or so. This brings up the topic of maintenance.

Hardware maintenance may be a little tricky. If I take the auto model, I will be seeing IoT product be returned for fix or updates. How is this paid for? How do you deal with recalls? Is the IoT device a “throw away” or something that is more expensive?

Next is software maintenance. The IT and mobile maintenance model has moved to constant software push updates. However not all IoT device may be able leverage this model and then I have security concern to think about in maintenance ops. Further some IoT device the software may be difficult to update unless is designed for an update (think about firmware updates). The automotive industry has to have cars brought into the shop to do a software update. Will IoT have this model?

We will guess like current industrial world maintenance will span from automated push model to the “you bring you device into a shop”. The choice will be a development and maybe marketing decision based on many factors. Just know the software industry is littered software updates that failed. Some care is advised.

Retirement-Disposal

What do I do with the IoT device when I are done with it?

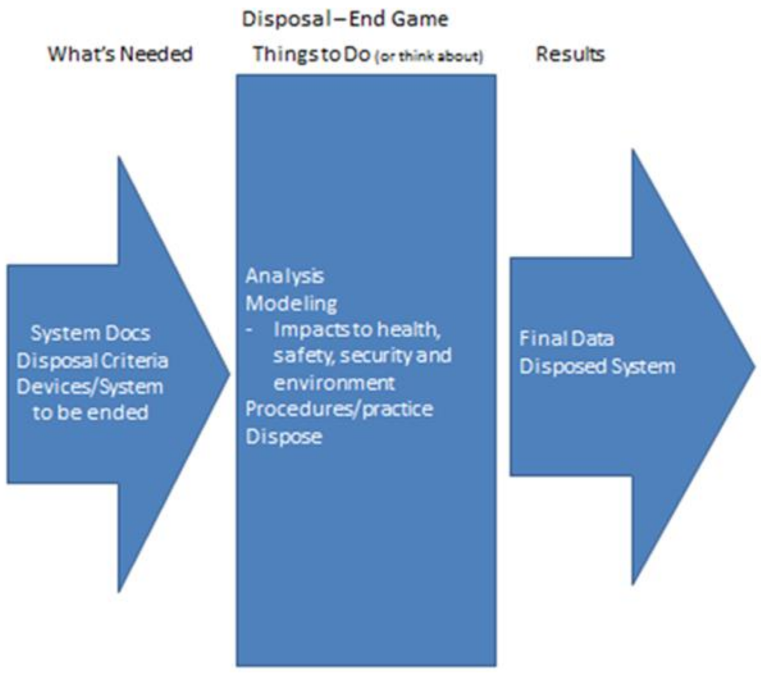


Figure: End of device life

Reference: Jon Hagar Built figure for IoT Classes 2016

We live in world where many products, including electronics and IoT cannot (should not) be thrown into the trash. There are batteries, chemicals, and construction that are worthy money (to be recycled) and/or constitute hazard for disposal.

When I develop and field an IoT product retirement and disposal must be considered and even tested.

How many people have old TV, electronics and equipment sitting around the house because the trash man will not take them?

However, this is just on the hardware side, what about software?

Well it is just bits (0 or 1) right?

Wrong. There is data that needs to be kept safe, secure and private. Disposal needs to include this type of software. Several of us have removed and trash memory and storage media to keep data “safe”. The development and testing of IoT need to include software and hardware disposal.

Modern life is not simple.

Communication and Integration - Outline

Expand networks

Network changes and options

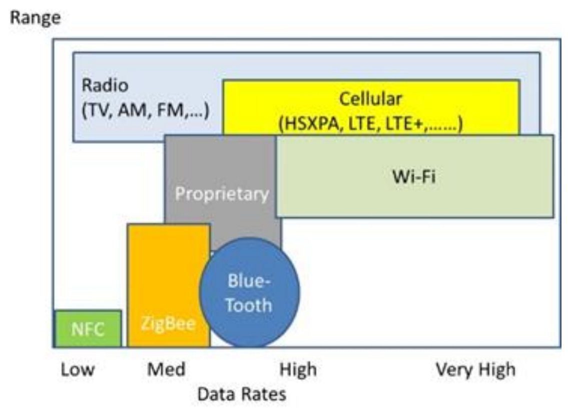


Figure Comm options vs Range and Rates

Reference: Jon Hagar Built figure for IoT Classes 2016

Integration with the new

What does V&V/test do here

IoT Operations and Data Analytics – Important- Outline

AI

Analytics

SODA – see charts

Protection

Analysis

Using the data – sales, test, dev and?

What does V&V/test do here

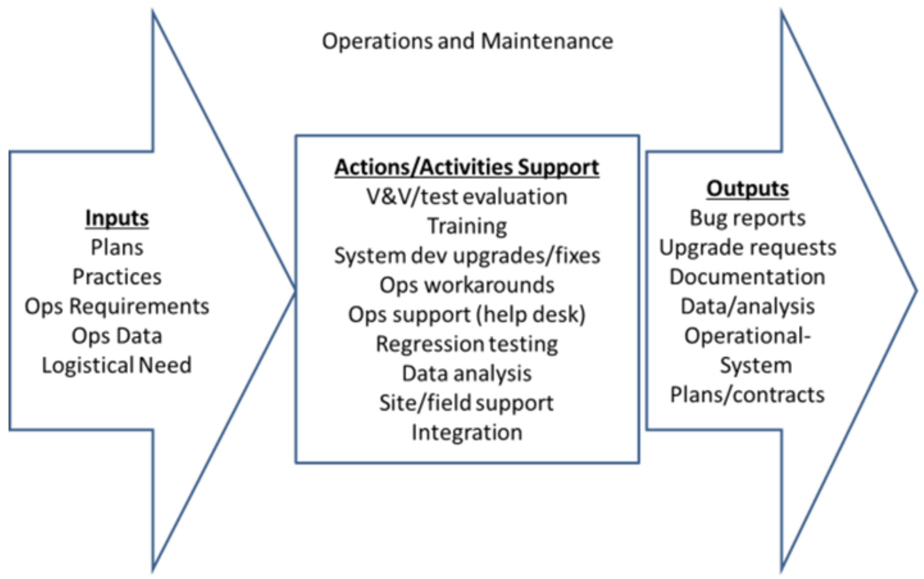


Figure: O&M -

Reference: Jon Hagar Built figure for IoT Classes 2016

TBD pic needs words

Release Deployment - Outline

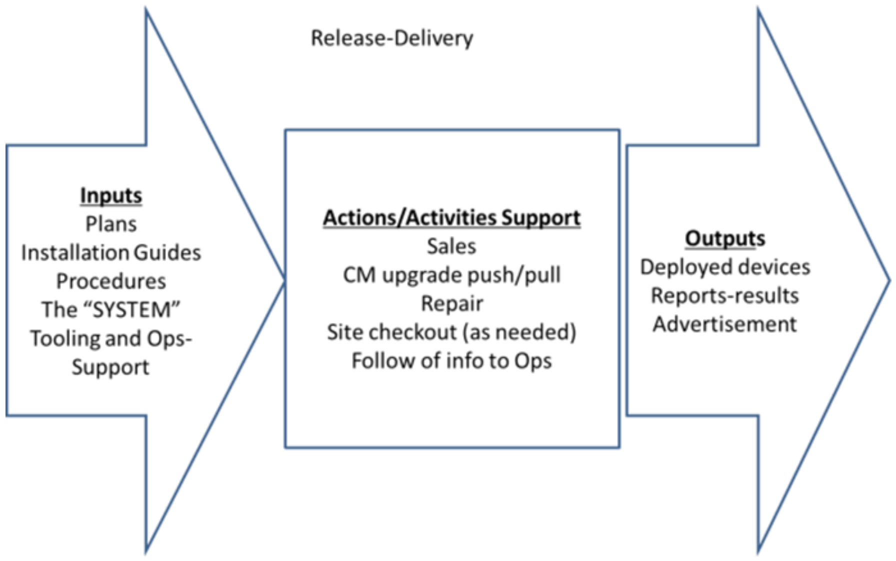


Figure: Release and Deployment

Reference: Jon Hagar Built figure for IoT Classes 2016

TBD

Picture figure for Data Ops and analysis

Collection

Protection

Analysis

Using the data

How fast is the issue id-fix cycle?

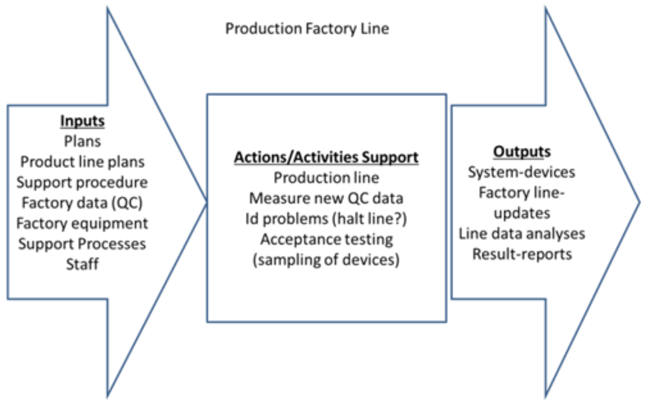


Figure Production Factory Line

Reference: Jon Hagar Built figure for IoT Classes 2016

Test and Quality

Products may have longer use lives that most development teams envision. The testing of IoT devices to provide information on the quality (s) of a device will need to address the near and longer-term usage of devices. Part 1, 3 and 4 address the test and quality factors in more detail. For this little short section, I wish the reader to think about the following:

- Testing provides the whole team information about the qualities of a device
- The perceived quality of a device may live forever in the user's mind
- Testing may be seen as necessary "tax" on device development which is to be minimized, but this can be short sited if the devices quality is not good enough for sales to succeed.
- Comprehensive testing is more complex than development, and so tester deserve equal "citizenship".
- Devices will often be used in new ways and environments that dev and testing did not address, and so these should be addressed during ops and maintenance.
- Stakeholder want a device to be successful and continue in recurring sales and updates, but this will mean regression and new function testing (testing never ends).

Evolving as you go

Our experience is that most project, processes, products and people evolve over time if they are going to be successful in the long run. This is likely for the small start-up and then even if a big company. This is also true if you are an Agile group or traditional hardware company. Those that do not evolve and change are likely to be overcome by market Darwinism. Evolution goes by names like Agile, process improvement, continuous quality, and many other "slogans.

For us, the key is to keep at evolution by trying new ideas and taking some risk, but do the change ins smaller "Baby" steps. DO NOT change everything all at once. I have seen many Agile team do this but then a large number of the Agile teams then failed in some aspect. I like slow evolution over time. This allows identification and correction of mistakes in the changes made. If you change to many things at once the real cause for the "failure" maybe lost in the noise of changes.

As I said, this approach to slow change works for project, processes, products and people. Keep in mind the change must be something that people pay for. For example, too often the high-tech world I see change to look and feel of things like the operating system, were the change just confuses users and violates are considerations for a ubiquitous UI. Yes, most change is good, but be careful.

Minimum testing to release IoT into the wild – a checklist

The following might be considered as a minimum for activities but this list can be debated.

Test Usability

- Check the UI (see tbd list)

Assess if the device is ubiquitous (see tbd)

Functional (requirements) and non- functional checks

- V&V hardware

- V&V software

- V&V system

- V&V ops & maintenance

Security test

Safety assessment and testing

- Integrity level

- Risks

Validate connectivity, communications and integration

- System

b. Networks

c. Standards

d. Lack of availability or limited data rates

6. Verify device performance and timing

a. Real-time response

b. Stress and load

c. Volume and capacity

7. Compatibility and interoperability testing:

a. Combinatorial testing on hardware, software, communication, etc.

8. Realistic field Testing

a. Chaos engineering tests

b. Crowd source tests

c. Smoke testing

9. Compliance testing with regulations, standards, and legal aspects

10. Software updates and CM

a. Regression

b. Security

c. New feature V&V

Risks and opportunities management in an IoT project

There are many risks and therefore opportunities in IoT. For example:

- You lose your money
- You lose someone else money
- Your product does not work (functional or non-functional qualities)
- You don't meet schedule
- You kill some one
- You go out of business.

I therefore recommend considering risks in IoT. There are the project types of risks outline above and other places in the eBook. Whether you do formal risk management activities or something less formal a basic process is given in the future below.

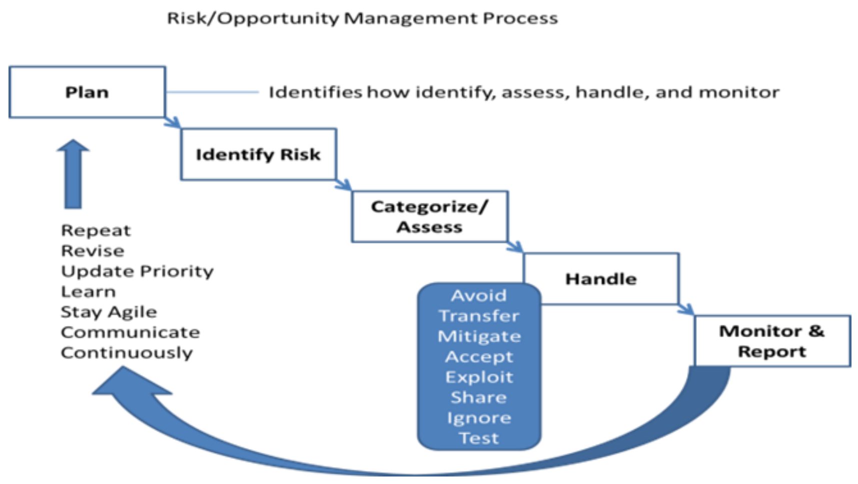


Figure: ROM process
Reference: Jon Hagar Built figure for IoT Classes 2016

This process would be for the project and can feed into the test effort. For a detail risk based test flow, readers should refer to other texts (tbd), Jon's first book (tbd ref) and/or particularly ISO 29119. Such risk process should be ongoing over the life of a project. They do not have to be heavy weight. I have used the above basic flow using sticky notes posted on a white board. This guided my teams testing and was very agile.

There and Back again to the Beginning: Development (Dev)

System IoT Considerations

As I have already mentioned in other books, one of the things that make IoT different from other tech areas is that I must consider hardware and systems engineering (SE). Readers should have a full book on SE and probably book addressing hardware in the IoT product domain. This should section will outline some basic SE concepts to get you start or for those that don't want to by a full SE text book.

Need pointers to books, standards, and web sites for SE TBD.

Formal Engineering – a Brief Introduction list

The table below list a large number of SE activities teams should consider. If these are not familiar, you should do more reading and research. Not every activity needs to be done formally (documented) and some Agile SE is possible, but the list gets one thinking. Underlined items are more critical

Table: Formal System Engineering Activities (in no particular order)

Table: Formal System Engineering Activities (in no particular order)

State the problem	**Design the system**	**Produce documentation**
Understand customer needs	**Do sensitivity analysis**	**Lead teams**
Discover requirements/stories	**Assess & manage risk**	**Assess performance**
Validate requirements and device	**Do reliability analysis**	**Prescribe and plan V&V/testing**
Investigate alternatives	**Integrate system components**	**Conduct reviews**
Do the quantitative measures	**Design & manage interfaces**	**Verify requirements and the system**
Define models	**Execute configuration management**	**Perform total system test**
Perform functional decomposition	**Integrate project management**	**Re-evaluate & improve quality**

V&V/testing are defined in my other eBooks, IEEE 1012, and ISO 29119. A starting point standard for SE is ISO 15288, the basic concepts of which are show in the figure below. Again, standard should be tailored and are only a reference point, but they do contain a base for project that have no base to start from.

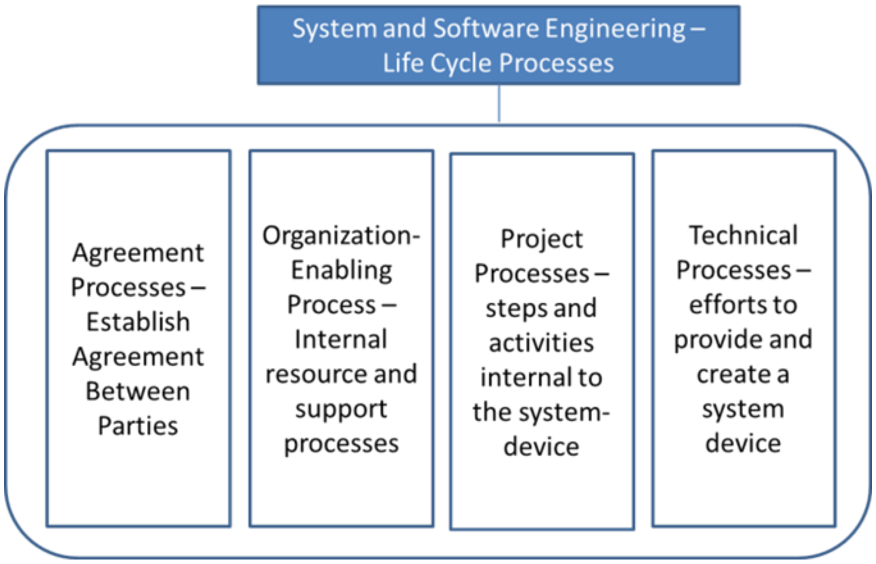

System and Software Engineering – Life Cycle Processes
Agreement Processes – Establish Agreement Between Parties
Organization-Enabling Process – Internal resource and support processes
Project Processes – steps and activities internal to the system-device
Technical Processes – efforts to provide and create a system device

Figure: ISO 15288 Concepts Picture – tbd link to 15288
Reference: Jon Hagar Built figure for IoT Classes 2016

V&V/test is integrated into SE at lower levels along with other activities are shown in the system interface figure below. No one group or person is likely to be able to do "everything" except on the simplest of IoT devices.

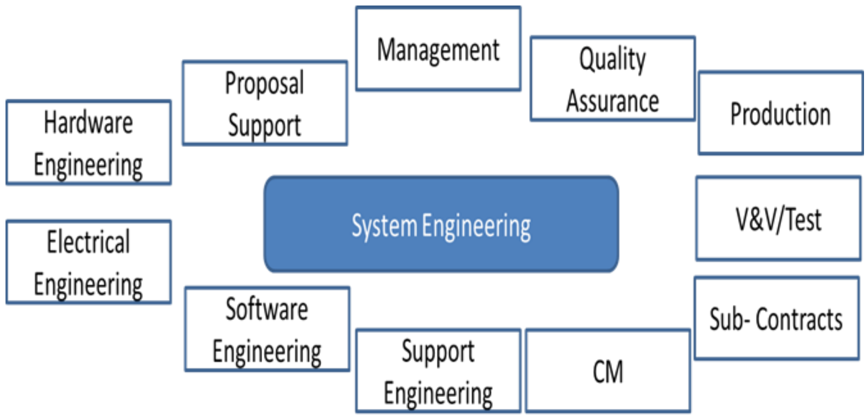


Figure: System Interfaces to other groups
Reference: Jon Hagar Built figure for IoT Classes 2016

Project and Organizational Project-Enabling Processes

Besides the basic engineering efforts of SE, hardware, software, and test engineering there a few important enabling efforts that teams should address to be successful. These include CM, measurement, architecture & design.

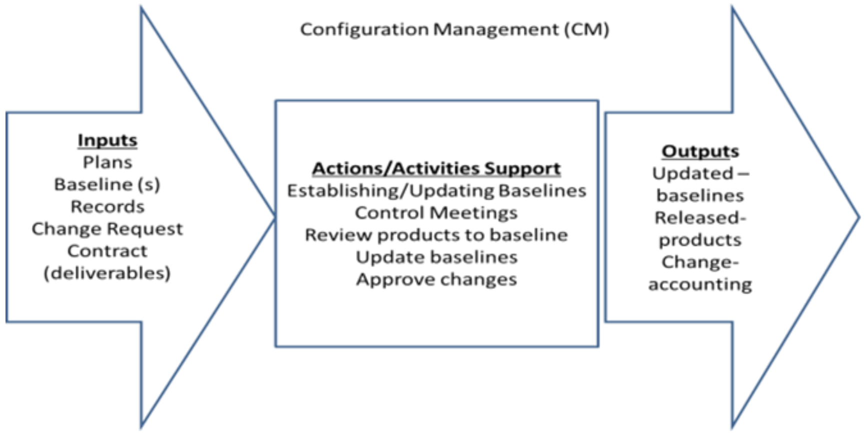


Figure: CM actions
Reference: Jon Hagar Built figure for IoT Classes 2016

I have taught many college classes where teams of students did group project for the first time. The team has done SE, software, and testing. I warned every team CM would be an issue because they were working in a group over a longer period of time. Every team had CM plans. Every student team wished they would have done better CM plans. The list of CM actions above is a minimal starting point.

Next support area is measurement. Most of us have a love-hate relationship with measures. This is because I use them, management, and stakeholders want to know them, and the same time somebody in the SE system will abuse them. A basic set of measurement action is shown below.

Measurement and status

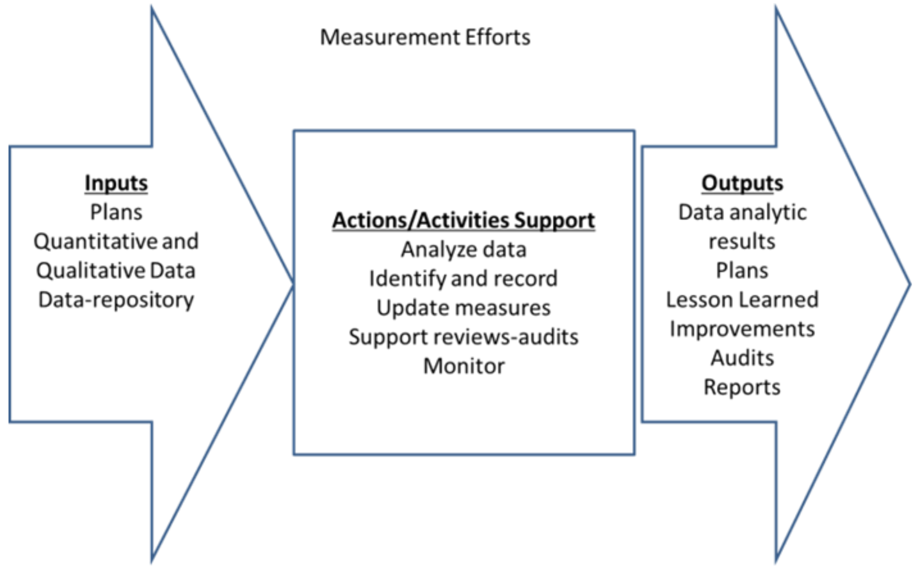


Figure: Measurement Efforts
Reference: Jon Hagar Built figure for IoT Classes 2016

Measurement:

Good – tells teams where they are, where they are going, and can communicate to stakeholders.

Bad – team get beat up when a measurement looks “negative”, teams will make measure look good to avoid punishment, and teams do not understand what the measure is tell them.

Ugly - teams will have too many measure to be useful and teams will not maintain/use the data

Test will have and use the following measures:

- ISO 29119 part 4 has a measure for each named technique
- Tester will count bugs and bug aging
- Testers will count thing like number of tests, number of requirements, and production rates

As a tester, I have used each of these. I have fought to avoid being abused by the measures. It is possible to use measures successfully, but it takes work (and is outside the scope of this eBook)

Architecture and Design Top Level

Architecture of a small IoT device may not be a critical SE task, but when I start talking about systems of devices and systems of systems of IoT, e.g. a smart city, the architecture will take critical items to work during SE. The architecture will apply to software, hardware, and operations. A basic set of architecture activities is show in the figure below

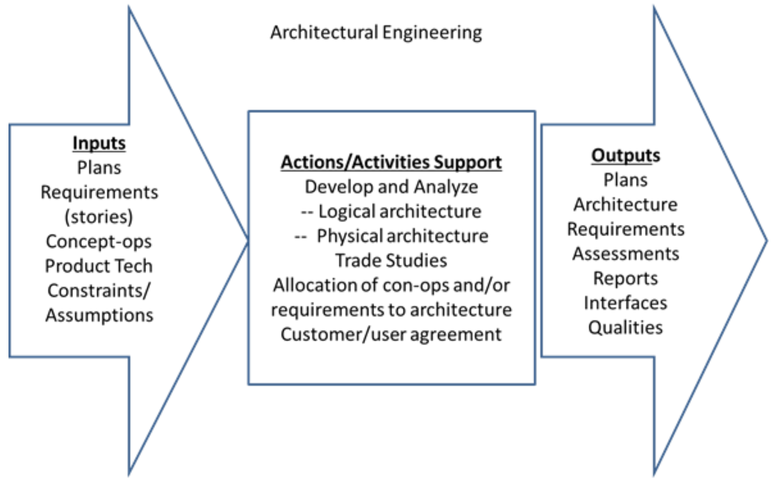


Figure: The top-level aka Architecture
Reference: Jon Hagar Built figure for IoT Classes 2016

To understand architecture, you should consider the company Apple. Apple has been very successful because they tend to provide architecture solution of integrated hardware, software, and systems. Their product integrates and "play" well together. They also have a consistency of usage that their consumers follow with loving devotion. IoT device teams would do well to understand the Apple "way".

Critical in system, system of systems, smart cities, big, etc.

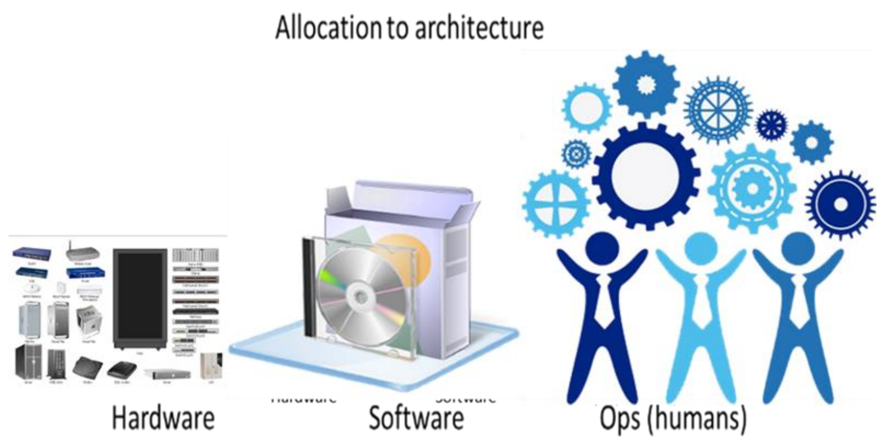


Figure requirements are allocated over three areas
Reference: Jon Hagar Built figure for IoT Classes 2016 and internet materials

Trade study – (AKA decision analysis)

An important part of SE that will be very important in IoT is decisions on what third party elements of hardware, software, and support to use. Most IoT project will "glue" elements together to create a device system. Teams will leverage existing elements, add in some customization, and integrate a new IoT device.

However, care must be exercised. A team may not want to buy the least expensive, or most featured element. The team must do decision analysis in the form of a trade study to evaluate candidates.

Here are four reasons to perform trade studies

1. Evaluate viable solutions against must have and wants (desired but not absolute)

2. Challenge requirements or resolve conflicting requirements

3. Identify an optimal, cost, quality, and/or schedule, design which balances competing requirements

4. Get customer/stakeholder buy-in

The figure below has an example trade study process, which can be used as a starting point.

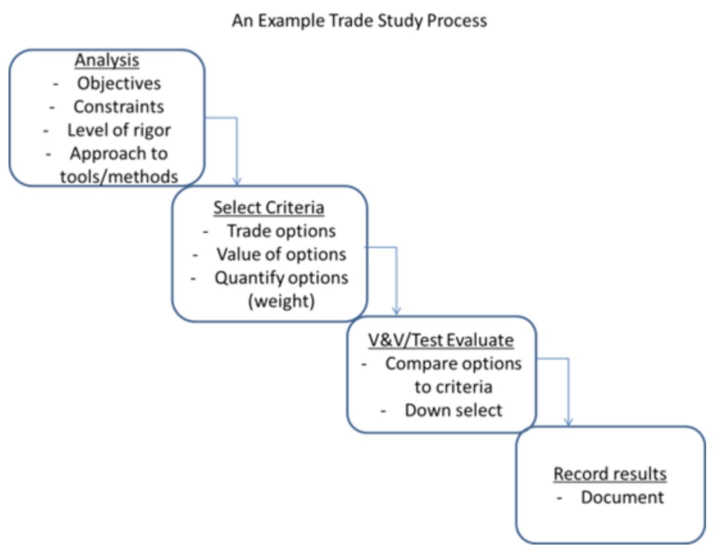


Figure: An Example Trade Study Process (recommended)
Reference: Jon Hagar Built figure for IoT Classes 2016

There is much more to a trade study (see tbd ref). However, since these books focus on V&V/testing, I will point out that a tester should be involved during this process because step "V&V/test evaluate" the options is where the decision is "made". Many teams make the mistake of "trusting" what they are told by vendors. Tester know how to "not trust" but test. Tester are also helpful at doing the analysis and criteria steps, as well as documenting the decision. Teams want to record decision, because it is very common to "revisit" trade studies later in IoT project efforts, because of changes, regressions, failures.

TBD link to decision analysis reference works

Design - Safety and Test Standards for IoT

What are all the safety standards? Here is a partial list that will likely change as time goes on:

IEC 62304

IEC 61511

IEC 61508

IEC 60730

ISO 26262:

ISO 29119

IEEE 1012

IEC 62061

EN 50128

IEC 61513

IEC 60335

IEC 61508 is more of an industrial standard (Examples: PLCs, drives and motors)

IEC 61513 is for the nuclear environment

ISO 26262 applies to automotive

DO-178C and D is for the Avionics industry

IEC 60730 is for appliances

EN 50128 is for the railways

IEC 61511 is for process industry

IEC 62061 is for machinery

IEC 62304 is for medical industry

IEC 60335 is for product manufacturing

UL listings

Ouch, there are a lot I might need to consider in testing and design of IoT

Hardware Design Considerations – outline

Is this a throw away device?

Is this big hardware

Is this mission or life critical (amount of loss?)

What does V&V/test do here

Ref that book on hw design

Software Design Considerations – outline

Dev – the 4 parts? And third party

Post processing sw

Security

Introduction

2. Engineering Roles

3. S/W Engineering Environment

4. Process Improvement

5. Requirements Analysis

6. Implementing Software Systems

7. Software Testing

8. Inspections (Peer reviews)

9. Software Measurements

10. Risk management

11. Planning

12. Summary

13. Optional Sections

    Agile

    COTS

    Objects

What does V&V/test do here

Communication and Integration - outline

Providers

Networks

Protocols

How big is the integration?

Who owns the integration

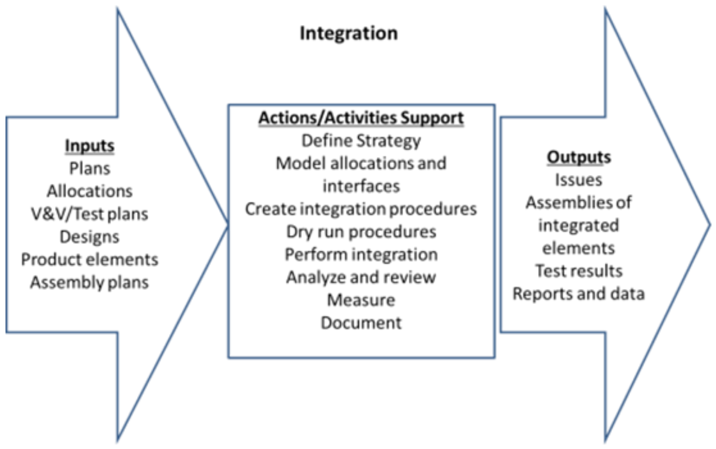


Figure: Integration of IoT
Reference: Jon Hagar Built figure for IoT Classes 2016

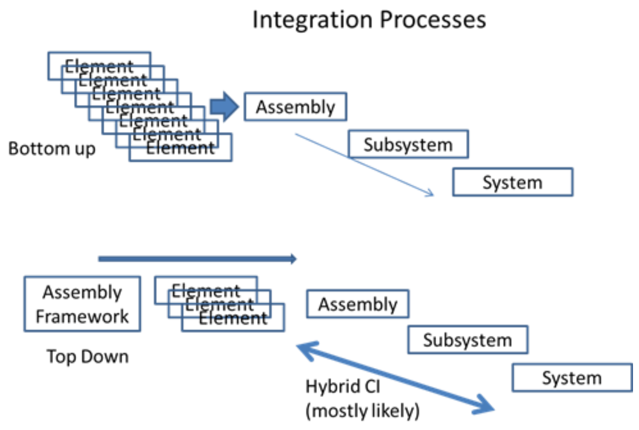


Figure – Integration process options: bottom up (hardware), top down and Hybrid CI (software)
Reference: Jon Hagar Built figure for IoT Classes 2016

What does V&V/test do here

Product and development life cycle impacts -

So much of the world, including hardware, is evolving towards Agile. A close concept to Agile is DevOps. In this eBook I have introduced, but not completely addressed ideas from both Agile and Dev-ops. Jon's suspicion is IoT will be done with DevOps and/or Agile. There will be some traditional waterfall particularly with groups that are hardware centric or historic. However, since many IoT teams will want to develop a product ASAP, Agile and Dev-ops will be life cycle approaches of choice.

Whatever the life cycle IoT team should ask "What is the minimum for a viable IoT product?" at each stage of the life cycle. IoT unique factor in order of importance influenced by life cycle to answer the question include:

- Time to market
- Costs including money and schedule
- Ubiquitous UI
- Functionality
- Security
- Safety
- Communication
- (other)Non-function qualities

Agile seeks to balance these and always be ready to deliver (maybe not to a consumer) a product. DevOps philosophy includes the user to aid in testing. Both approaches include ideas of short development cycle steps including continuous dev, testing, integration, and delivery.

IoT data reporting abilities because of communications will support concepts like data analytics and dev/test use of in-use data (see section eBook tbd).

The context of the product will help determine life cycle choices and evolution. As I have said, how a IoT game device is developed will be different then a life critical driving system which is linked to a city-wide network. Both will have risk and likely important quality characteristics to dev and test around, but they are different context and so there is no "one size" fits all in life cycle choices.

IoT teams and management may be looking for the one size magic bullet, but there is no such thing. If our jobs were easy, robots would be doing them. Our jobs and life cycle choice are hard, fun, interesting, and I get gook pay for them. Be happy.

Summary

This part of the IoT testing eBook series was to help testers, and maybe developer teams understand some of the big picture that surround IoT development. No short book addressing the topics of funding, planning, development, and supporting areas can really address these topics fully in under 1000s of pages. However, I tried to hit the ideas, topics, and pictures to get tester/workers started. If you want more in these areas there are hundreds of books and web sites dealing with these topics in great depth. One should refer to these.

The idea of the eBook was to give testers some knowledge so they can be more effective on IoT teams. I expect many IoT teams will have few or even no testers at first, but at some point, the IoT device team will say “gee, I need some V&V/testing”. If this happens early on or later, some of the information in this part of the eBook may help.

Our experience is that many testers and even developers have great ideas in the local area of expertise but miss the other parts of the dev-ops-test and management picture. This makes communication with the stakeholders, managers, customer, and even users harder than it needs to be. This book tries, in a “reader digest picture/list format” to provide starting points.

Please provide feedback on what you like, dislike, and find missing. I have our view and bias, but they are limited.

Appendix A: Skills needed for Dev

The following is list of knowledge and skill areas that people working in IoT Dev-Ops-Testing may want to have or build. The list is certainly not completed and may miss your favorite items. I organize by traditional discipline though should not be taken to mean traditions are the way to go in IoT Dev-Ops-Testing or that more than huge numbers of skilled people are needed. Further, people focused in one traditional discipline will typically need support skills in or from other areas.

Management

- Finance
- Negotiations (art of the deal)
- Communication
- Planning and Scheduling
- Estimation
- Delivery and customer relationships
- Human Resource understanding
- Support areas
  - QA
  - Tech writing and documentation
  - CM/SCM

- Dealing with people
- Setting goals, providing feedback and praise
- Teamwork
- Agile

System Engineering

- Modeling
- Thinking
- Full life cycle see ISO 15288

Hardware Engineering

- Mechanical
- Sensors and actuators
- Electrical
- Packaging
- Production and manufacturing
- Maintenance
- CAD/CAM

Software Engineering

- Requirements (needs)
- Modeling
- Design (how)
- Implementation (coding)
- Evaluation (test, inspection, reviews, analysis)
- Debug
- Life cycle understanding see ISO 12207

Test/V&V

See the other eBooks in this series

Operations and Maintenance

Analytics

Customer support (help desk)

Re-delivery

For extended meaning and definition of the above, I recommend further reading (see reading list) and/or internet searches

Appendix B: Dev-Test check list

The following is a checklist for projects. It is only a start to get you thinking

I. Estimation, planning and strategy

a. TBD

II. Development

a. TBD

III. Testing

a. TBD

IV. Qualities: TBD

a. TBD

V. Quality: Security

a. Ensure only legitimate users access the device. Use authentication at start up and log in (see tbd)

o This applies to both network and physical (port) access

b. Devices have digital identification and trusted computing ability to prevent hacker counterfeiting devices or data streams.

c. SCM policy is clear and address software updates with downloaded updates having authorization authentication. This keeps device software updated. Users should be allowed to have both push and pull updates with network checks to keep users informed of "issues.

d. Internal and external data is encrypted.

e. See article

f. Links:

References for Additional Learning

TBD

Glossary of Definitions

In this book, I have followed—as much as possible, common usage and definitions from the following sources.

- SEvoc — http://pascal.computer.org/sev_display/index.action
- IEEE — ISO/IEC/IEEE 24765:2010 Systems and software engineering (Vocabulary)
- eBook 1

Please refer to one of sources listed here, or one of the references given throughout the book, or you can do an Internet search for the term. (Google can be your best friend in helping you to find things). Finally, you can always contact us.