Testing IoT Systems Using V&V, Security Hacks, & Data Analytics

Jon D Hagar
Grand Software Testing, LLC
Hot Sulphur Springs, Co, USA
jon.d.hagar@gmail.com

Abstract

The Internet of Things (IoT) systems have become a hot growth area in tech, which in part will drive the modern economic world. Engineers are busy creating a connection to the internet for nearly every electronic device, meaning that many "things" or devices that were never before connected to the internet or other devices or networks, will now be, and that presents a "boundary issue" of where to stop testing for testers as well as management. Many IoT managers and stakeholders may wish to limit the testing of these devices or their connections to control costs by only covering the IoT device software. Their rationale being the "thing" or "system" may be well known, and only the IoT software is "new". This limited testing may be acceptable for a small percentage of products, but lower, system-level integrated testing may leave other IoT fielded projects with severe problems—especially with regard to security. This paper considers several important IoT system-level test activities and the technical skills to support the maturing world of IoT system testing.

In many cases, the IoT device, software, edge interfaces, and finally, the cloud elements of the IoT system will need to be tested. Simplistic, manual testing of IoT needs to evolve into complete Verification and Validation (V&V) during the system lifecycle while often including levels of independence in the test team organization. Testing activities such as: analysis, reviews, inspections, modeling, structural tests, functional tests, etc., will need to be appropriately budgeted and then allocated and detailed in testing plans and strategies in order to address the risks of IoT systems testing beyond just the IoT device itself.

Traditional manual testing will have limited application in this new domain compared to test automation or using Artificial Intelligence (AI) with data analytics ("a method of logical analysis," Merriam-Webster) to drive the IoT evaluations. To support these new levels of testing, testers from more traditional software will benefit from learning and practicing advanced engineering skills. Testers will have advantages when building on traditional software test concepts, as well as expanding their use of newer software, hardware, and system engineering concepts. Test practitioners that can master advanced testing skills will be in demand and may find the IoT world very challenging, exciting and fun.

Biography

Jon Hagar is a systems-software tester consultant supporting software product integrity, verification, and validation, specializing in embedded, IoT, and mobile software systems at Grand Software Testing in Colorado. Jon has worked in software engineering, particularly testing, for over 40 years. He has managed test teams and built test labs using test automation. He teaches classes at the professional and college level. Jon regularly publishes with over 100 presentations/papers, best paper STARWEST 2011, a contributor to three books, and author of three books: "Software Test Attacks to Break Mobile and Embedded Devices" (CRC Press), a children's book, and "IoT Testing" (due out in Sept 2022). Jon is an IEEE/ISO editor and author on various international testing standards. Finally, Jon is a skier, biker, builder of a one-of-a-kind passive solar, thermal mass home, and heavy equipment operator. Projects he has supported include: control systems (avionics and auto), mobile-smart devices, spacecraft, and ground systems information technology (IT), as well as consulting on attack-based testing on smartphones and IoT systems.

1. Introduction to IoT and challenges testers now face

IoT puts computer processing and communications in every imaginable device, including things that were not originally electronic or connected. Figure 1 presents a partial listing of common IoT systems, including consumer, industrial, and items in the middle between these two. This figure is not a complete classification of all IoT ‘‘things” but presents the large scope of possible IoT systems.

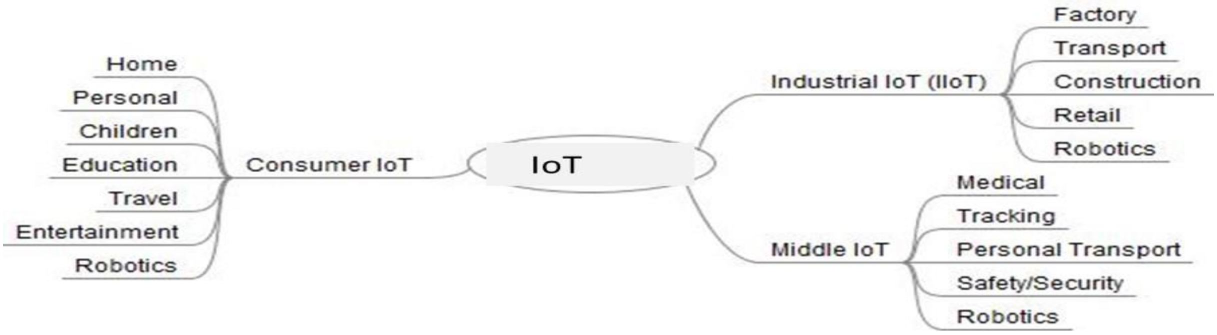

Figure 1: IoT will be in everything and everywhere [1]

Questions arise with IoT, such as: “How and to what level do projects test such devices, such systems or even collections of IoT systems?” Many projects and testers are used to simply testing the software in isolation on a PC or web page, leaving the complex system space to users and stakeholders. This paper provides pointers to classic testing of the software device itself. However, it is my position that just testing the device in isolation is only a beginning. The real problem for IoT systems will be going beyond the device to the higher levels and interfaces. In addition to classic testing, a key point for IoT systems will be the need for system and software level testing and assessments, which some teams may wish to minimize.

When there is limited system-level testing, I advise development to disclose the level and type of testing in the product information. For example, disclose that the IoT device was tested at the functional level of assessment in defined limited environments. This disclosure will allow later customers and stakeholders of the IoT device to incorporate the IoT device into their extensive system and to understand when they may need to be conducting additional system-level Verification and Validation (V&V [2,3]) with testing to ensure the whole system is viable.

V&V/IV&V, testers need to understand these terms.
Verification is assessing a system to ensure it meets requirements and design specifications. Validation assesses a system to assure it meets stakeholder needs beyond the published specification [3]. While verification can use checking against things like requirements during testing, validation is more open and uses the understanding of the engineers doing the work. The use of independent test, V&V, and/or IV&V organizations have been used in many areas of industry [3,4] for years when system level assessments are warranted. Using this approach aligns with much of the historic IT world (companies, large organizations, and governments), where large and complex software, hardware, and systems are subject to integrated systems testing by the owning stakeholders or their representatives before deployment to users. For example, many companies have IT departments evaluating or testing products before integrating them into their corporate IT systems or on their networks. Testing beyond an individual IoT device should be considered because many issues and faults will only be detected at an integrated system level. Industrial, government, and commercial stakeholders will understand such limitations because they have dealt with integrating IT systems for years. Users, even testers, used to simple ‘‘plug and play” may not understand the importance of testing a fully integrated system nor the limitations or consequences of testing just the device.

Such users may be disappointed in an IoT device, stop using it (so much for product viability), or worse many users will publicly complain about the system on social media (bad product advertising). While this does not sound like a significant problem at first, complaint reviews can kill a product with a few bad posts from unhappy users in these days of social media. IoT companies wishing to stay competitive should be aware of defining where and how to finish the assessment of IoT device qualities. Testers may want to recommend and advocate when more extensive system-level testing may be needed to satisfy customers. Stakeholders building complex IoT-based systems will need to consider this information when expanding testing beyond the IT department into their IoT world. This may lead those procuring IoT systems and doing integration toward the concept of Independent Verification and Validation (IV&V [3])

I assume the IoT development team will use some common testing practices [1,7,8,9,10,11,12,13] on the device itself and beyond the device to a limited extent. However, many IoT devices will become part of a system of tens (in a smart home) or even thousands of connected devices (in a smart city). In these cases, the idea of an independent test team beyond the developer testers may come into play. A classic concept of IV&V [3], or an independent test organization [4], has a history in the tech industry. IV&V has a long history in critical government systems.

This paper continues next considering areas of IoT system V&V/IV&V assessment. A broader picture for IoT system testing/V&V is presented in [1]. Next, the impacts of AI and data analysis as part of testing are presented. There will be many opportunities for testers and companies in the IoT domain for these independent test areas.

The present economic society of Technology-Capitalism [5] is driving companies, IoT teams, and testers to add activities and new skills. This paper includes some of the disciplines of test technology used by skilled engineers. Additionally, the paper briefly considers the impacts on generalized IoT stakeholders. Testers will particularly need to think about the impacts of AI, DA, and software test architectures (STA [6]), as well as classic testing concepts. Even IoT advocates have a hard time envisioning the whole nature, scope and expanse of IoT system impact on stakeholders.
Note: STAs are defined here as the hardware, software, test, and system elements that support test execution. They are part of test plans and include test strategy, very high-level test design approaches, models, and practices. They are analogous to the system, software, and hardware architectures yet are not universally accepted by the industry.

2. IoT Testing and IV&V Assessment Beyond the Basic Device

I expect that most testers will understand the importance of test planning and having a comprehensive strategy. Planning is often considered an essential first step considering test cost, schedule, risks, staffing, environments, techniques, and documentation. More information can be found in [7,8,9,10,11,12,13]. As a start, most testers plan to target the IoT device itself. However, I recommend due consideration of aspects found in Figure 2. This figure shows a smart home that contains IoT devices. The devices are connected to an edge system, which could be a PC with a router, a cell phone or another edge device. Edge devices provide the first line of advanced processing for IoT, particularly in AI, DA, and early user feedback. IoT devices are often small and do not perform much advanced data processing. The edge devices are the first point where such processing can actually be performed, and hence why *system IoT testing is important*. After edge processing, the IoT data and information will be passed into the cloud for additional processing and analytics. Therefore, the team must ask an early question in test planning: where does their testing start and end? It is possible that just testing the device will leave issues in the edge and cloud undetected resulting in unhappy stakeholders which may eliminate product viability or worse, violate security. In the IoT test planning, the team should consider a test strategy where they weigh concepts such as: test automation [14,15], environments, models, tools, risk-driven testing, and what test approaches or techniques should be used. The team should ask what the needed product qualities are including: IoT security, performance, safety, reliability, etc. [16]. Will any project be successful if the team restricts itself to just the device's structural and functional tests and does not address these other topics? The elements of Figure 2, test plans/strategies, and concepts form a sample IoT STA. An STA can assist in answering some of these questions.

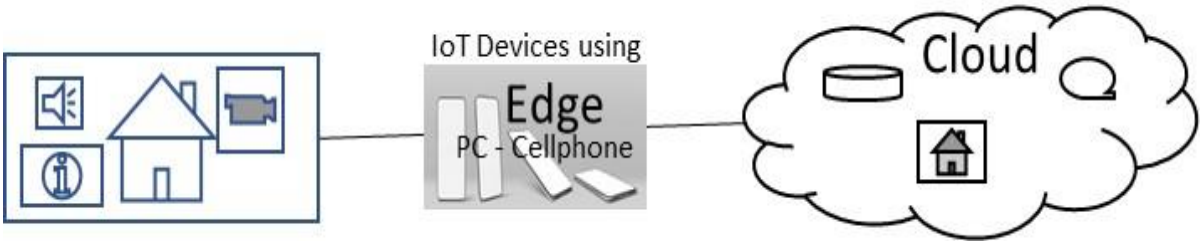

Figure 2: The IoT System from the device to the edge to the cloud [1]

There is no one answer for IoT test planning and STAs. Risks, costs and schedules must also be considered. Stakeholder needs must be factored into the test planning. The disclosure that only the device has been tested under set plans and strategies may force customers or users to decide what other test activities may be advised. For example, a medical device used at home for only personal information may not need IV&V. However, a similar medical device that feeds data to an AI or doctor for monitoring a critical condition (across networks) may need IV&V as well as regulatory approval (e.g., FDA and [1]). Of course, the cost and planning for each configuration would be quite different. Unfortunately, customers may not fully understand the IoT system risks and may not be in a position to make informed decisions.

This kind of IoT situation should lead one to consider system-level IV&V and V&V, as well as independent, from the development team, third-party testing in IoT planning.

3. IV&V/V&V assessment at system and software levels

A key point here is that for IoT evaluation and quality, test activities need to extend beyond the IoT device to the internet systems or other networks to which the device is connected. There will need to be industry and societal considerations of ownership of this extension. In their literature, individual device manufacturers may rightly claim that their quality ownership responsibility *stops* at the connection to the edge or, more extensively, to the internet/network. This leaves an open question of where this responsibility falls and an opportunity for test organizations to assume ownership. For example, I have a semi-smart house that generates much of its own power and heat. We see problems with the connections inside the house, between the solar panels, and the outside power grid. The solar panel manufacturer was little help, leaving the ownership to us. The problem lies with the connection to the cellular network for the panels to ‘‘call home" to report energy generation. And that is where the vendor had not tested since they did not plan, have an STA, or test for that possibility. We have figured it out, but we are technologists. Most home customers will not want this ownership. The problem becomes more prominent when we look at examples such as medical devices or IoT systems to control smart cities [17]. For these more extensive areas, we see the rise of ‘‘test cities," [1, 17] V&V, and IV&V organizations to conduct the quality assessment of the systems or system-of-systems. Organizations, companies and testers will need to step into these gaps doing IV&V/V&V. An example allocation of activities is shown in Figure 3.

The names of these activities are defined in texts such as [1,3,7,8,9,10,11,12,13]. Readers unfamiliar with them should use these references to gain a better understanding. Also, the allocation can be done differently based on context. An example is the IoT world that Aerospace has moved into, where pretty much all of these are practiced. When consulting with large Aerospace companies, I was surprised to see very familiar test plans, system architectures, computers, techniques, and tools between the organizations that had minimal communication but were solving common IV&V/V&V allocation problems. This commonality tells me that while Figure 3 is an example of allocation to IV&V, for IoT industries looking to solve system-level quality test problems, the commonality is a good starting point. In critical, large, and/or complex IoT systems, many of the activities in Figure 3 should be considered.

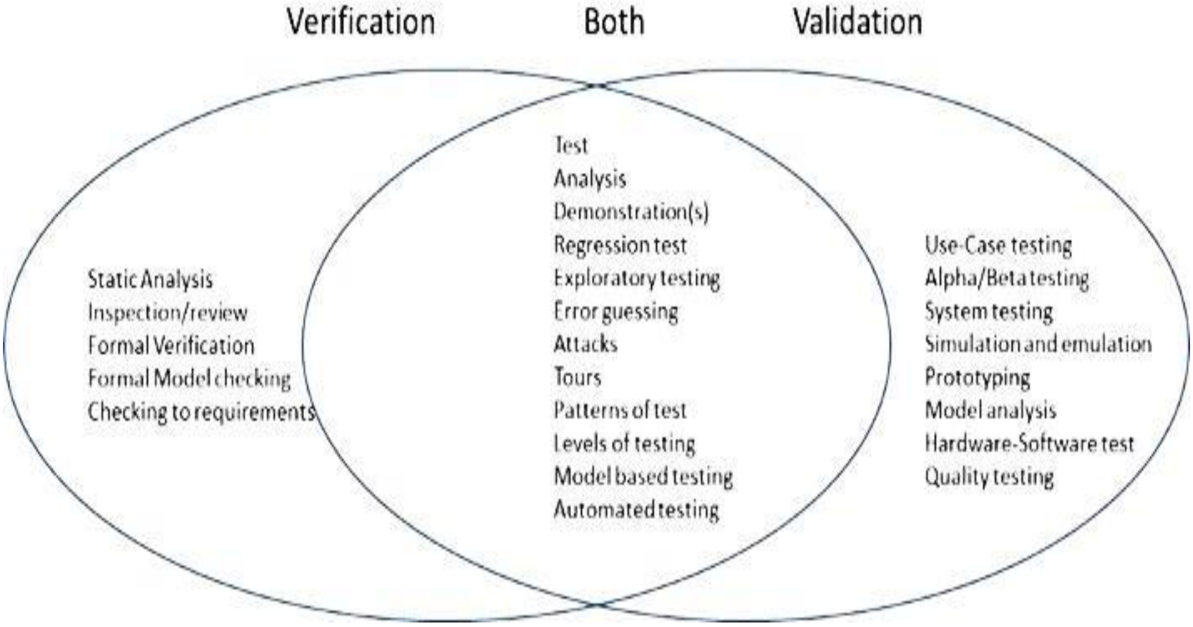

Figure 3 – Example activities of V&V/IV&V [1]

For IoT, the key V&V activities from Figure 3 are: static analysis and reviews of the code, testing, analysis using modeling, exploratory testing, attacks to break the system, model-based testing with automation, quality(s) assessment with security testing, simulation, and system assessment with hardware in the loop driven by use-cases or acceptance driven testing. Verification teams use these activities to focus on functional and structural assessment. For IoT validation, teams are assessing the system using V&V activities addressing the stakeholder needs. A skills list for these V&V activities includes [19,20,21] and those found in the following list:

- Hardware engineering and testing/assessment
- Software engineering and testing/assessment
- Systems engineering and testing/assessment
    - Operations and deployment of particular AI and data analytics
- Identification of potential project/test risks and impacts
- Documentation and technical communication of testing/assessment
- Working knowledge of programming as related to test automation and analysis
- V&V processes of [3]

4. Security testing and assessment will be a top task

Having a security plan, outlining risks and mitigation plans is an excellent start for any project. A variety of articles have cited that IoT systems are ripe for hacking. While there are many important qualities to plan for and test, currently, security testing should be near the top of the list. Figure 4 gives a starting point for security test plan assessments and designs.

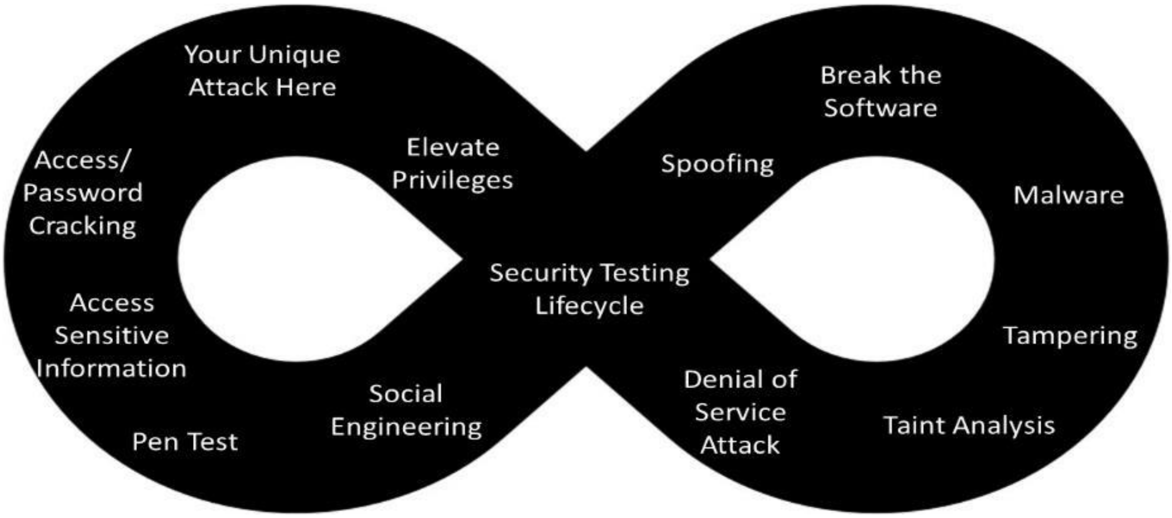

Fig 4 The security infinite test diagram – you are never done with security [1] testing

The tests in Figure 4 are common security test approaches [1,18]. Security testing is a specialized area that IoT teams will need to plan for ahead of time. The bad guy hacker depends on development teams or IV&V teams not doing a good job on security testing. Hearing your company's name broadcast on the nightly national news is certainly not good for any business. Given the "internet" connection part of IoT, the door is always open for a hacker, and much more so than older non-connected, embedded cyber-physical systems. The security test planning starts at the IoT device, but as shown in Figure 2, testing includes the edge and cloud. The use of AI and data analytics will help the security testing, but all of these levels of testing and analysis will need to work together to secure IoT. From here, I define the topics of Figure 4 beginning where the hackers might.

- Social Engineering [1]

Testers need to think and act like the "bad guys" in social engineering. They ask: "How can we trick the users and system?" Testers must think the way the hackers do, but before the bad guys can cause problems. An excellent model to follow is the cyber kill-chain model [1]. *I remind you to be legal and ethical* (white hat). Also, readers will note that in the IoT security model of Figure 4, blanks for "your unique attack" may need to be customized to your local threats and risks found during social engineering.

- Penetration (Pen) testing [1]

Pen testing is where the security tester attempts to penetrate the IoT system's security measures. Pen test results include testing software, issues, reports on what was done, and prioritization or guidance on untested areas. The tester should think of this security Pen testing as a service on software and a collaboration between the teams doing development. When the Pen results come through, developers can then seek clarification and advice from testers and V&V and the security team on the issues and how best to resolve them. I recommend outside testing groups or V&V conduct Pen testing when risks justify it.

- Access Sensitive information [1]

This activity is a continuation of social engineering and is part of gathering information to support security risk assessment. The hacker-tester tries to access information about the system for which they are not privileged to access. They may "play" the role of a user with limited access and then try to gain privileged access or even play an outside hacker trying to get in. Testers should try to access the IoT device, edge, fog, development areas, and even the cloud. Testers also look for data and information used in later testing and attacks in this information gathering phase.

- Access control and password cracking

A favorite process that is a very productive subset of Pen testing is password cracking or gaining system control access. To do this, there are tools such as PKcrack [1]. Even worse, many IoT systems use default passwords and tools such as those found on the website Shodan™ [1], exposing these systems to anyone looking to do a hack. Why? Because the bad guys will do this.

- Elevate access privileges [1]

After the Pen or cracking testing works and the tester-hacker is within the IoT system, they should gain elevated access privileges. This will expose vulnerabilities and can provide yet more information to use in security attacks and testing. Again, the bad guys will try to do this, so the hacker-tester should try to do it first.

- Denial of Service (DoS) Attack [1]

IoT systems are very attractive because many IoT devices make a large attack surface, and many of the security features of IoT devices are lacking. Bad guy hackers will attack all IoT systems; therefore, you will want your system to respond as "securely" as possible (safe-secure engineering is a design issue that can be addressed in an STA [1, 16, 19, 20]). The bad guy hackers will start by attacking the company's systems, including IoT devices. Hackers will start with easy kills, like DoS on IoT. A DoS attack attempts to block the IoT site operations by sending many requests that exceed the network bandwidth and system responsibility.

- Security Taint Analysis [1]

Security taint analysis is associated with malware, code faults, and a DoS to some extent. It is the process of staff checking the flow of user input in the IoT inputs to determine if unanticipated input can negatively affect program execution during each functionality step. The staff doing this analysis can include developers, security staff or testers. The analysis staff will "play" different roles and attempt to detect disruption of normal input processing flow. Taint analysis can help check the IoT attack surface and define new avenues of attack. The test staff should know the following kinds of information, which can be tainted: inputs, data values, sequence of inputs, messages output, file outputs, response actions, and links to other elements of the IoT system, such as the edge, fog, and/or cloud. During the analysis, the staff "user" tries to corrupt (taint) the inputs and then reviews outputs for improper actions and outputs. The output could be reviewed in real time, but the output must also be post-processed. Post-processing can be enhanced if a dynamic runtime taint monitoring or AI system [1] is included in the taint configuration.

- Tampering Attack [1]

In tampering, testers try to "break" into the hardware and/or software to gain access. This is closely related to password cracking but takes other approaches such as inserting code, data or hardware into an IoT configuration (for configuration management (CM)/software configuration management (SCM) tools and/or processes). The IoT tampering attack can modify parameters exchanged between the device and edge, fog, or cloud. The goal is to manipulate IoT data such as passwords, output values, permissions, etc. The tester is trying to understand the following:

  - Does the external CM/SCM system or internal version checking detect tampering?
  - Does the system itself notice and inform stakeholders or provide notification of the attack?
  - Can bad data be input, output or used?

- Malware and Security Attacking the Off-the-Shelf (OTS) Software for Malware [1]

Most IoT systems will include OTS software from open sources or procured from third parties. OTS introduces security risks. The use of tools and analyses to support testers when conducting this attack is a good idea. When you do not trust the OTS software, conducting these scans and analyses may help, in addition to running security test attacks. In this analysis, the tester will need access to the source or binary code using a tool for the OTS. The OTS code should be visually checked during an inspection meeting by a team, but humans will still miss things, so tools are also recommended.

- Spoofing [1]

A spoofing attack uses the information to impersonate an authorized device or user. Once in the system, spoofing can: steal data, insert malware, or bypass access control systems for later access. For IoT systems, connections can be spoofed, including:

- Website or Domain name for IoT edge, fog or cloud services
- IP addresses
- Address Resolution Protocol
- GPS spoofing of the location
- User (man, system, hardware)-in-the-middle
- Identity of user or login
- Report how hard or easy it is to read the file (unformatted clear ASCII text is bad; encrypted data is much better since it is another stumbling block for a hacker to overcome).

5. Break the Software

For test methods to “break the software” security see the classic book references [4, 8] which provide many security test attacks.

6. Security testing needed beyond a single IoT device

These security assessment elements include the hardware, system, and other IoT devices, edge, fog, and cloud when dealing with vulnerabilities. These assessments are common with a single IoT device, but I advocated that the security threat assessment must continue at a system level for IoT. While ownership of these other levels of IoT security testing may not be clear, the device IoT test team should at least bring risks forward so stakeholders can make informed decisions after the single device is assessed.

7. Skills for Security Assessments

A sample skills list for security assessments includes [1,16,18,19,20,21] and the following:

- Software engineering
- Systems engineering
- Partners and suppliers of support products from a security risk viewpoint
- Test governance, standards, certification, and guidance
- Testing of quality characteristics, which are essential to the security of IoT systems
  - Maintainability
  - Performance
  - Interoperability and integration
  - Reliability
  - Security attacks
  - Dependability

8. AI and DA Testing Assessments for IoT

AI and DA will be essential engineering areas for IoT systems. The first use many people think about for AI and DA is helping users and improving sales (for marketing). However, this simplistic view overlooks the important areas of development and test activities. The IoT device will feed massive amounts of information into AI and DA. We already see this in the web world. The edge and cloud will collect and analyze data and feed AI and DA. Testers that want to improve their assessments. Testing and product quality areas are advised to learn and use AI and DA. Since the data values will be large, tools, STAs, AI and DA must be incorporated into an integrated system, as shown in Figure 5. Ever-improving AI and DA should drive the testing and assessments over the lifecycle. Testers need to do the top-level test planning and product quality assessments. Manual testing will become a minority activity as test automation, AI and DA use increases.

In Figure 5, AI and DA are part of a constant flow of information from the branches of the IoT system, device, edge, and cloud, into development, test, and as needed, V&V/IV&V. Continuous lifecycle development, integration, test, and deployment is part of this really important model for IoT. Testers and companies that do not learn new skills and improve the IoT systems based on the AI and DA feedback will become like the programmers of FORTRAN and Cobol systems i.e., dinosaurs (if you are not sure what those terms are, search the Internet).

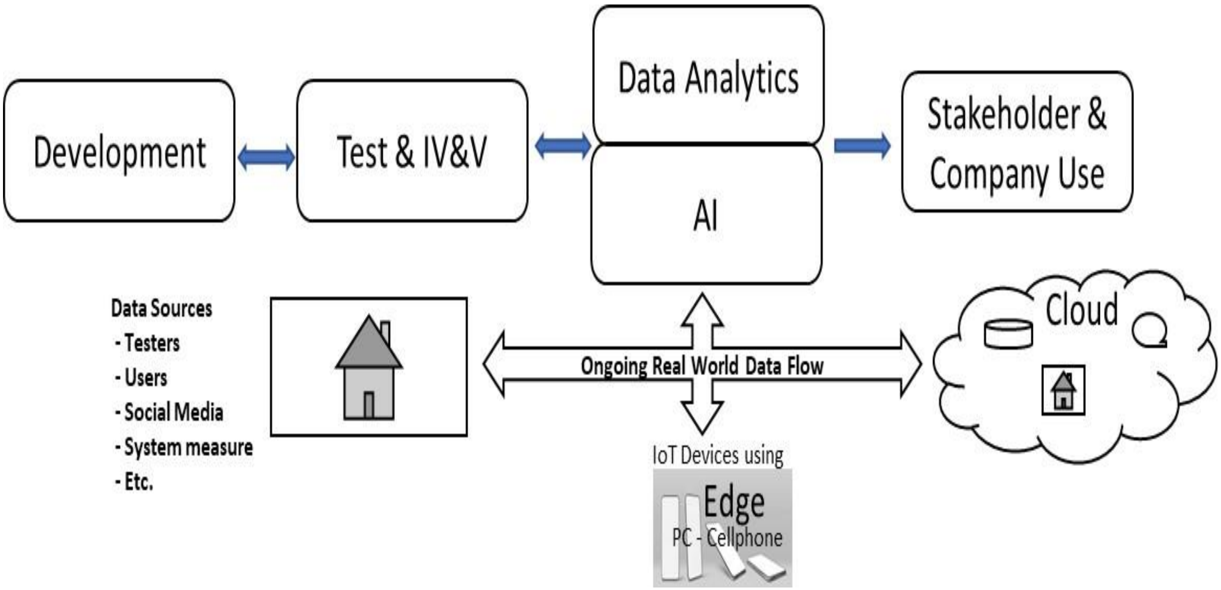

Figure 5: AI and data-driven development and test

A continuous process of improvement and development will rely on the information coming from AI and DA. The massive data millions of IoT devices and systems can generate exceeds human capabilities. The data flow from the real world of local IoT devices (the home), the edge, and the cloud requires constant analysis. While based on statistical tools and concepts, DA will need AI support. Some of the flow of AI and DA information to stakeholders and companies will be for things like improving sales and help desk support. At the same time, teams cannot forget the flow of AI and DA information into development efforts as well as to testing, V&V and IV&V. IoT products will need to be iterative as updates and new product features are implemented. Many IoT teams may forget to include development and test in the information coming from the AI and DA flow by choosing not to update or improve products. This will be a mistake. In particular, test teams doing assessment and V&V activities will need to ensure they are part of this information flow while learning new technical skills. It is important to consider that V&V and IV&V will need to include AI and DA modelling, as well as product improvement beyond what the basic, initial requirements dictate.

For IoT systems, these assessments will need to move beyond just running regression tests. For example, as the test architect on a project, I used data from the test issue database and help desk reporting systems to conduct system-level reliability and security analysis during product usage. This allowed the development team to estimate when failures or threats might occur and identify areas for improved development and testing. While development management did not like to hear about these analysis predictions and increased costs, it allowed them to allocate more resources toward process and product improvements while staying business.

Skills list for these AI and DA activities include [1,19,20,21]:

- Systems engineering
- Knowledge and familiarity with test framework tool sets (e.g., unit test; automation; AI, data analytics, STAs with continuous integration, test, and deployment frameworks)
- Statistics and data analytics math concepts as applied to IoT testing
- Understanding AI processing, tools and analysis
- Creating software test architectures (STAs)

9. Future work

The nature of IoT will unfold and evolve the way general IT and the web did. The impacts of AI, DA, system-level V&V/IV&V/testing will evolve as assessments of IoT come of age, and the changing skill base of the development and test team evolve too. Teams must use analytics and AI to conduct process and product improvements. The available information needs to be applied, analyzed and, as needed, used to make new IoT systems. The ideas in this paper should be expected to be updated by analyzing home, personal, medical, societal and industrial usage of IoT.

10. Conclusion

This paper advocates the need for IoT software system-level testing beyond just the IoT device, because software IoT systems will need to go beyond the basic testing of the device to satisfy stakeholders and security requirements. Although testing the device is important, testing *only* the IoT device software is insufficient. To achieve this, testing and V&V/IV&V, particularly at the system level, will be optimal for many IoT systems. Testing/V&V should include focus areas such as security assessments, use of AI, and data analytics.

References

1. Book: IoT Testing by Jon Hagar, scheduled for publication Sept 2022, Springer Nature
2. Software Verification and Validation (V&V) by Deutsch, 1997
3. “IEEE Standard for System, Software, and Hardware Verification and Validation,” in IEEE Std 1012-2016 (Revision of IEEE Std 1012-2012/ Incorporates IEEE Std 1012-2016/Cor1-2017), vol., no., pp.1-260, 29 Sept. 2017
4. Software Test Attacks to Break Mobile and Embedded Devices by Jon D. Hagar, 2013
5. Educating Students on Techno-Capitalism’s Impact to STEM, by Jon Hagar, 2021
6. Software (Test) Architecture Elements Applied to Software Test: View, Viewpoints and Containers, by Jon Hagar, InSTA conference, 2022
7. The Art of Software Testing by Myers (the grandfather of them all), 1979
8. How to Break Software: A Practical Guide to Testing by James Whittaker, 2004
9. A Practitioner’s Guide to Software Test Design by Lee Copeland
10. Domain Testing Workbook by Cem Kaner
11. Lessons Learned in Software Testing by Kaner, Bach, Pettichord
12. Testing Computer Software by Kaner, Falk, and Nguyen, Van Nostrand Reinhold, 1993
13. Systematic Software Testing by Craig and Jaskiel
14. “IEEE/ISO/IEC 29119, ISO/IEC/IEEE International Standard - Software and systems engineering --Software testing” –Parts 1 to 5 series
15. Software Test Automation by Fewster and Graham
16. IEEE 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability, in revision as of 2022
17. Smart Santander: IoT Experimentation over a Smart City Testbed, Luis Sancheza et al, 2020
18. How to Break Software Security by James A. Whittaker and Hugh Thompson, Addison-Wesley Professional, 2004
19. ISO Standard 15288 Systems and software engineering — System life cycle processes
20. ISO Standard 12207 Systems and software engineering – Software life cycle processes
21. ISO 9000:2015 Quality Management Systems (series)- Fundamentals and Vocabulary