

Software Test Challenges in IoT Devices



Jon D. Hagar, Consultant, Grand Software Testing
embedded@ecentral.com





The IoT Opportunity

- Challenges in both hardware and software development
 - Can we produce quality within schedule and cost constraints?
 - Merging of physical, cyber, and networked worlds
- All the problems of IT and Mobile Software
 - It only takes a few minutes of using an App before users like or dislike it
- Worse than that. . .
 - IoT can Kill: You may be on the nightly news (bad press is not good)
 - Companies want a piece of the IoT pie (4-10 trillion USD in next 10 years)





My Top IoT Challenges

(and which ones I cover in red)

- Complex software and hardware (for testing)
 - Sensors and the “real world”
 - How to conduct development
- Numbers of devices and configurations (and how to test)
 - Configurations and compatibility
 - Reliability and fault tolerance
- Big data and analytics
- Privacy and Security
- Connectivity (systems and systems of systems)
 - Integration
- Safety
- Life cycle – unified hardware-software dev-test-ops (a dream)
 - Tools to support development, ops and tests
 - Cost and schedule
 - Concurrent software and hardware development
- Integrated Operations - given the above
- International standards for devices and protocols



Basic Definitions

- Test – *the act of conducting experiments on something to determine the quality (s) and provide information*
 - Many methods, techniques, approaches, levels, context
 - Considerations: input, environment, output, instrumentation
- Quality(ies) – Value to someone (that they will pay for)
 - Functional
 - Non-functional
 - It “works”
 - Does no harm
 - Are there (critical) bugs?

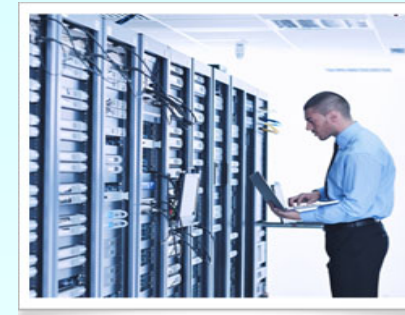
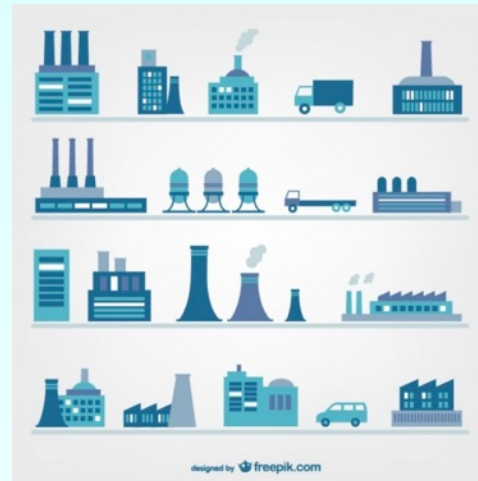




TECHNOLOGY SPACE



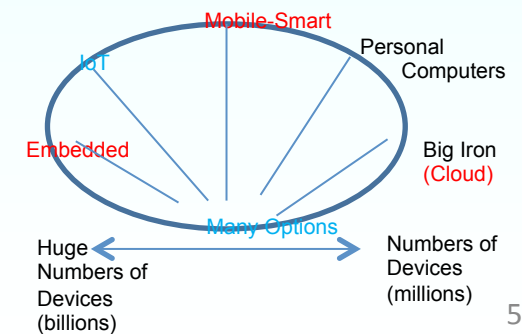
Physical
Systems
(circa 100,000 BC)



Cyber
Systems
(1950s)

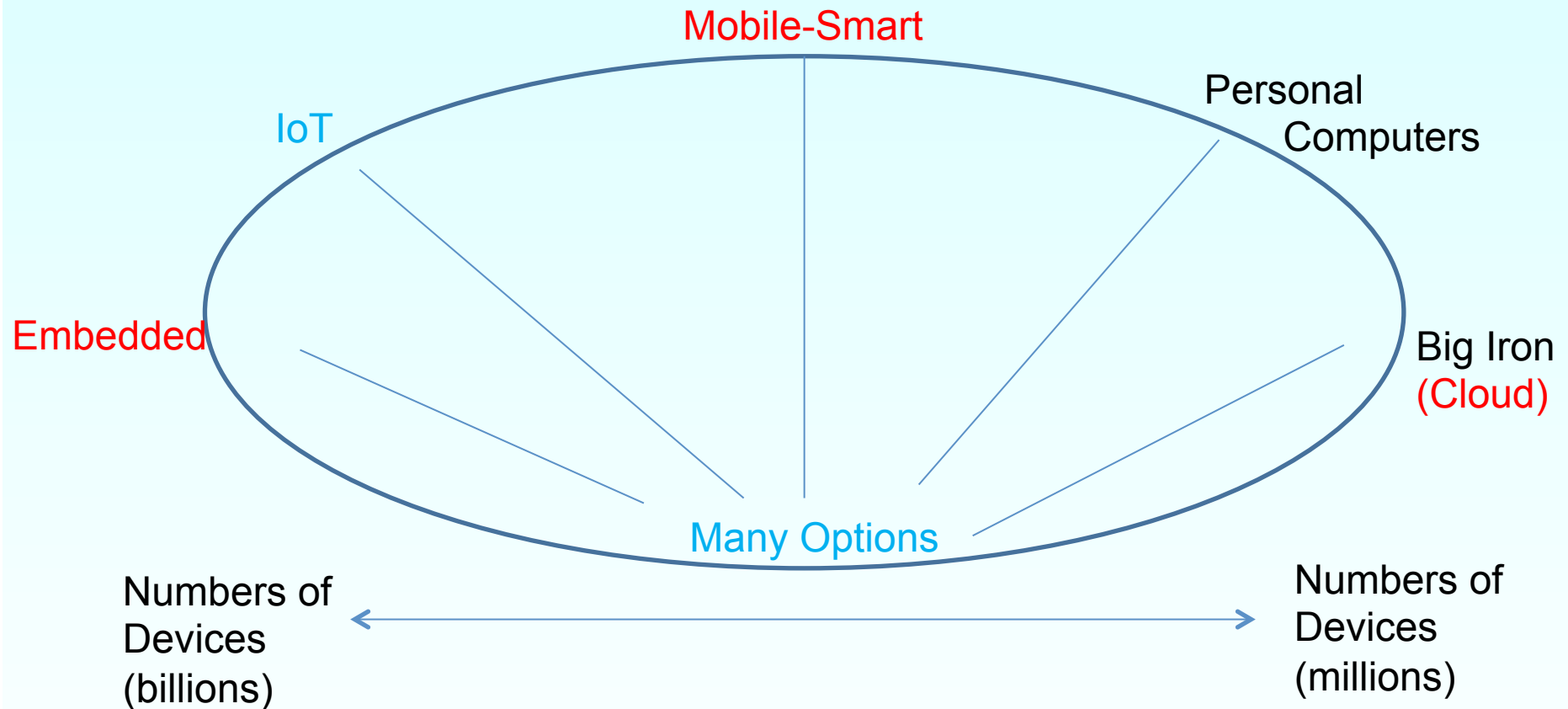


Cyber-Physical
Systems (today)





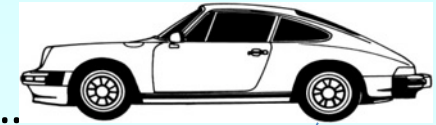
Where are IoT Devices in Computer Space?





What is an IoT Device

- Embedded – Software contained in “specialized” hardware...
 - Minimal networking-communications



Test Brakes

PLUS

- Mobile and handheld smart devices—small, held in the hand, highly connected (web, cloud, servers,...)



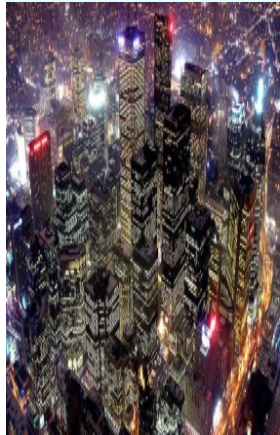
- IoT – Internet of Things are “traditional” and new devices with software and communication added





IoT – 2.5 Main Segments (to name a few)

Industrial 4.0



Cities/States/Nation

- Health
- Safety
- Info
- Control and Monitor

Transportation

- Vehicles
- Navigation
- Logistics

Worksite/Factories

- Ops
- Control
- Info



Industrial Value
2x Consumer

Middle



Vehicles

- driverless
- monitoring
- Infotainment

Office

- Security
- Energy
- Worker info

Medical

- Health monitor/control
- Records

Retail

- Ordering
- Checkout
- Advertize



Consumer

Home

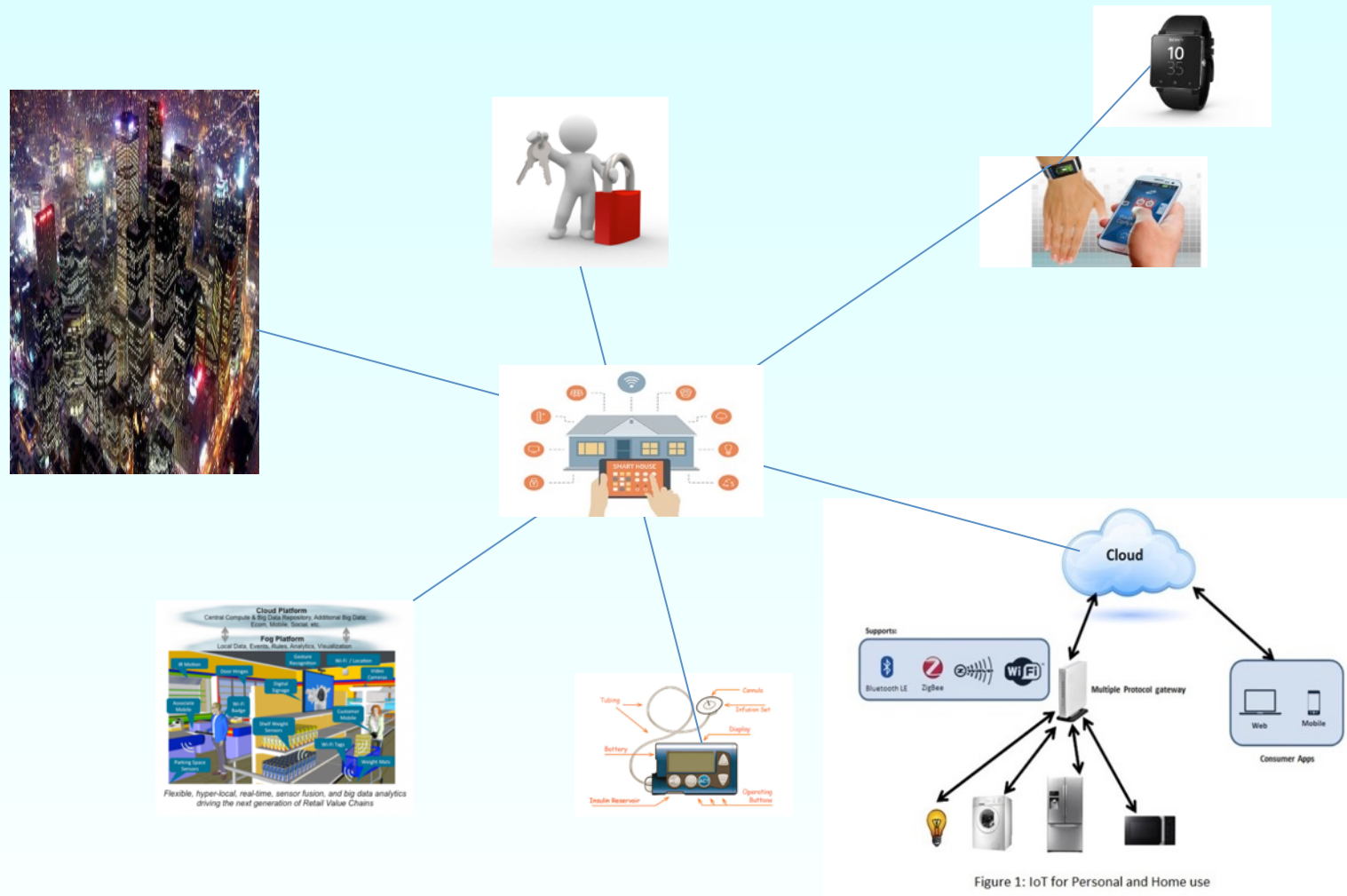
- Security
- Monitoring & control
- Infotainment



Human

- Health
- Fitness
- Info

Must be Interoperable
Across segments





Is IoT new?

We have had embedded, control, M-2-M, the internet

Why is IoT so different?

It brings together:

- Connectivity
- Big data
- Resource limitations – size, batteries, processing, memory, other
- Numbers and types of devices
- Mixes cloud, PC-IT, mobile, embedded, network, and user
- Security and privacy

In Parallel and Supporting Development





IoT Testing Opportunities

- Requirements verification checking
 - Necessary but not sufficient
- Risk-based testing
 - Historic but tried and true
- Pattern or **attack**-based exploratory tests
 - Pattern 1: Model-based testing
 - Pattern 2 (and Challenge 2) : Math-based testing
 - Pattern 3: Skill/experience-based testing
 - Pattern 4: Standards/process-based testing





Pattern 1 : Model-Based Testing

- Address systems, software, and hardware test
- Developer and Independent Modeling
- Improved Understanding



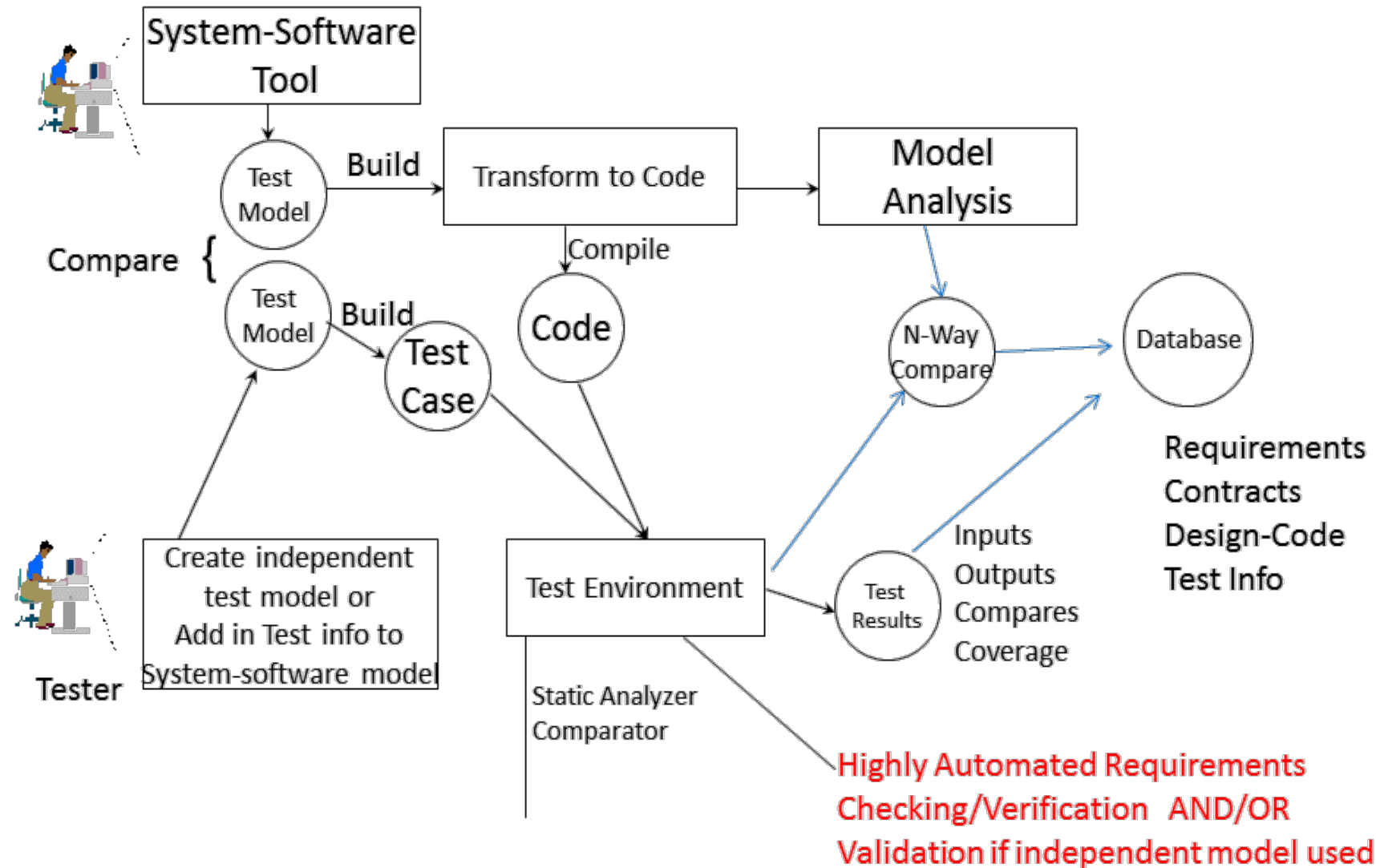


Model-based Testing in IoT

- Interest and use of model-based testing is growing in industry segments
 - Telecom, finance, automotive, aerospace
 - European and USA interests
- IoT “high integrity” areas will need it
- Model-based testing can support:
 - Generation of test cases from models into test automated execution engines directly using scripts or through the use of keywords
 - Early testing with improved understanding of the system and risks
 - Use of models to support simulations to drive test environments
 - Verification via compares between development and test models
 - Generation of test result oracles or judges
 - Support of independent testing such as Independent V&V (IV&V)
 - Model analysis and formal verification



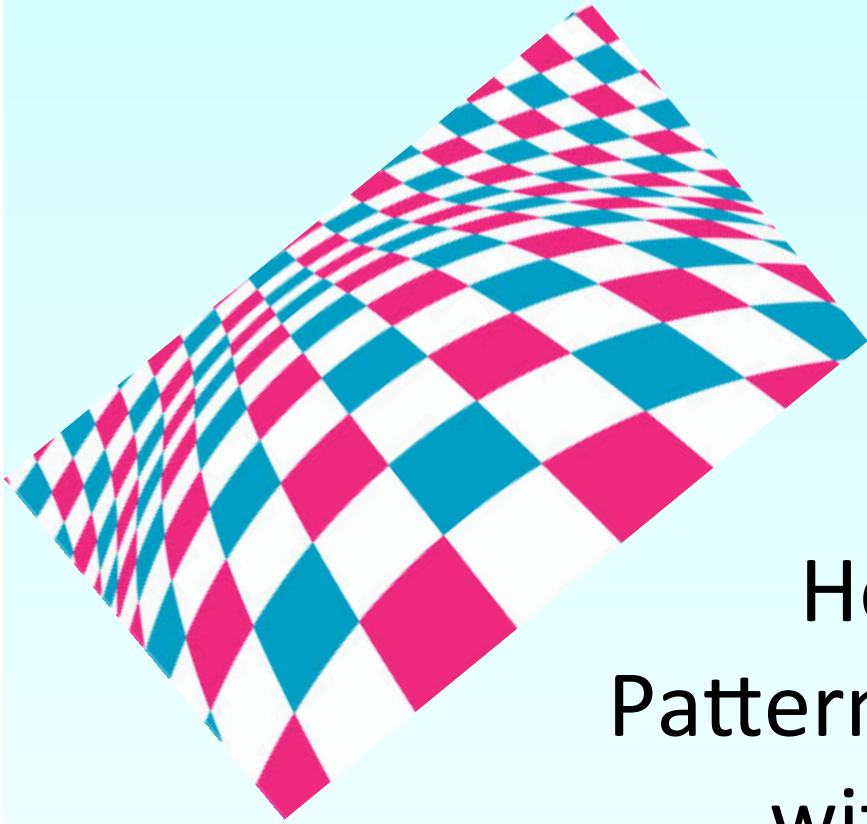
An Example Test Flow with Modeling for IoT





Model-based Test Advantages and Considerations

- UTP 1.2 (soon 2.0) standard in place
- Tool support in place
 - Produce test automation
 - Graphic views aid understanding
 - Serve as an oracle
- Aids in avoidance and/or identification of issues early in lifecycle
 - Before code or hardware complete
- Considerations for growth and continuing usage
 - N-version problem
 - Self-checking problem if only one model is created
 - Skilled modelers and testers needed
 - Correct development/test environment must be in place

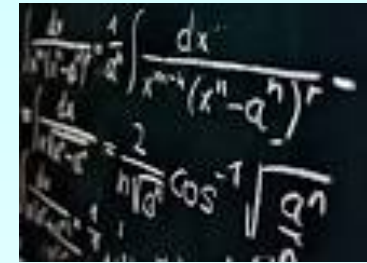


How to Address Pattern 2 and Challenge 2 with one concept

*How do we handle
many configurations,
options, and even test
data sampling?*



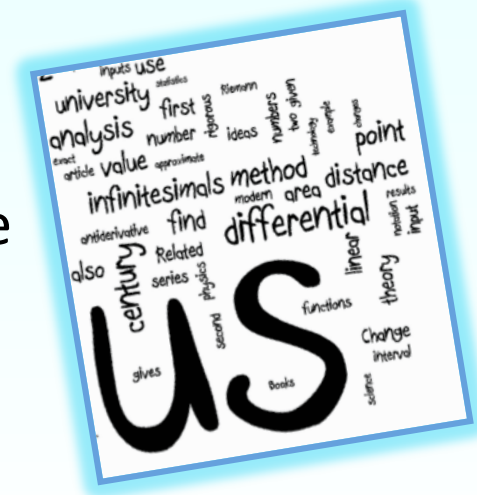
Pattern 2: Math-based Testing



Testing is a sampling problem:

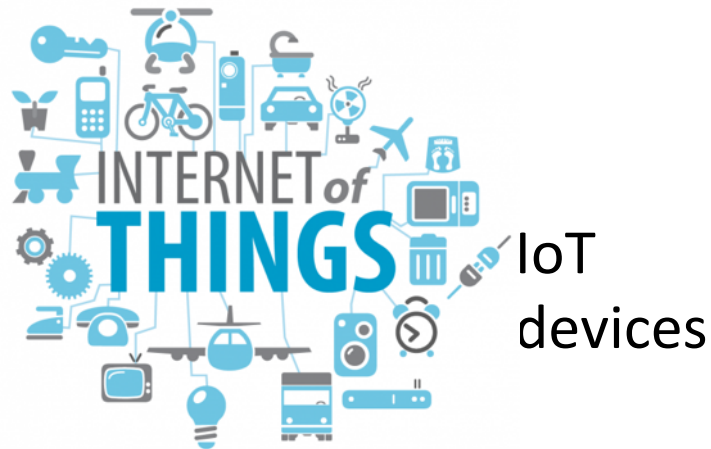
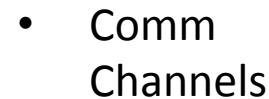
How can testing use Math to help?

- Test systematically the numbers of devices, configurations, networks, etc.
- Sampling in environments and quality control
- Selection of data from the input domain space
- Big Data analytics fed into testing





- Andriod



edureka!

www.edureka.co/software-testing

to address, data, configurations, devices
Comms, resources, integration, resources

19



Using the ACTS Combinatorial Tool: Example

Parameters:

Andriod AppPlatform

[Device 1, Device 2, Device 3, Device 4, Device 6,
Device 7, Device 8, Device 9, Device 10]

IoTProtocolHome

[true, false]

IoT Devices

[Refrig, Stove, mircrowave, TV, front door,
Garage door, Home gaurd, Stereo, Temp Control,
Lights, Drapes, Water Heater, window openers]

Routers

[0, 1, 2, 3, 4, 5]

Comm providers

[Cell1, Broadband, cable, Cell 2, Space based,
Vendor godzilla]

Data

[1, 0, -1, 99999, -99999, 100, -200]

Test Case#	Andriod AppPlatform	IoTsHome	IoTDevices	Routers	Comm providers	Data
0	Device 1	false	Refrig	1	Broadband	0
1	Device 2	true	Refrig	2	cable	-1
2	Device 3	false	Refrig	3	Cell 2	99999
3	Device 4	true	Refrig	4	Space based	-99999
4	Device 6	false	Refrig	5	Vendor godzilla	100
5	Device 7	true	Refrig	0	Cell1	-200

→ 119 Test
Sample



Statistical Math Tools

General Technique Concept	Tool Examples	Examples of where technique can be used	Specific sub- technique examples
Combinatorial Testing	ACT, Hexawise rdExpert PICT	Medical, Automotive, Aerospace, Information Tech, avionics, controls, User interfaces	Pairwise, orthogonal arrays, 3-way, and up to 6 way pairing are now available
Design of Experiments (DOE)	DOE ProXL DOE++ JMP	Hardware, systems, and software testing where there are "unknowns" needing to be evaluated	Taguchi DOE
Random Testing and Fuzz testing tools (security)	Random number generator feature used from most systems or languages	Chip makers, manufacturing quality control in hardware selection	Testing with randomly generated numbers includes: fuzzing and use in model-based simulations
Statistical Sampling	SAS	Most sciences, engineering experiments, hardware testing, and manufacturing	Numerous statistical methods are included with most statistical tools
Software Black box Domain Testing	Mostly used in manual test design, though some tools are now coming available	All environments and types of software tests. These are "classic" test techniques, but still underused	Equivalence Class, Boundary Value Analysis, decision tables

Pattern 3: Skill/Experience-based Testing



Exploratory Testing - Definition



- Quoting James Bach: “The plainest definition of exploratory testing is test design and test execution at the same time. This is the opposite of scripted testing (predefined test procedures, whether manual or automated). Exploratory tests, unlike scripted tests, are not defined in advance and carried out precisely according to plan.”

http://www.satisfice.com/articles/what_is_et.shtml



Exploration: An Important Skill for Testers

- Some people think that all testing is exploratory
- Scientific methods
- Used at different times
 - Early
 - Performance
 - On Hardware
 - Late
- Based in patterns of **attack**





Pattern-based Testing

*What is an **attack**?*

- A pattern (for testing) based on a common mode of failure seen over and over
 - Maybe seen as a negative, when it is really a *positive*
 - Goes after the “bugs” that may be in the software
 - May include or use classic test techniques and test concepts
 - See Lee Copeland’s book on test design
 - See many other good test books
- A Pattern (*more than a process*) which must be modified for the context at hand to do the testing
- Testers learn mental **attack** patterns when working over the years in a specific domain

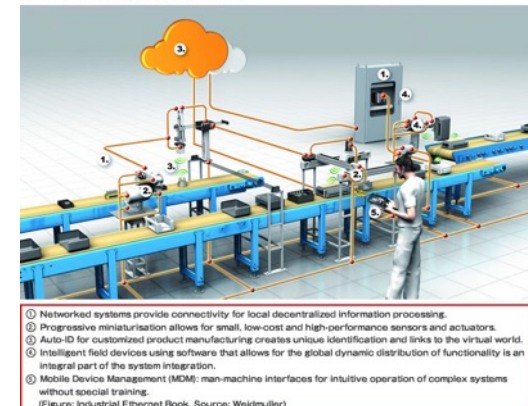




Hardware Test Planning with Exploration Concepts

- Verification checking (tests) of requirements is common
 - Expected necessary testing, but by itself is not sufficient
 - Singular focus of many test teams that misses errors and needed information
- Incorporates risk and **attack** testing within exploratory, experience-based test planning
 - Allows rapid test exploration without limitations of highly scripted tests
 - Requires “skilled” test teams
- Exploratory testing must be balanced with strategies
 - Verification and Validation using standards
 - Math-based
 - Model-based

Figure 1: The Industry 4.0 factory organizes itself based on a network of communication-capable components.





Exploratory Testing In IoT

- Rapid feedback
- Learning
- Upfront rapid learning
- **Attacking**
- Address risk(s)
- Cover data
- Reliability
- Performance
- Independent assessment
- Target a defect
- Prototyping
- Need info for developers
- Test beyond the requirements
- Cloud
- Fault Tolerance



More Examples Software Attacks for Exploratory Testing

Excerpted from "Software Test Attacks to Break Mobile and Embedded Devices"

<u>Software Test Attack Type</u>	<u>Attack Finds</u>	<u>Notes on the Attack</u>
Developer level attacks	Code and data structure problems	Almost a quarter of errors in mobile and embedded can be found by structural testing
Control system attacks	Hardware and software control system errors	Many critical errors in mobile and embedded are centered in the control logic, for example analog-to-digital and digital-to-analog computation problems
Hardware-software attacks	Hardware and software interface issues	The software should be tested to work with any unique hardware
Communication attacks	Digital communications problems	Software communicates with hardware, network, and other software with complex interfaces that should be tested
Time attacks	Time, performance, sequence, and scenario errors	System software can have critical timing and performance factors that testing can provide valuable information about
User interface attacks	Problems between man and machine	The usability of devices and software are critical to success
Smart/Mobile/Hardware attacks	Issues specific to smart device configurations including cloud issues	Cloud-hybrid computing comprises a majority of the new software systems being deployed
Security test hacking attacks	Software errors that can expose devices to security threats	Security of devices or systems is increasing in importance and attacks include, for example, GPS and identity spoofing
Generic functional verification attacks	Requirements and interoperability errors	Basic checks that testers should conduct on systems and software
Static code analysis attacks	Hard to find errors that classic testing often misses	Can often be done by the development group but sometimes the test group must run this analysis

Pattern 4: Standards-based Testing

For Processes, Not Products



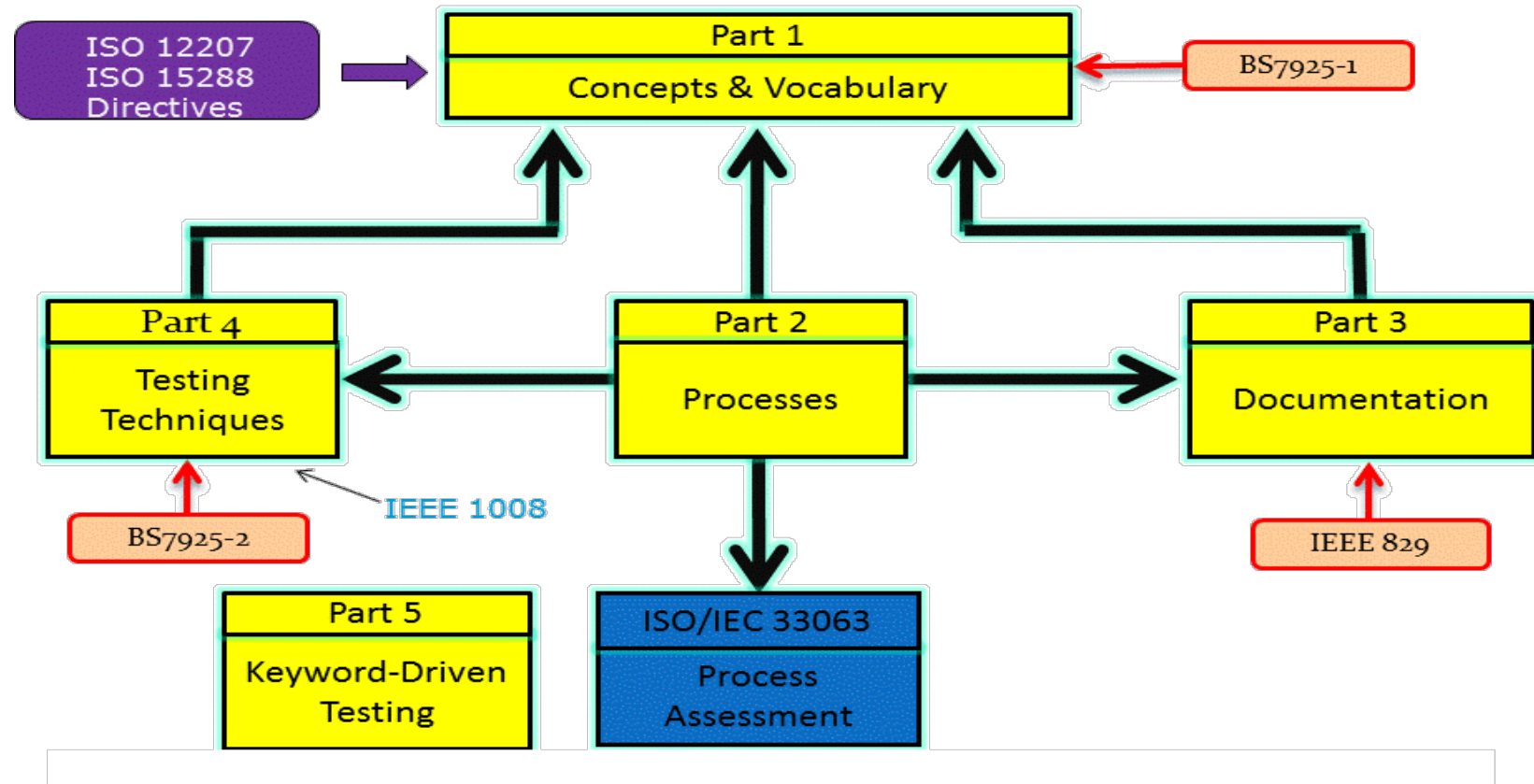


IEEE 1012-2012 Verification and Validation (V&V) Planning Standard

- IEEE 1012 is a standard that defines V&V processes
 - Specific activities and related tasks
 - Addresses V&V at system, hardware and software levels
 - Can be applied to a full system, sub-system, or element
- Features in the standard include:
 - Integrity levels
 - Minimal V&V tasks for each integrity level
 - Intensity and rigor consideration applied to V&V tasks
 - Detailed criteria for V&V tasks



ISO 29119 Software Test Standard

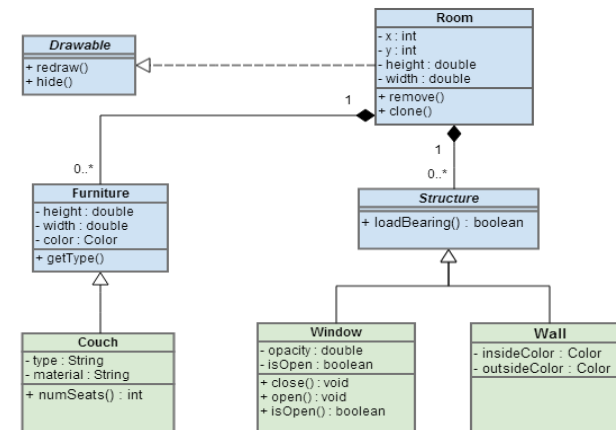




OMG UTP

- Addressed basics of Modeling earlier

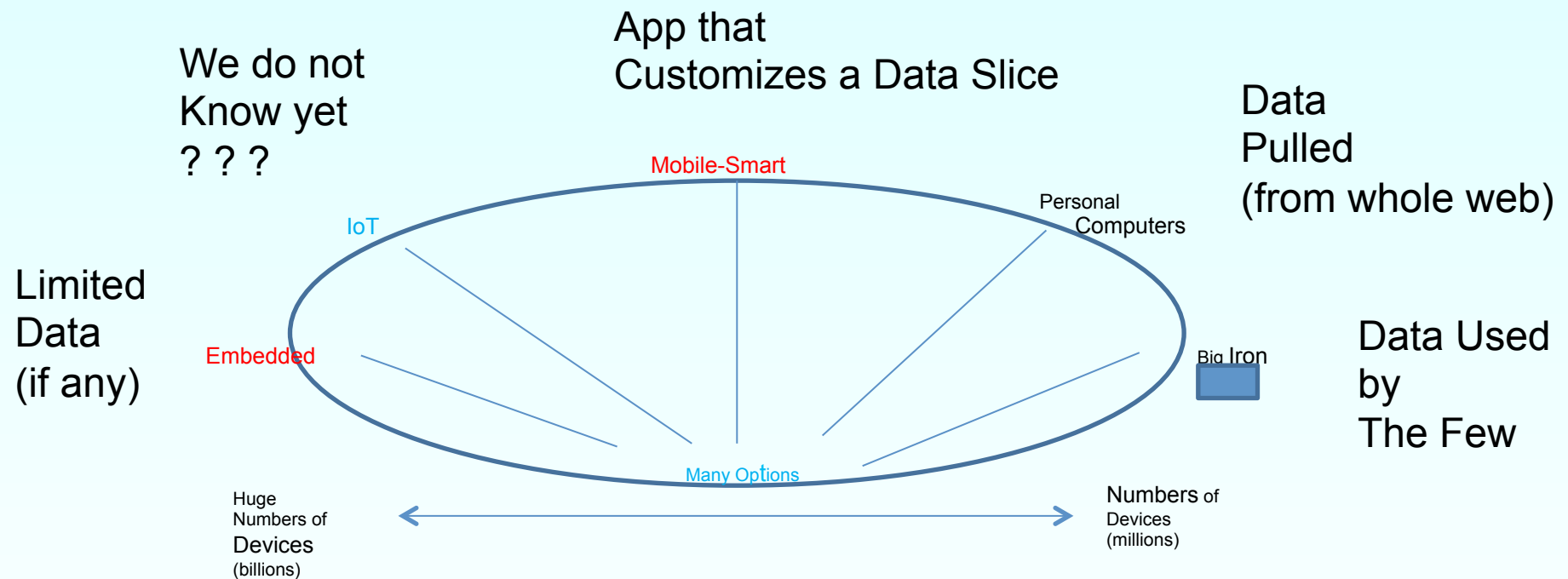
- UTP is a “language”



- ISO is considering a model-based process standard using UTP 2.0



The Evolution of Computers = The Evolution of Data Usage





IoT to Generate Huge Amounts of Data

(Petabyte, Exabyte, Zettabyte, Or a Yottabyte)

Current analytics focus is on marketing/sales

If user is a tester generating data.....

Testers will need to use data analytics

But for what?



An Example of Using Data Analytics for Testing

Software Errors: A Bad Situation to Avoid



- From Wikipedia:

Taxonomy is the practice and science of classification. The word finds its roots in the [Greek](#) τάξις, taxis (meaning 'order', 'arrangement') and νόμος, nomos ('law' or 'science'). Taxonomy uses taxonomic units, known as **taxa** (singular [taxon](#)). In addition, the word is also used as a [count noun](#): a **taxonomy**, or taxonomic scheme, is a particular classification ("the taxonomy of ..."), arranged in a [hierarchical](#) structure.



- Field data helping to “understand and know” errors to improve IoT development and testing





Error Data Taxonomy (Early version of research)

Super Category	<u>Aero-Space</u>	<u>Med sys</u>	<u>IoT / Mobile</u>	<u>General</u>
Time	3	2	3	
Interrupted - Saturation (over time)	5.5			
Time Boundary – failure resulting from incompatible system time formats or values	0.5		1	
Time - Race Conditions	3		1	
Time - Long run usages	4		1	20
Interrupt - timing or priority inversions	0.7	3		
Date(s) wrong/cause problem	0.5		1	
Clocks	4		2	
Computation - Flow	6	23		19
Computation - on data	4	1	3	1





IoT Data Analytics – One Future

SODA – Self Organizing Data Analytics



- The tools and data are organized to support all aspects of IoT with Artificial Intelligence and customized selection based on nature of user
- Users = customer, middle men, governments, developers, managers, etc.
- Research topic

Challenge 4:

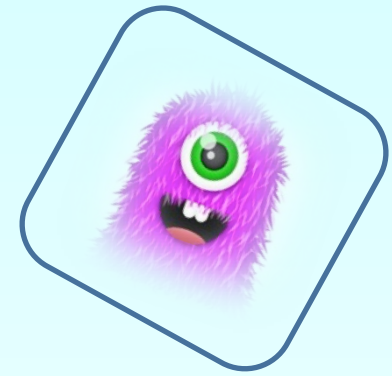
IoT Security and Privacy



(Many experts think these are top priority)



Example of What Worries Me at Night: *Security and Privacy*



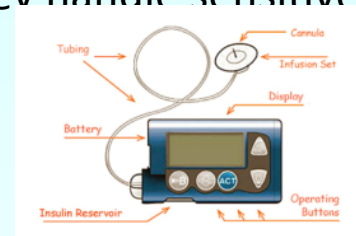
- Your IoT App gets on the nightly news
- Your team sees security as someone else's problem
- You lose personal data or your App makes personal data available to anyone





The Current Security Situation

- Mobile/IoT – IoT systems are highly integrated hardware–software–system solutions which:
 - Must be highly trustworthy since they handle sensitive data
 - Often perform critical tasks



- Security holes and problems abound
 - Coverity Scan 2010 Open Source Integrity Report - Android
 - Static analysis test attack found 0.47 defects per 1,000 SLOC
 - 359 defects in total, 88 of which were considered “high risk” in the security domain



- Cars and medical devices hacked





Security Errors

(refinement of the software error data taxonomy)

- Fraud – Identity
- Worms, virus, etc.
 - Fault injection
- Processing on the run
- Hacks impact
 - Power
 - Memory
 - CPU usage
- Eavesdropping – “yes everyone can hear you”
 - Hijacking
 - Click-jacking
 - Voice/Screen
- Physical Hacks
 - File snooping
 - Lost phone

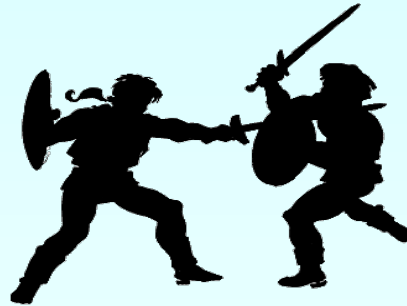


Are you giving away
someone else's keys?



Security Attacks

(from “Software Test Attacks to Break Mobile and Embedded Devices”)



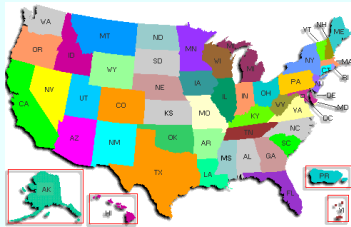
- Attack 28 Penetration Attack Test
- Attack 28.1 Penetration Sub-Attacks: Authentication — Password
- Attack 28.2 Sub-Attack Fuzz Test
- Attack 29: Information Theft—Stealing Device Data
- Attack 29.1 Sub Attack –Identity Social Engineering
- Attack 30: Spoofing Attacks
- Attack 30.1 Location and/or User Profile Spoof Sub-Attack
- Attack 30.2 GPS Spoof Sub-Attack





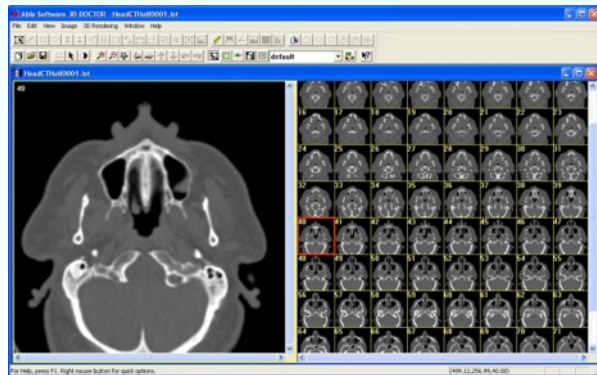
Privacy – Restricted Data

- Different from security



» More of an issue in some countries

- Examples I might not want exposed



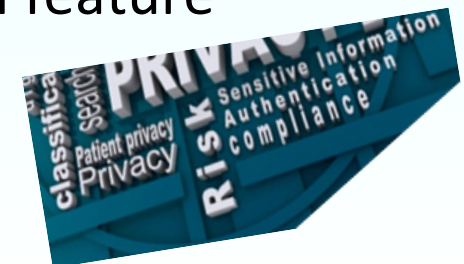


Privacy Impact on Data

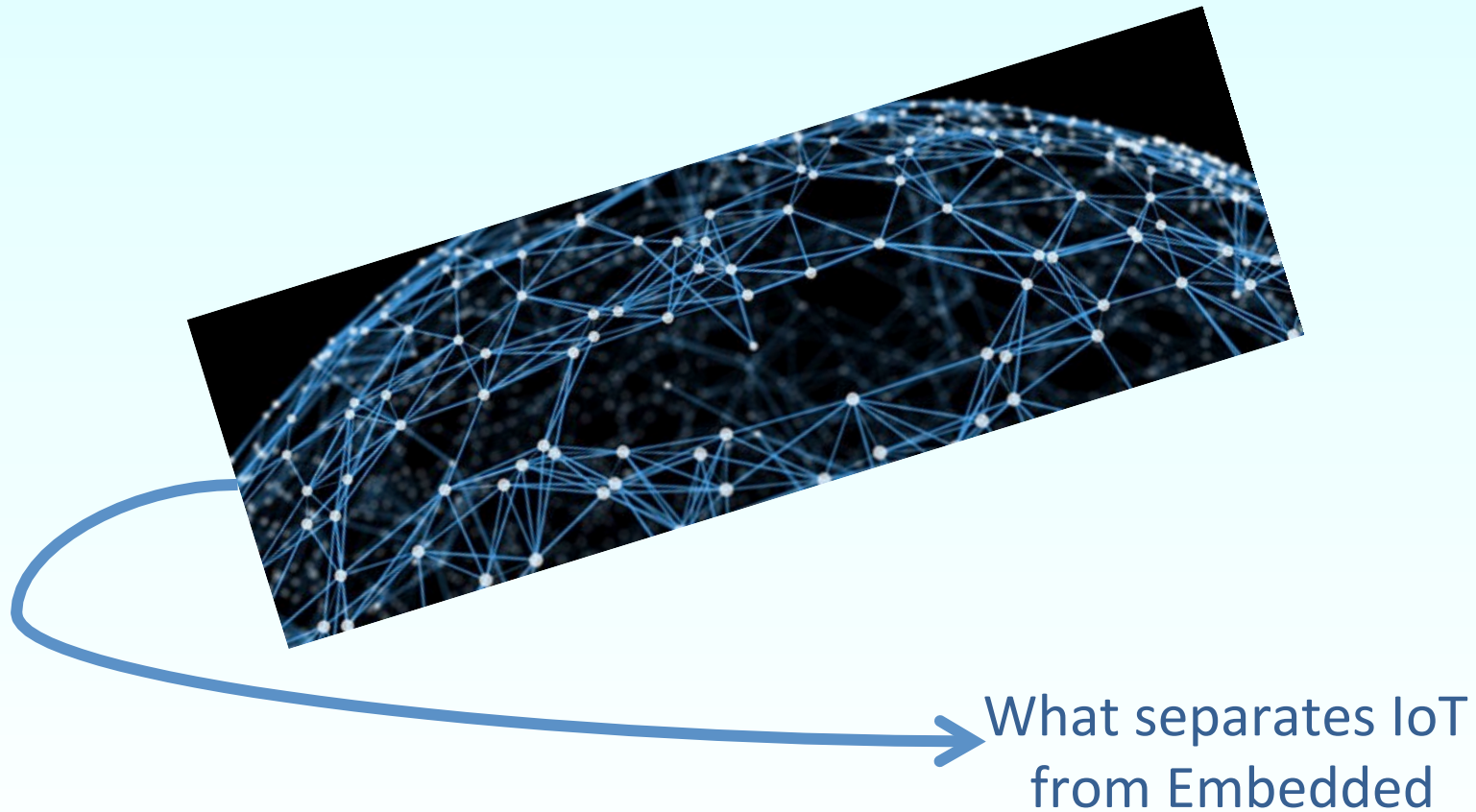
- Companies must leverage the data coming from IoT
 - Sanitize data
- Big data analytics
 - Improve Test and Dev-Ops
 - How to maintain Privacy?
- Likely will need opt in/out with “motivation feature”



ALL THIS WILL NEED TESTING



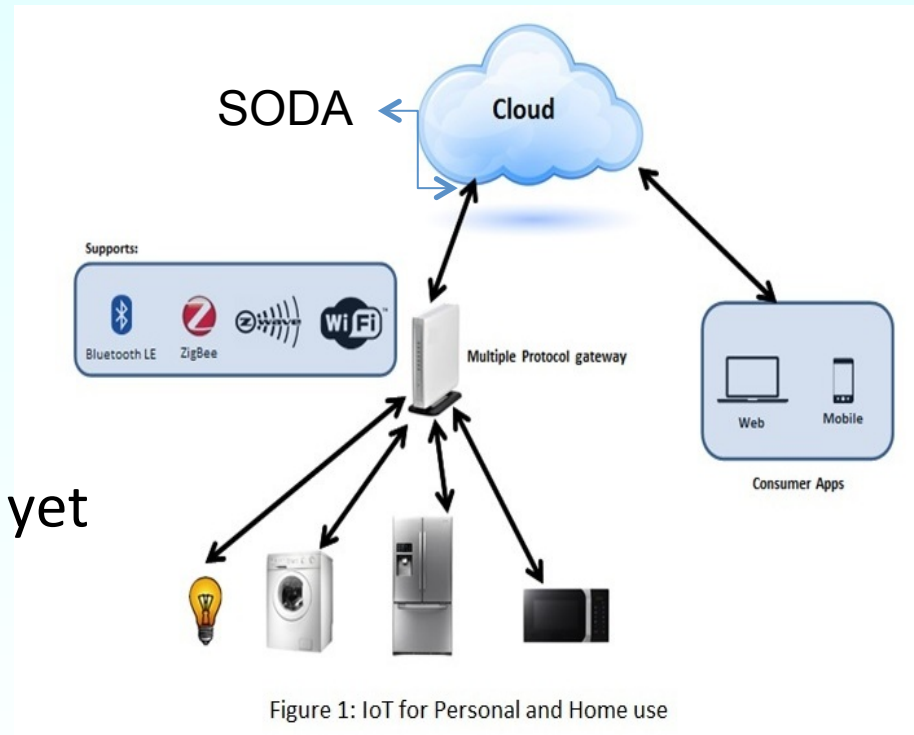
Challenge 5: **CONNECTIVITY**





Connectivity Opportunities

- Testing
 - Test the device, the network, the cloud, the app, and ????
 - Issues in connectivity
 - Security and privacy (again)
 - Protocols – no clear winners yet
 - Many options





Testing Options for Connectivity

Test Early

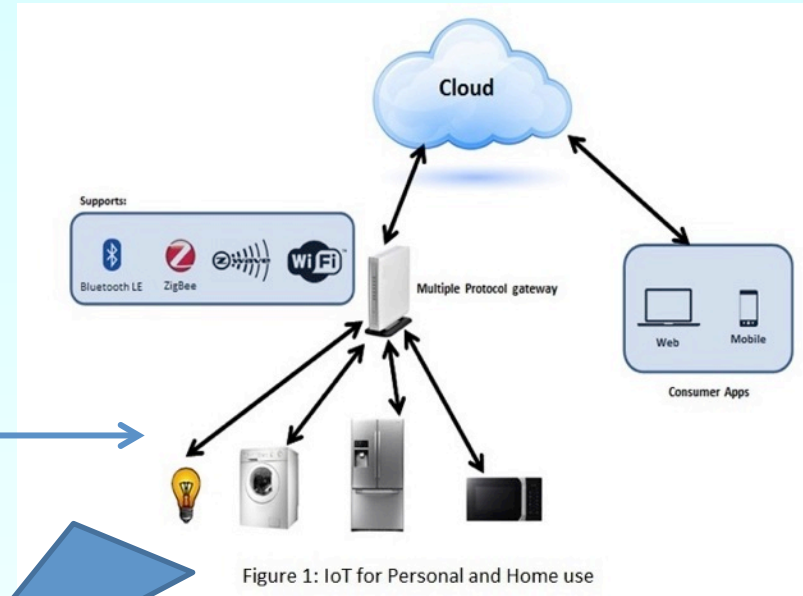
- Model-based testing
- Math-based testing

Test Often

- Test labs

Test Consistently

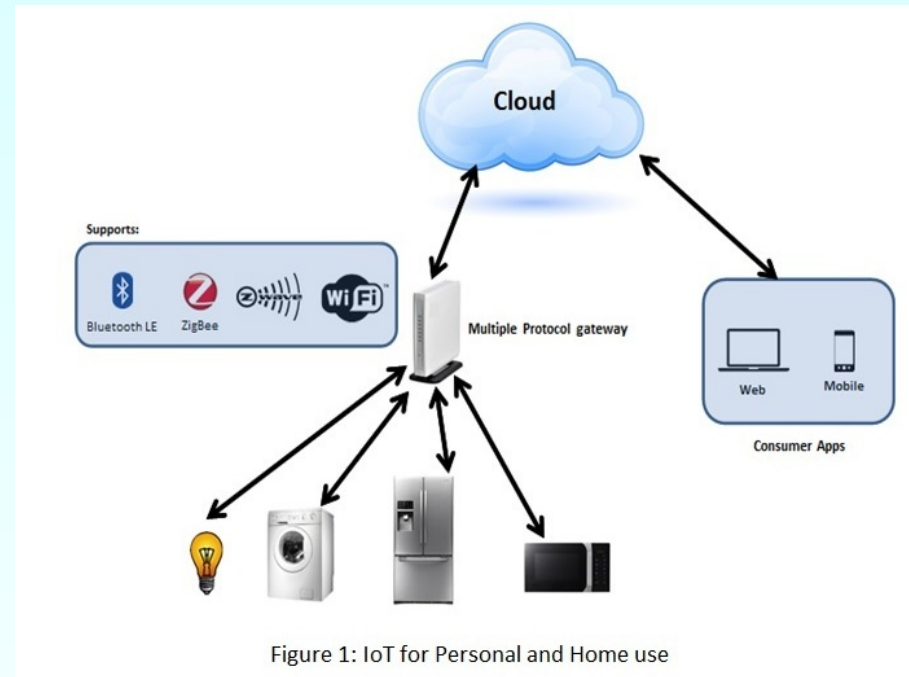
- Risk-based testing
- Requirements verification checking
- Automation





Connectivity

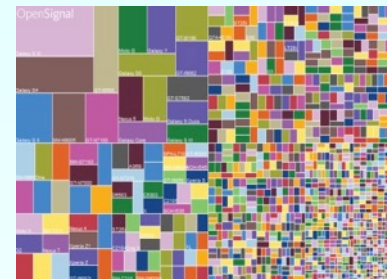
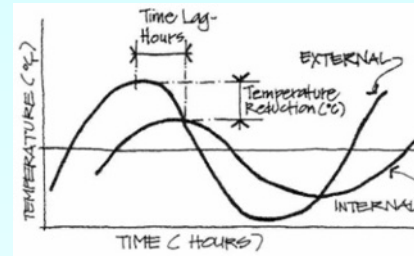
- To the User
 - Software
 - Hardware
 - Human
- To the system
- To the system of system
- To the data





Connectivity Concerns

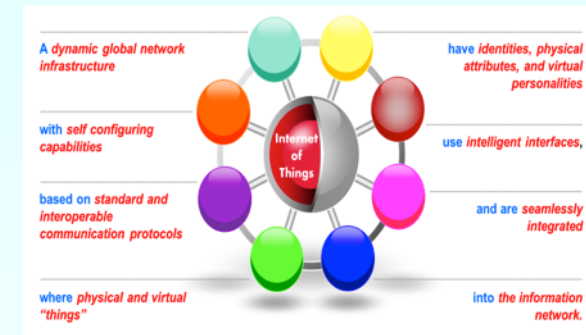
- Time lag
- Data correctness
- Different configurations
- Data completeness
- Privacy and security (yet again)





Connectivity Factors


- Test environment
- Interface Protocol options
- Device options





IoT Testing Summary



- To defeat an enemy, you must know the enemy 
- The IoT test data is limited,
 - What exists has implications
- There are challenges and patterns of opportunity
- Software will be in *very nearly everything*
 - Testing may be a limiting factor





References (my favorite books)

- “*Software Test Attacks to Break Mobile and Embedded Devices*”
– Jon Hagar
– **IoT Tests Book in 2016**
- “How to Break Software” James Whittaker, 2003
– And his other “How To Break...” books
- “A Practitioner’s Guide to Software Test Design” Copeland, 2004
- “A Practitioner’s Handbook for Real-Time Analysis” Klein et. al., 1993
- “Computer Related Risks”, Neumann, 1995
- “Safeware: System Safety and Computers” Leveson, 1995
- Honorable mentions:
 - “Systems Testing with an Attitude” Petschenik 2005
 - “Software System Testing and Quality Assurance” Beizer, 1987
 - “Testing Computer Software” Kaner et. al., 1988
 - “Systematic Software Testing” Craig & Jaskiel, 2001
 - “Managing the Testing Process” Black, 2002





More Resources

- www.stickyminds.com – Collection of test info
- www.embedded.com – info on attacks
- www.sqaforums.com - Mobile Devices, Mobile Apps - Embedded Systems Testing forum
- Association of Software Testing
 - BBST Classes <http://www.testingeducation.org/BBST/>
- Your favorite search engine
- My web sites and blogs (see front page)



References for Statistical Math Tools

- IEEE 1012, Standard for System and Software Verification and Validation- <http://standards.ieee.org/findstds/standard/1012-2012.html>, IEEE press, 2012
- ISO 29119, Software Test Standard - <http://www.softwaretestingstandard.org/>
- Hagar, J. *Software Test Attacks to Break Mobile and Embedded Devices*, CRC press, 2013
- Kuhn, Kacker, Lei, *Introduction to Combinatorial Testing*, CRC press, 2013 (includes the tool ACTS)
- Tool: Hexawise - app.hexawise.com/
- Tool: rdExpert – www.phadkeassociates.com/
- Tool: PICT – msdn.microsoft.com/en-us/library/cc150619.aspx
- Reagan, Kiemele, *Tool: DOE Pro XL - Design for Six Sigma*, Air Academy Associates, self publish, 2000
- DOE++ - www.reliasoft.com/
- SAS - www.sas.com/
- Kaner, Hoffman, Padmanabhan, *The Domain Testing Workbook*, self publish, 2013
- Bailey, *Design of Comparative Experiments*. Cambridge University Press, 2008
- Kacker, Kuhn, Hagar, Wissink, "Introducing Combinatorial Testing to a Large System-Software Organization," scheduled-2014, *IEEE Software*
- Whittaker, James 2003, *How to Break Software*, Pearson Addison Wesley
- Whittaker, James and Thompson, Herbert, *How to Break Software Security*, Pearson Addison Wesley, 2004
- Andrews, Whittaker, *How to Break Web Software*, Pearson Addison Wesley, 2006
- Levy, *Tools of Critical Thinking: Metathoughts for Psychology*, 1996
- Bach, Bolton, "Testing vs. Checking," www.developsense.com/blog/2009/08/testing-vs-checking/
- Hagar, "Why didn't testing find the embedded GM Truck fire system error?"- www.breakingembeddedsoftware.wordpress.com/
- OMG UTP 1.2, www.omg.org/spec/UTP/1.2/
- Baker, Dai, Grabowski, Schieferdecker, Williams, "Model-Driven Testing:Using the UML Testing Profile," 2008
- Green, Hagar, "Testing Critical Software: Practical Experiences," *IFAC Conference 1995*
- Boden, Hagar, "How to Build a 20-Year Successful Independent Verification and Validation (IV&V) Program for the Next Millennium," *Quality Week Conference 1999*
- [Port](#), [Nakao](#), [Katohira](#), [Motes](#), *Challenges of COTS IV & V*, Springer press, 2005