**Navigation system of an aircraft**

- An aircraft is highly dependent on the navigation system.
  - Air navigation is accomplished with many methods.
  - Dead reckoning utilizes visual checkpoints along with distance and time calculations.
  - The flight computer system aids the pilots to calculate the time and distance of the checkpoint that they set.
  - The radio navigation aid (NAVAIDS) enables the pilots to navigate more accurately than dead reckoning alone, and in conditions of low visibility, radio navigation is handy.
  - GPS is also used by pilots and uses 24 U.S. Department of Defence satellites to provide precise locational data, which includes speed, position, and track.

**Nuclear reactor safety system**

- A nuclear reactor is a system that controls and contains a sustained nuclear chain reaction.
- It is usually used for generating electricity, but can also be used for conducting research and producing medical isotopes.
- Nuclear reactors have been one of the most concerning systems for public safety worldwide because the malfunction of a nuclear reactor can cause a serious disaster.
  - Controlling the nuclear reactor system is accomplished by stopping, decreasing, or increasing the chain reaction inside the nuclear reactor.
  - Varying the water level in the vertical cylinder and moving adjuster rods are the methods of controlling the chain reaction when the reactor is operating.
  - Temperatures, reactor power levels, and pressure are constantly monitored by the sensitive detectors.
  - If a nuclear power plant system malfunctions, it can result in accidents such as radioactive leakage due to an uncontrolled chain reaction. This can result in acute radiation syndrome to the people around the area.

**Types of Critical Systems**

- There are four different types of critical systems: mission critical, business critical, safety critical and security critical.
- The key difference between a safety critical system and mission critical system, is that safety critical system is a system that, if it fails, may result in serious environmental damage, injury, or loss of life, while mission critical system may result in failure in goal-directed activity.
- An example of a safety critical system is a chemical manufacturing plant control system.

**Mission Critical**

- A mission critical factor of a system is any factor (component, equipment, personnel, process, procedure, software, etc.) that is essential to business operation or to an organization.
- Failure or disruption of mission critical factors will result in serious impact on business operations or upon an organization, and even can cause social turmoil and catastrophes.
Mission Critical Computing Systems Defined

- A mission critical computer system is a computer, electronic or electromechanical system that is fundamentally necessary to the success of a specific operation.
- When a mission critical system fails or is interrupted, the damage is often quick and substantial.
- Mission essential equipment and mission critical applications are also known as mission critical systems.
- A few examples of common critical computer systems include:
  • Railway/aircraft operating and control systems
  • Electric power grid systems
  • First responder communications systems
  • Online Banking
  • Electric Power systems

Business Critical Systems Defined

Similarly, a business critical system is a system that an organization relies on to carry out the normal business operations that keep the business running successfully.

Essentially, when mission-critical systems are necessary for the success of a business, they become business critical systems.

When a business critical system fails or is interrupted, organizations can face financial losses, customer dissatisfaction and reductions in productivity.

A few examples of common business critical applications include:

- Online banking systems
- Cloud-based data storage and networking systems

Mission Critical

The difference between mission critical and business critical lies in the major adverse impact and the very real possibilities of loss of life, serious injury and/or financial loss.

Mission Critical Systems

- Every business and organization will have mission critical systems.
  - A downed filtration system will cause the water filtration company to malfunction. In this case, the water filtration system is a mission critical system.
  - If a gas system is downed, many restaurants and bakeries will have to shut down until the system functions well again. In this case, the gas system is a mission critical system.
  - There are various other mission critical systems that, if they malfunction, will have serious impacts on other industries or organizations.

Mission Critical Application

A mission-critical application is a program or suite of related programs that must function continuously in order for a business or segment of a business to be successful.

Email is pervasive across all aspects of all businesses and has become highly business critical.

Why has email become mission critical?

- Besides the original purpose of being a carrier of documents and alerts, the email platform has been used as a digital carrier service to connect business applications, trigger business functions, program work flows, support customers etc.

Mission Critical Application

Examples of mission-critical applications vary from industry to industry.

For example, an automatic vehicle locator (AVL) app might be mission-critical for an ambulance company but if a plumbing business uses the same software, it may be characterized as being important, but not essential.
Mission Critical Applications

The architects, developers, testers, and IT operations and support teams supporting a mission-critical application must value stability and availability. Efforts to ensure continuous operations include redundant copies of an application, IT systems and data center infrastructure on which it runs; hot backups; duplicate staging and production environments for thorough testing; and other measures. An organization might choose not to update mission-critical applications as frequently as lower priority applications to reduce the risk of changes causing problems.

Power of 10 rules

- The Power of 10 Rules were created in 2006 by Gerard J. Holzmann of the NASA/JPL Laboratory for Reliable Software. The rules are intended to eliminate certain C coding practices which make code difficult to review or statically analyze. These rules are a complement to the MISRA C guidelines and have been incorporated into the greater set of JPL coding standards.
- Avoid complex flow constructs, such as goto and recursion.
- All loops must have fixed bounds. This prevents runaway code.
- Avoid heap memory allocation.
- Restrict functions to a single printed page.
- Use a minimum of two runtime assertions per function.
- Restrict the scope of data to the smallest possible.
- Check the return value of all non-void functions, or cast to void to indicate the return value is useless.
- Use the preprocessor sparingly.
- Limit pointer use to a single dereference, and do not use function pointers.
- Compile with all possible warnings active; all warnings should then be addressed before release of the software.

Toyota Unintended Acceleration

The first major cause of unintended acceleration was found in March 2007, when an engineering analysis showed that unsecured all-weather mats had led to pedal entrapment and drivers accelerating up to 90 mph with decreased braking power.

One component worthy of investigation was the electronic throttle control Toyota introduced on the Camry in 2002, now virtually universal in the industry.

This device replaces the mechanical link between the accelerator pedal and the engine’s throttle valve with a sensor at the pedal and an electric motor on the throttle.

Was this new device going haywire?

The NASA study of the Toyota Electronic throttle control firmware found at least 243 violations of the power of 10 rules.

Toyota Unintended Acceleration

NHTSA and the NASA Engineering and Safety Center (NESC) published reports of their joint investigation into the causes of unintended acceleration in Toyota vehicles.

NASA’s multi-disciplinary NESC technical team was asked, by Congress, to assist NHTSA by performing a review of Toyota’s electronic throttle control and the associated embedded software.

In carefully worded concluding statement, NASA stated that it “found no electronic flaws in Toyota vehicles capable of producing the large throttle openings required to create dangerous high-speed unintended acceleration incidents.”

The Toyota Electronic Throttle Control (ETC) was far more complex than expected involving hundreds of thousands of lines of software code.
Toyota Unintended Acceleration

NASA apparently identified some lesser firmware bugs themselves, saying "[our] logic model verifications identified a number of potential issues. All of these issues involved aerodynamic timing delays in the multiplexing, asynchronous software control flow." NASA also spent time simulating possible race conditions due to worrisome "recursively nested interrupt masking"; even though that simulation uncovered a sufficient speed of luck of sorts. As well, the NASA team seems to recommend "reducing the amount of global data" and eliminating "dead code."

The standard gcc compiler version 4 generated a number of warnings (larger than 100) about the code, in 11 different warning categories. Coverity version 4.2 generated a number of warnings (larger than 154) about the code, in 10 different warning categories. Codesonar version 3.6p1 generated a number of warnings (larger than 136) about the code, in 10 different warning categories. Uno version 2.12 generated a number of warnings (larger than 72) about the code, in 9 different warning categories.


What is the development process for mission critical software?

* Linus Torvalds argues that most applications do not really care about the IEEE standard and in most applications the speed is more important.
* But what about applications where accuracy and standard compliance is important.
  * Like space programs or software running on commercial aircraft.
  * A very simple but interesting example is where the result is off for certain range of numbers because of the way binary floating point numbers are expressed.
  * Another simple example is square root or division by zero.
  * One way to not get nan from square roots is to add an if before every single square root operation and check if the number is positive, for division check if denominator is not zero.
  * It is hard to enforce it and much harder to detect if there is a case that doesn’t check for validity of division/root.
  * (Just assume adding an if condition before every single division, it’s very ugly).

What is the development process for mission critical software?

* Maybe the majority of these cases is expected to be revealed during tests but there is no guarantee.
  * Compilers usually have some flags for FPU control words to detect such cases, but is it enough?
  * Does it really work in practice?
  * Skimming through the Joint Strike Fighter C++ Coding Standard did not provide that much detail, or guidelines on working with floating point arithmetic and numeric calculations.
  * Is there a C++ library that handles/detects these cases and possible errors introduced by compiler optimization?

What is the development process for mission critical software?

* Independent Verification and Validation (IV&V) is an objective examination of safety and mission critical software processes and products.
  * Independence:
    * Three key parameters:
      * Technical Independence
      * Managerial Independence
      * Financial Independence
What is the development process for mission critical software?

- Systems Engineering:
  - Determines if the right system has been built and that it has been built correctly
  - Will the system’s software…
    - Do what it is supposed to do?
    - Not do what it is not supposed to do?
    - Respond as expected under adverse conditions?

From the NASA Safety and Mission Critical Software guidelines:

- Assurance Strategy is driven by the specific needs of an individual project
- Implemented via an Assurance Design
- Communicated via Assurance Statements

The Assurance Design specifics the Technical Reference, inputs, analysis techniques, and objective evidence necessary to achieve the Project’s Objectives

- Like the Assurance Strategy, the Assurance Design is specific to the needs of an individual project
- Constructed to allow the Project to generate evidence to assure the critical capabilities and mitigate system risk
- Areas of risk identified are key inputs into the development of the Assurance Design
- Assurance Statements are utilized to communicate the results of the implementation of the Assurance Strategy
- A statement of the assurance that is being provided (or intended to be provided) to a stakeholder or stakeholders on a system or subsystem
- Assurance statements are typically formulated at the beginning of a Project and refined as necessary throughout execution
IoT and Mission Critical

- In the consumer-based IoT world, a failed weather sensor or a dropped audio stream is an inconvenience.
- Likewise, a smart coffeemaker that malfunctions might make the consumer unhappy, require an expensive service call, or even result in a costly product recall, but it is not life-threatening.
- In the mission-critical IoT world, the consequences of an unreliable device or device failure can be catastrophic.
  - A failed pacemaker could result in a patient's death.
  - A lost boiler sensor connection might cause the boiler's tubes to overheat and fail, resulting in an explosion.
- Mission-critical IoT is not the same as IoT. They both share common technologies like sensors, cloud platforms, connectivity, and analytics; however, the similarities end there.

Mission Critical and IoT

IoT products once thought of as luxury items for the consumer market will evolve with longer battery life and more robust functionality and performance to become part of the new mission-critical IoT.
Likewise, mission-critical IoT, once defined as critical applications in the healthcare, industrial and power/energy industries, will expand to embrace a broader range of applications in wearables, smart homes and smart cities, among others.
It's happening already.

Mission Critical IoT

The Internet of Things (IoT) represents a vision in which interconnectivity is extended from computers to real-life, everyday objects.
But can a 'mission critical IoT' become a reality?

Mission-Critical IoT is the term used to describe the intelligent network utility companies implement to enable near-real-time communication of their water, electricity and gas networks.
A failure in any of these networks could result in catastrophic consequences for consumers and untold financial losses for the utilities.
The critical aspect these networks represent mandates the importance of choosing the appropriate communications technology supplier.

Mission-critical applications on public cloud

The IT industry is embroiled in a debate about the merits and dangers of hosting mission-critical applications in the public cloud.
The choice depends on many variables, including regulatory compliance, security, performance and availability.
The regulatory variables include government obligations or laws that restrict where applications and data can be hosted and stored.
Public cloud technology has matured in the areas of security, performance and availability since its inception.
As public cloud providers, such as AWS, grow larger and more accomplished, security becomes less of a concern.

In some cases, an organization can save money by relying on security services from a public cloud vendor rather than investing in specialized tools and staff.
Similarly, some organizations prefer to control the entire IT infrastructure for mission-critical applications to ensure the availability of resources for optimal performance, while others turn to cloud providers and specify the capacity and scalability required, leaving the vendor to manage the infrastructure and resources.

Availability is another major factor for organizations thinking of moving their mission-critical applications to the cloud.
Availability depends on the ability of the cloud provider to keep their service up and running.
Some industry experts argue that public cloud providers are better at maintaining infrastructure uptime than individual IT organizations running data centers.
However, if a public cloud provider's services become unavailable, the users' organizations will be unable to resolve the issue.
**Mission-critical applications on public cloud**

In 2013, Amazon Simple Storage Service (S3) experienced slow performance for around four hours.

Websites experienced increased loading times of more than 1,000% of normal activity.

The region in which S3 slowed down, U.S. East-1, was a major hub for customer data and client usage.

Larger companies that hosted business and mission-critical applications using S3 lost around a million dollars during the slowdown.

**Mission Critical and IoT**

**Picture this.**

An Alzheimer patient slips quietly unnoticed out of a healthcare facility.

Or a young child playing in his yard unattended walks off and in an instant goes missing.

It is an all too scary scenario that occurs every day around the world.

In Korea alone, 10,000 Alzheimer’s patients go missing each year.

However, it does not need to be that way.

A startup manufacturer of GPS trackers recently embedded LoRa wireless technology into a low-cost IoT bracelet specifically designed for Alzheimer sufferers.

When one leaves a designated zone, a caregiver is alerted.

Within the first three months of its use, 26 patients were saved.

It is a new type of mission-critical IoT product — and is just the tip of the iceberg.

**Mission Critical and IoT**

**By 2028, virtually everything everywhere will be connected.**

Consumers will expect it.

And, they will expect those connected “things” to be mission-critical, that is, to work as anticipated, without fail, every time.

**Mission Critical and IoT**

But the longer the battery life required, the more time and effort needed to optimize the product’s power consumption.

Signal coverage also needs to be addressed, since if the coverage is too low, the battery will drain much more quickly.

Fully understanding these requirements and how they impact a product’s design is the quickest and easiest way to avoid costly missteps.

It also helps ensure that the IoT product and network perform as expected, regardless of the location or deployment environment.

**Mission Critical and IoT**

- Designing any product is hard.
  - It is even harder when it’s destined for mission-critical IoT. Here’s why:
    - Electromagnetic interference can be problematic in scenarios like hospitals or manufacturing facilities where large numbers of IoT products operate simultaneously near one another.
    - To avoid any problems, interference issues must be dealt with early in the design process where they are easier and cheaper to fix.
    - Often, mission-critical IoT products must perform in the presence of multiple users, with different wireless technologies, in the same spectrum.
    - Verifying that a product can handle the load is critical to ensuring robust wireless connectivity.
Mission Critical and IoT

- With lots of mission-critical IoT products entering the market, peaceful coexistence between them can be complicated.
- It’s especially problematic in hospitals where IoT monitoring devices share the 2.4 GHz ISM band with the likes of cordless phones, wireless video cameras and microwave ovens.
- Making sure products can work as anticipated in this type of environment is crucial.
- Mission-critical IoT products support a broad range of wireless technologies.
- Networks must support them as well, and in a range of different environments and locations with differing RF conditions.
- To prevent network disruptions in quality or performance, any issues impeding network readiness must be identified and eliminated.

Mission Critical and IoT

- Roughly 50% of all IoT products come from companies less than three years old.
- Some products have been thoroughly tested, but not all.
- Those products may behave erratically and even allow malicious agents to bring a network down.
- Ensuring the network can handle erratic products and the security concerns they enable needs to be a top priority.
- Continuous updates/upgrades to network equipment keep networks in flux.
- Whether existing network devices can even support new services is a question often left to chance, and that makes verifying that deployed networks are reliable and offer a high quality of experience essential.

Real time and mission critical

Real time and Mission critical are often confused by many people but they are not the same concept.

Real time is responsiveness of a computer that makes the computer to continually update on external processes, and should process the procedure or information in a specified time, or could result in serious consequences.

The 3D computer games or movies are examples of real time since they are rendered by computer so rapidly, it is hard for the user to notice delays.

The speed of rendering graphics may vary according to the computer systems.

Differences

Real time is a software that if a specific time is not met, it fails, but mission critical is a system if failed, will result in catastrophic consequences.

Although they go hand-in-hand, since real time can be mission critical, they are not the same concepts.

These two are often confused by many people, but they are different concepts, but associated with each other.

Real time and mission critical

Types of real time systems

Hard real-time system shouldn't miss the specified time or can result in serious consequences.

- It is non-negotiable in timing and it is a “wrong answer” if the deadline is missed.
- An example of hard real-time system is airbags for cars.

Soft real-time system has more loose deadline.

- The system can handle the problems and functions normally even though the deadline is missed, but their functionality depends on fast-paced processors.
- An example for soft real-time can be typing, which, if delayed, people will get annoyed, but the answer still is correct.

A Non real-time system doesn’t have certain or absolute deadlines.

- However, the throughput of the activity of performance can still be very essential.

Safety Critical Software

Safety-critical systems go through a rigorous development, testing, and verification process before getting certified for use.

For avionics software and other airborne systems, the de-facto standard for software development is RTCA/DO-178C Software Considerations in Airborne Systems and Equipment Certification (also published in Europe as EUROCAE ED-12C).

Achieving certification for safety-critical airborne software is costly and time consuming.

Once certification is achieved, the deployed software cannot be modified without recertification.
Safety Critical Software
Unauthorized modification of the certified software presents a significant risk to safe operation, and today’s safety-critical systems face a variety of threats from unintentional and malicious actors.
If the software is changed maliciously or even unintentionally from the certified configuration, it is no longer safe.
Bottom-line: a system that is not secure puts safety at risk.

Safety Critical Software
A real-time operating system (RTOS) has a central role in safety and security. While there are several operating systems that have completed safety certification as part of an overall safety-critical system, there are very few designed for security-critical applications and fewer still that have achieved any official security certification.
It is not sufficient to have separate RTOS products with one being safe and another being secure, because the primary RTOS or hypervisor needs to be safe, as well as secure.
The only way to meet this requirement is to build safety and security into the same RTOS.

Safety Critical Software
- When several applications exist within the same system, those applications often handle tasks of varying importance — often called different criticality levels.
- Integrated modular avionics (IMA), for example, combines many avionics processing functions onto a set of shared processing resources.
- For flight safety, those different criticality levels are called design assurance levels (DAL), ranging from DAL A, where a failure would be catastrophic, to DAL E, where a failure would have no safety impact.
- The safety-critical RTOS is responsible for providing the memory, time, and resource partitioning to protect each application from being affected by all other applications; it must protect higher criticality applications from negative or unintended affects of lower criticality applications.

Safety Critical Software
- Similarly, different applications running on the same system can be at different security levels.
- To ensure that those applications access only the information for which they are authorized, the system needs to implement a multiple independent levels of security (MILS) operating environment.
- A MILS operating system isolates applications and their data into different security domains, and provides mechanisms for permitting authorized communication across different domains.
- A MILS operating system should support hosting multi-level secure (MLS) applications, which typically are cross-domain solutions (CDS) that filter specific information flow from higher security levels to lower security levels.

Safety
- Mission critical systems (in general) should remain very secured in all industries or organization using them.
- Therefore, the industries are using various security systems to avoid mission critical failures.
- Mainframes or workstations based companies are all dependent on database and process control, so database and process control would be mission critical for them.
- Hospital patient recording, call centers, stock exchanges, data storage centers, flight control tower, and many other industries that are dependent on communication system and computers should be protected from the shutdown of the system and they are considered mission critical.
- All companies and industries are susceptible to the unexpected or extraordinary problems that can cause shutdown to the mission critical systems.
- To avoid this, using the safety systems is considered very important part in the business.

The Coming Software Apocalypse
It’s been said that software is “eating the world.” More and more, critical systems that were once controlled mechanically, or by people, are coming to depend on code…
Software is different. Just by editing the text in a file somewhere, the same hunk of silicon can become an autopilot or an inventory-control system.
This flexibility is software’s miracle, and its curse.
Because it can be changed cheaply, software is constantly changed; and because it’s unmoored from anything physical—a program that is a thousand times more complex than another takes up the same actual space—it tends to grow without bound.