5 Ways to Overcome Embedded Software Development Challenges In 2022

With its presence in industries across verticals, **Embedded Software Development** is encapsulating different machines and devices to facilitate their control using computer software. Embedded software is programmed into those devices that are built around microprocessors and implements higher-level features of the machine or device.

To optimize these features and functions, **embedded software development** drives the innovative designing, developing and implementing the programmed software. Moreover, with technological expansion and the growth of the Internet of Things (IoT) devices, **embedded software development** has picked up pace to bring unmatched innovations in the electronics industry.

So, for a device such as a smartphone or a digital watch, a consumer expects real-time response while operating under the processing power, limited memory and energy supply. For a more complex embedded software technology, such as software for connected cars, the embedded software code should meet the safety requirements too. While integrating with IoT devices, the coding for embedded software should work with the complexities of voice/face recognition.

Development organizations are still battling with some challenges in the embedded software industry. To help you with the challenges, we walk you through a 5-way guide to overcome those challenges:

- **Connectivity**

Developers must comprehend the various software stacks to enable the hardware, just as they use the different and best fit options from Wi-Fi, Ethernet, Cellular, LoRa and Bluetooth. By the same token, there are an array of options for communication technology to choose from the local devices of servers and sensors.

These communication technologies include NFC (Near Field Communication), RFID (Radio Frequency Identification), WiFi and Bluetooth. To facilitate sharing of information, it becomes vital to support server-to-server communications (S2S), device-to-device (D2D) and D2S (device-to-server). Moreover, an **embedded software development company** must also consider other software communication layer protocols such as:

- IPv6 (Internet Protocol version 6) packet switched link layer networks, providing an infinite number of addresses.
- LPWAN (Low Power Wide Area Network) sends small data at a low rate over a long-range, suitable for high bandwidth that is not time sensitive.
- UDP (User Datagram Protocol) provides low-latency and loss-tolerating connections between applications.
- COAP (Constrained Application Protocol) web transfer is often used to enable simple, constrained devices to join the IoT even through controlled networks with low bandwidth and low availability.
- TCP/IP (Transmission Control Protocol/ Internet Protocol) sends packets across the internet and ensures the successful delivery of data and messages over networks.

- **Over-the-air (OTA) updates**

After successfully connecting a device to the internet, its firmware needs to be updated. However, there is likely a high probability that the firmware would be in a remote location and its update would also need to be done remotely.

To perform the task, there are some fundamental requirements such as:

- Regulate when there is a suitable update
- Create a firmware update
- Protect updates to ensure it cannot be interfered
- Maintain configuration after updates
- Confirm bandwidth for update
- Encrypt updates to disrupt vulnerabilities from entering

- **Security**

A golden rule to always remember **embedded software development services** is that every connected device has the potential visibility to everything else on the internet. Therefore, security risks are easy to be exposed and this demands a system that can be isolated and enables the encryption of data at both in transit time and at rest.

Mitigating risk must be well thought out and should be on priority throughout the whole IoT lifecycle to achieve organic scalability.

- **Debugging**

With the exponential increase in the size of development teams, there is an increase in the complexity of connected devices, which leads to more debugging efforts. Conventional methods such as integrating open-source software is likely to introduce an unanticipated behaviour within the system, unless there is a calculated task done together. To contain this issue, it is vital that various debugging techniques are educated to the development team so they get a fair understanding of the type of bugs and impede the risk by analyzing it.

- **Testing**

A robust embedded software system testing has similarities to the software application testing framework. While software application testing draws its vitality on graphical user interface (GUI) and black box, without stressing on source code, the Embedded software testing considers simulation, hardware, real-time behaviour and down time. This is because the ultimate goal of an **embedded software development** testing depends on a myriad of aspects.

This however is not an easy task to perform, but usually lacks the bandwidth and expertise to perform testing. Therefore, to keep the testing as required, the development team can incorporate aspects like GPS or cellular to make tests unique considering the satellites are continuously moving under the influence of environmental factors.

To get the best embedded software services, you should opt for the reliable Product Engineering and Digital Transformation industry. For that, you can make Sasken a go-to choice. Sasken has over two decades of deep domain expertise in providing the services of embedded systems, protocol and middleware services to global customers. It offers **embedded software development** solutions such as end to end testing across product life cycle, App FW, OS, BSP and driver validation, MM algorithms and middleware testing Protocol stack, FT, IOT Standard conformance and certification testing.