



# BEWARE EXTREME SOFTWARE



Many have the perception that taking the middle ground is to accept a compromise, which is a positioning that seeks to make most people happy—but in reality leaves everyone wanting.

The degree of disappointment scales with how far towards the lowest common denominator any set of fulfilled requirements descends. Is this principle true when it comes to manufacturing software? **Should we seek extremes in order to avoid compromise, or is there a balanced approach that could completely deliver on our digital transformation?**

Negotiation around a situation can be a complex matter. As human individuals or groups, we each have different experiences, perspectives, expectations, and priorities. The trick in getting uncompromised consensus is first to understand each parties' true needs, wants, and likes, and to then understand the root causes and potential consequences of each, like thinking it through. Extreme viewpoints tend to dominate conversations, as they have at their core a very strong and compelling value from a certain perspective, which overwhelms other factors and considerations. Putting every element into perspective is extremely important, though not exciting.

Elements within software development and solution packaging have attracted numerous different, and sometimes extreme, viewpoints, each of which influence software solutions, their core capabilities, and ultimate sustainability. The purchaser of software solutions needs to be skilled at navigating around negotiations with vendors to reveal not only the highlighted values, but also the true costs of ownership and potential limitation of further opportunity.

**Two fundamental areas of crucial importance that are almost always overlooked by extreme solutions are that of a defined ontology and security.** Ontology within software represents mechanisms behind the data structure and information rule modelling, which is the essence of creating value. Without this, any extreme solution can capture data and display it in exciting ways, but create very little actual value.

Digital transformation is not simply about automated data collection and reporting. From the security perspective, many extreme solutions gather any and all data together, creating "big data" repositories, without any thought for the security of intellectual property content or privacy of key information. The result is a "free for all" approach with the hope that no one will access data without authorization, steal, alter or tamper with data, or even introduce nefarious cybersecurity-related issues, all while achieving very little value.

Different solutions are based on their individual philosophies which, at the extremes, are very significantly different. Yet, factors that contribute to each in terms of value, cost and risk are remarkably similar. Each extreme view represents a domination of one philosophy over another, creating an unbalanced solution model, which unknown to potential users, can easily tip return on investment into significant loss of opportunity. **Looking into these extreme approaches reveals a great deal about the long-term viability and sustainability of any software solution.** This whitepaper considers some examples at the commonly experienced extremes of such philosophies in terms of MES software.

## Table of Contents

- 2 Introduction
- 3 Extreme: In-House Development
- 5 Extreme: Open-Source
- 6 Extreme: Commercial Solution Dominance
- 7 Extreme: App-based Technology
- 10 Extreme: Interoperability
- 12 Extreme: Flexibility
- 14 Conclusion

## Extreme: In-House Development

**Software developed in-house—ranging from complex solutions to customized middleware—originated with the necessity for the “do it yourself” approach, which persists today.** Such solutions are tightly scoped and focused on current needs, developed by an internal development team and / or outsourced partner.

There is a clear and unambiguous specification, which once agreed, needs little flexibility. The software itself can therefore be made smaller and simpler, architected and data-modelled in the way that best fits the immediate project. The developer of the software feels as though they are in full control, and users feel comfortable that the application works in the best way for their specific roles. Benefits are relatively easy to measure, and the solution quickly becomes incumbent. This is a very common “origin story” for thousands of “in-house developed” or “home-grown” point solutions that prevail within manufacturing today; each of which can be considered an element of MES functionality. Small manufacturers may feel that this approach is a low-risk starting point for their digital transformation. Larger companies feel that having such technology firmly under their own control is favorable.

---

Small manufacturers may feel that this approach is a low-risk starting point for their digital transformation. Larger companies feel that having such technology firmly under their own control is favorable.

---

With headline values strongly highlighted, the many associated challenges, however, are often overlooked. **With extreme self-development, the specification of the solution rests with a very limited number of internal solution architects.** There will be a subject matter expert or two to help drive this, though the result is a very narrow and simplistically focused data model and ontology. It is designed to get currently defined values out of the data, with no regard for other potential values either now or later.

If developed by an internal team, active members will vary over time as their roles change within the company or they seek other opportunities. Retained know-how relating to the data model, algorithms, and methods within the software is usually lost, leaving others with difficulties related to how to continue development and maintenance of the software, which often then requires reverse engineering. Incorrect assumptions are often made and there is a lack of professional-level documentation.



### Key Challenges with Extreme In-House Development



Narrow Data Model



Simplistic Solution Architecture



Loss of Tribal Knowledge Over Time



Difficulties with Continued Development



High Cost of Modification

Where such solutions are developed as customization by outsourced partners, the same issues exist. Teams are associated with each customer's project, which are not likely to be the same should changes or additions be needed. Specific know-how is retained only in the documentation and specification. This leads to far greater than expected costs should any changes or maintenance be required at any point.

**The simplistic solution architecture becomes a major barrier to expansion or further innovation.** The need for connectivity between such point solutions and other automations, including the need for cybersecurity, are often completely missing. Solutions based on extreme self-development therefore quickly become outdated, constraining manufacturing operations to legacy practices, preventing true digital transformation as compared with peers in the industry. Increasing dependencies are made on key people who become more difficult to retain, until ultimately, such solutions become "untouchable." No one remaining in the team has the confidence to even maintain the system in line with operating system upgrades and the application of security measures, never mind making added-value changes.

It is common to see internal "champions" for such solutions, keen on maintaining a high degree of momentum and saving face, repeating the same projects over and over, with costs that increase faster than the inclusion of new benefits. A radical change within an organization, at all levels, is required to break this cycle and dependency on legacy practices and move forward.



## Extreme: Open Source

Another extreme approach in software development, whether part of an in-house self-developed solution, or for filling in gaps of functionality and connectivity between existing point-solutions, is the use of open-source software components. The idea of sharing the burden of development costs across a community of developers can be very attractive, with the feeling of getting, “something for nothing.”

Open-source software is usually available without cost and often without restrictive licensing. Software libraries, development kits (SDKs), and modules of code can be sourced and integrated with in-house solutions to reduce the burden on development and customization teams. Trusted open-source software, for example, where provided by a trusted industry-consensus based organization for a specific purpose can be of significant benefit. In the vast majority of cases, however, there are some significant security and integration challenges.

Though there are a significant number of open-source software projects available, whether a significant part of an MES solution could be put together is very doubtful. Very few open-source participants share truly valuable Intellectual Property (IP). **Open-source components tend to be enablers rather than solutions.** Care must be taken with trust in the origin of such open-source code, as many developers from many backgrounds, with potentially conflicting motives, are likely to have contributed. This can affect the quality of the code, which may have acquired several layers of complication, as it has been manipulated to perform differently across different application use cases.

**This results in open-source software being much larger and more complex than would otherwise have been needed.** It makes the job of verification of such software (e.g. to eliminate potential malware or other unwanted effects) very much more difficult, with line-by-line forensics required if open-source code is to be used in a business-critical environment.

**The potential downsides of security and understanding the full code intent, as well as the relative lack of specific MES-orientated functionality, make the inclusion of open-source software from non-trusted organizations impractical and dangerous.** The connective code needed to connect disparate open-source modules and to support robust business logic and data modelling exposes increasing problems and compromise, driven directly by inherent limitations.

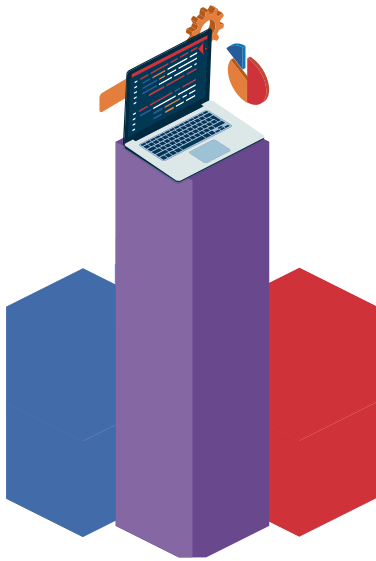


### Key Challenges with Extreme Open-Source

-  Security Challenges
-  Integration Challenges
-  Poor Quality Code
-  Bloating and Excess Complexity



## Extreme: Commercial Solution Dominance



**Market domination is a business doctrine based simply on revenue maximization, designed to eliminate competition and interoperability.** The domination strategy is seen in many solution providers in the market, from the largest and most expensive, all the way down to the simplest app-based environments.

The strongest hatchling in a nest will use their wings to push aside their sibling competitors as they aggressively fight for food. Companies bring this principle to dominate customers with their solutions using the same logic in order to win all of the available potential business. When such an MES supplier is asked by a customer to provide information about the functions and features they provide, for example, against a check list of five key needed features, the dominating companies will always try to tick every box, irrespective of whether their existing functionality meets the specified needs. This appears then to align with the goal of those responsible for the procurement of such solutions and is seen as a compelling value.

Often, the reality is that the supplier can satisfy perhaps three of the five requirements well, with a fourth partially supported, having perhaps been recently developed for a prior customer. The fifth requirement may not be supported at all, but the supplier will say that they will commit to develop a “solution” for the customer. The proposal is often spun so as to say that since it will be developed specifically for that customer, it will satisfy them to a greater extent than any off-the-shelf solution from one of their competitors. They effectively position their weakness as a strength. **The result, however, is a customized, bespoke solution, very much looking like an extreme in-house developed solution in terms of structure, values, and risk.** The delay to deployment and additional costs involved are often overlooked, as this supplier becomes the only one that “has ticked all the boxes.”

In all cases where the practice of solution dominance are seen, there is the fundamental restriction that the customer can only follow practices and operational flows that are dictated by the solution, without flexibility to connect solutions from other parties or providers. Anything that falls outside of existing options in either case, can only be satisfied with the relatively significant cost of customization. In doing so bespoke point solutions are created that exist outside of regular support and maintenance contracts. Whenever any update of the core system is made, such bespoke additions need to be reviewed and potentially redeveloped.

**Another aspect of having a dominating solution is that the whole digital transformation process is then constrained by whatever the single vendor has to offer.** For simple app-based platforms, these will be solutions with the lowest common denominator of functionality. For large solutions, use

of additional functionality is likely to have poor ROI, as such solutions may be far more complex than needed. Industry best practices and digital transformation technologies continue to evolve rapidly, often leaving users of these dominant solutions in limbo, progressing at a pace and in a way that may not be appropriate for their business.



### Key Challenges with Extreme Solution Dominance

-  Bespoke Solution
-  Rigid Operational Workflows
-  High Customization Costs
-  Restricted Based on Vendor Innovation
-  High Level of Complexity



## Extreme: App-based Technology

Since the 1990s, it has frequently been said that software development engineers would soon become obsolete; software solutions would be created without the need to develop code. Over and over, this has proven not to be the case. History has shown, in all but a few very specific cases, that this is nonsense.

---

**There are many gaps where specific customer requirements are left unfulfilled, related to business logic, incorporation of the latest AI-based algorithms, machine and device interfaces, and more.**

---

The idea behind more recent app-based MES solutions assumes that software libraries can contain all of the necessary code that supports user-based creation and configuration of any solution as a series of simple-looking apps. The apps can be changed at any time without the need to change the core code within the libraries.

There are compelling headlines supporting this “no-code,” app-based idea. Many in manufacturing have experienced, or have heard cases where dominant MES solutions have been adopted that promise the world, and deliver a street. Years of effort and customization by the vendor, customer, and third-party middleware providers have then taken their toll, with high costs, many deployment issues, and lack of coherent support of the self-developed customizations. **It is quite natural to be seduced by the promise of a software environment that can create an app a day to satisfy requirements.** App-based platform solutions often include open-source communities that share ideas and coding, to help customers configure their apps.


App-based solutions utilize a series of libraries that contain high-level MES application / solution-specific code, which rapidly ages and becomes out of date, especially as such solutions evolve from primitive origins. Ironically then, today’s app-based MES solutions exhibit the same risks and challenges that are in common with extreme in-house software development, extreme open-source, and those associated with commercial solutions that seek to dominate their market. **Any changes within the app-based application libraries risk affecting and invalidating the entire layer of existing apps that have been created and configured by the customers, as well as all of the methods, code, and functions that have been provided by the support community.** The responsibility of keeping solutions working then falls to the customer, often without warning, as apps start to behave in unpredictable ways. In addition, there are many gaps where specific customer requirements are left unfulfilled. These may be related to business logic, incorporation of the latest Artificial Intelligence (AI)-based algorithms, machine and device interfaces, etc.

Therefore, there is inevitably the requirement for customers to retain the ability for extreme self-development of code, albeit just beyond the exact scope of the app-based solution itself. This includes the need for connectivity with other solutions and sources of data, filling gaps of specifically needed functionality, as well as the triage of support when core solution libraries are updated. Code is therefore needed, whether developed in-house, by an outsourced customization partner, a community, or by the adoption of third-party middleware. Such band-aids rarely include the maturity to cover future needs or use case flexibility. **Rather than being a no-code solution, app-based platforms bring us full circle to the same development and support issues faced by early software developers, albeit in a far more attractive way.** The need for code is simply positioned outside of what is provided, and responsibility taken, by the supplier.






## Extreme: App-based Technology

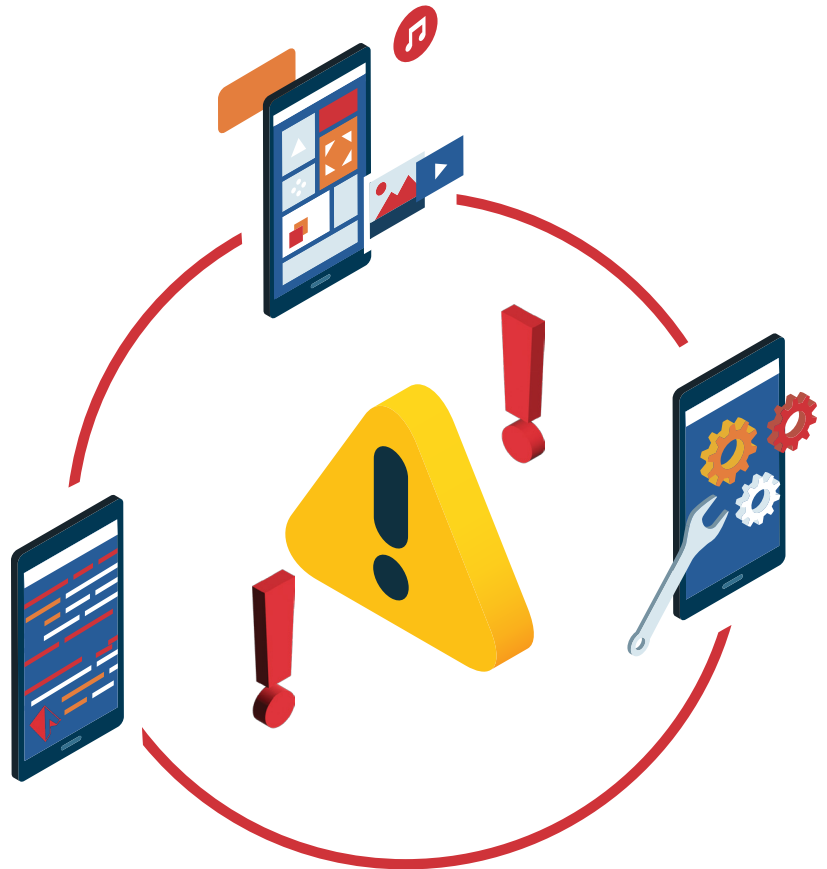
The argument is then made to effectively freeze the core library functionality as far as possible with only incremental changes that are backward compatible. Every manufacturing company, however, has a business mission to evolve and expand, attracting new customers and business opportunities. Increasing the number of products produced, the number of variants, the types of products in different sectors (each with a differing nature of operational and exception flow models), as well as differing types of materials, automation, flexibility needs, frequency of engineering changes, batch sizes, quality management policies, conformance, and traceability requirements are all just a few of the many inevitable sources of variation seen across factories. **The inability—or extreme effort—needed to change and adapt app-based platforms in line with these changing needs therefore becomes a significant business limitation.** Even the most subtle differing requirements can make or break the implementation of a simple, single task-orientated solution. Simple, app-based modular solutions are not able to cope with such challenges, and are therefore orientated successfully only towards short-term, simple production scenarios.

Digging deeper, the situation becomes worse. App-based MES solutions typically require the configuration of data elements and relationships by the customer, with very little, if any, control of naming, formatting, specification of relationships and dependencies, as well as risking duplications, overlaps, and inconsistencies across apps. The responsibility for the ontology and data models falls ultimately to the customer, who are usually not themselves data scientists nor solution architects. Customer-created data models are not likely to be robust, either to satisfy immediate requirements nor flexibility to support future opportunities, in the same way as self-development of solutions has shown.



### Key Challenges with Extreme Solution Dominance

-  Out-of-Date App-Specific Code
-  Difficulty with Customization
-  Unpredictable App Behavior
-  Need for Code Band-Aids
-  Poorly Defined Data Models
-  Security Vulnerabilities





The lack of a robust data model also impacts another area—connection of app-based solutions—side by side with other MES solutions, often mentioned as a strong-point related to app-based MES implementations. Successful integration, however, requires that the data models of connected or associated solutions be matched, enabling the free flow of information exactly as required by the business logic of each application. This introduces very complex integration work, involving significant code development.

**With such a mish-mash of code being put together, including several configured apps, open-source functions, in-house or outsourced customization and middleware, the result is a minefield of security issues and vulnerabilities.** IT teams have a major role to play in manufacturing, securing what has been a “wild west” of digitalization of the operational (OT) network. Inevitably, as production consumes and creates data, traffic between the OT and IT networks is a necessity. Security vulnerabilities that can so easily disable production operations, leak IP, and compromise product quality, need to be extremely well governed and managed in order to avoid successful cybersecurity attacks. The app-based platform approach is headed in the opposite direction with some even promoting the fact that the solution is further away from IT security, maintenance, and control, which to many represents a very significant issue.

The unfortunate result is that while the simpler app-based environment may be initially compelling, the longer-term cost and risks of ownership is unsustainable, and something that such solutions providers neglect to fully disclose.

---

The app-based platform approach is headed in the opposite direction, with some even promoting the fact that **the solution is further away from IT security, maintenance, and control, which to many represents a very significant issue.**

---



## Extreme: Interoperability

With other extremes of software development strategy seen so far, no matter what approach is considered, the same issues come up again and again. **Is there ever going to be a way to break through issues that seem inherent for the MES environment?** Representing the opposite of extreme domination, extreme interoperability represents a framework in which the continuous selection of the “best tool for the job” can be done, either from within a single overall flexible solution or with the freedom of selection of solutions from different market players, each supporting their own niche of functionality.

**The compelling value of this approach is the ability to mix and match solutions that exactly meet requirements, working together without conflict, and are easily independently upgradeable at any time.** Such interoperability is a relatively new concept that is gaining momentum rapidly. From the digital transformation perspective, one of the main challenges to interoperability (that is the exchange of shop-floor manufacturing data) has been resolved, for example, by the use of the IIoT message-based IPC Connected Factory Exchange (CFX) standard—where a single data model is used for all machines and operations, allowing complete plug and play interoperability and avoiding vendor-related dependencies.

---

A certain amount of interoperability is required to integrate current and future AI-based applications into existing MES frameworks. **AI applications do the “thinking” while MES manages the actions.**

---

**The lesson learned from the CFX example is that interoperability needs to be based on open standards, otherwise connectivity is limited by proprietary interfaces and / or data model representation, which once again attracts the need for customization, data model translation, and the use of middleware.**

Rapid development of AI applications relating to manufacturing operations is ongoing, which includes those related to Machine Learning (ML), closed-loop feedback solutions, and live analysis of patterns and trends in data relating to topics such as predictive maintenance, energy saving, zero defects, bottleneck resolution, etc.

A certain amount of interoperability is required to integrate current and future AI-based applications into existing MES frameworks. AI applications do the “thinking” while MES manages the actions. Interoperability is essential in these areas, which is yet to evolve. Proprietary “ecosystems” carry a very high risk, where not based on open standards. While good as a support for proofs of concept, such ecosystems bring aspects of extreme domination, building dependencies on multiple points of integration with each solution.

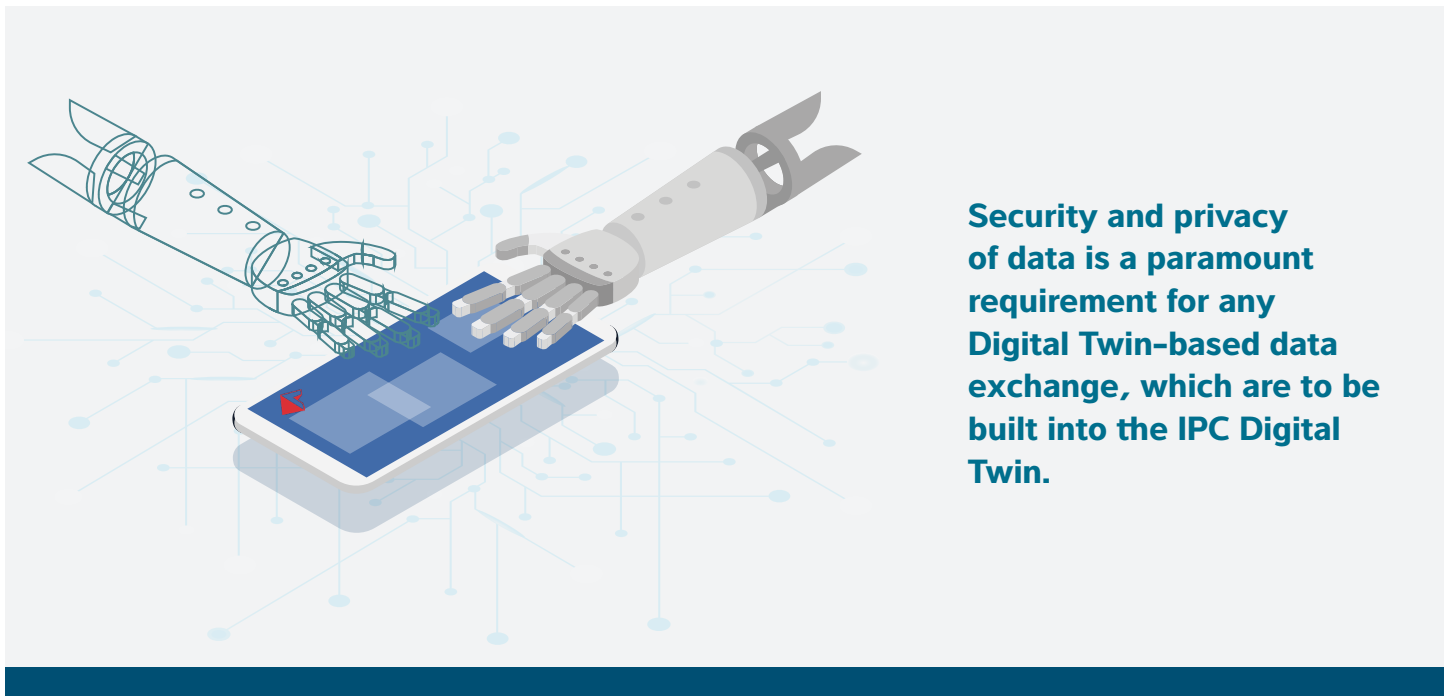


**A certain amount of interoperability is required to integrate current and future AI-based applications into existing MES frameworks.** AI applications do the “thinking” while MES manages the actions. Interoperability is essential in these areas, which is yet to evolve. Proprietary “ecosystems” carry a very high risk, where not based on open standards.

In the example discussed earlier as part of the extreme domination approach, three out of the five customer's MES needs were very well supported by one vendor. With extreme interoperability, it would be possible to easily integrate off-the-shelf solutions from other niche vendors so that all five original requirements can be exactly met and run successfully together, avoiding the need for in-house development, commercial customization, or the use of middleware. Each niche solution can be selected to meet the exact needs in terms of required level of functionality—for example, where a very highly sophisticated MES-based quality solution is needed, but just a simple visual solution for planning and scheduling will suffice. **Each element of an overall MES solution can be specified according to the exact need, reducing the costs and complexities of digital transformation, thus avoiding the redundancy of over-complex or overlapped solutions.**

Another existing open standard example that intends to support extreme interoperability is the IPC Digital Twin, which describes and indexes digital twin content throughout design, manufacturing, and operation of products in the market. The IPC Digital Twin standard promotes interoperability of applications in the same way as CFX promotes interoperability for data messaging, where possible, the existing popular formats of data. Security and privacy of data is a paramount requirement for any Digital Twin-based data exchange, which are to be built into the IPC Digital Twin.

Many organizations are currently creating their own methods for how to represent data about “things” in a digital twin format, breaking down significant assemblies and machines into their base component elements, about which, data is created that describe their form and function. The results of these projects will provide a choice of formats, each of which can be declared through the creation of a cell within the IPC Digital Twin structure, much in the same way as a modern media file contains metadata related to the format, encoding method, bit rate, etc., of video data.



**Security and privacy of data is a paramount requirement for any Digital Twin-based data exchange, which are to be built into the IPC Digital Twin.**

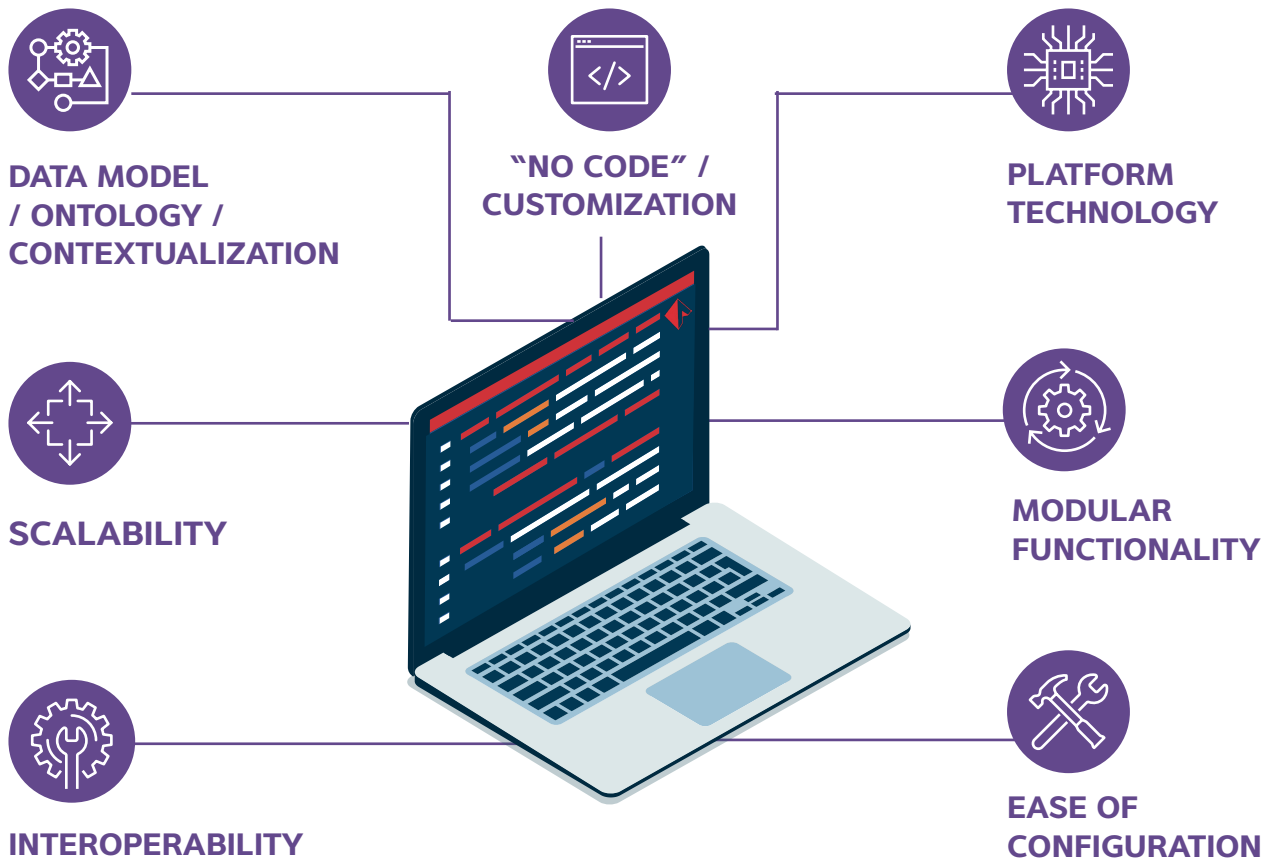
## Extreme: Flexibility

As we have seen, for each extreme approach to software development and solutions, there are compelling reasons for adoption, which disguise very significant, long-term business-limiting downsides, and usually far outweigh the benefits. The reality of any MES solution is that there is complexity. **Even those created for single site implementations need the flexibility to evolve over time as manufacturing needs change.** From the commercial solution perspective, we see that no two manufacturing operations—within the same company—are the same.

The simplification of MES to be a low-level solution based on common elements, as seen in the apps-platform case, is not realistic because the data model, ontology, connections, and integrations remain a unique requirement for each manufacturing operation. They are not satisfied by the solution itself, meaning that investment in code development, middleware, and significant solution engineering are left to the customer to discover post-purchase. None of the code pain has really been avoided.

If all of this is true, how can you best minimize the cost and risk of MES solutions? **Flexible MES solutions contain many configuration options, which tailor the ways in which it works, representing choices related to common operational functions and also the satisfaction of advanced requirements.** These configuration options may appear overwhelming at first glance without guidance, but represent the sustainability of the solution in that the MES can adapt to significant production changes and best practice adoption. Such options help drive digital transformation forwards by providing the ability to adopt superior practices over time.

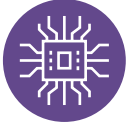
### Specific considerations to consider for MES flexibility are:





## DATA MODEL / ONTOLOGY / CONTEXTUALIZATION

The main values of an MES solution are not simply representable as a list of checked boxes. **The cleverness and intelligence within an MES solution is enabled by the understanding of the interaction of the thousands of data points that are continuously recorded,** the rules and algorithms that create information relevant for the visibility, management, and optimization of the production operation in real time. The evidence of such technology comes as software is seen to create solutions to production challenges before they become significant, as opposed to simply automating existing operations without any additional support. The data model is the most fundamental starting point for MES solution intelligence.



## PLATFORM TECHNOLOGY

The underlying platform architecture should **be built upon a single technology, preferably IIoT-based,** avoiding the case where older solutions have been combined in a way that simulates integration, but in reality, is synchronization, which causes inconsistencies in operation due to underlying differences between the solutions.



## MODULAR FUNCTIONALITY

There are many different opinions throughout the industry as to what is the true scope of an MES solution. Adopters of MES should not need to care about these arguments, and instead **model the solution on their current needs and future expectations.** The ability of an MES solution to provide the modularity that enables managed, phased adoption according to customer need is essential.



## SCALABILITY

MES solutions should be **scalable in terms of depth of functionality, as well as by the number of users.** Manufacturing requirements are strongly determined by the nature and classification of the product. It is quite normal that a very high specification may be, for example, required for quality management, whereas for scheduling, visibility, and routing enforcement may require only the essential tools.



## EASE OF CONFIGURATION

As important as the day-to-day solution operation, **configuration options should be clear and intuitive.** The vast majority of software solutions, even such packages as Microsoft Excel, are not utilized to their full potential simply due to a lack of awareness and understanding of features and functions that are available.



## INTEROPERABILITY

As standards that promote interoperability become adopted (such as the IIoT messaging standard IPC CFX and interoperability of data between solutions), using the IPC Digital Twin standard, the need for bespoke interface development is reduced and eventually will be eliminated. Solution providers should be sought that **provide thought leadership for such standards, as well as native support where appropriate.**



## "NO CODE" / CUSTOMIZATION

In a perfect world, there should not be the need for any code development for the adoption of an MES solution. As interoperability throughout the industry continues to evolve, customization is still required, including for apps-based platform solutions, albeit outside of the core solution. Application-bespoke code, whether inside or outside of the core solution should be avoided wherever possible, as it presents a high risk of failure and a high cost of ownership, as such code is often not properly maintained, or considered as solutions evolve. **Any code that is required as an addition should be integrated into the core solution itself, so as to be maintained as part of the holistic solution.** Such a policy also eliminates any duplication of code development and increases the value of the solution for all users. With this policy, we see evolution of MES solutions as part of the complete digital transformation process, as opposed to the stagnation of functionality where customization remains a unique "bolt-on" for specific customer use cases.



## Conclusion

**The consideration of extreme approaches to MES software shows that there are some important decisions to be made when selecting solutions, going far beyond the marketing messages that highlight extreme approaches.** A simple lowest common denominator-based solution approach to MES will never be realistic. Any claimed “no code” approach has shifted inevitable coding needs outside of the core solution, if anything, making them more difficult to manage, and doing so at the expense of not having a clear or robust built-in data model.

Customers should not be tied into a limited proprietary solution by a dominant vendor, paying dearly for missing functionality that will forever require expensive custom support. **Until true standards-based interoperability becomes mainstream, full interoperability is out of reach, a goal towards which the true open standards-based ecosystem of MES and AI applications aims to achieve.**

MES solutions should work simply “out of the box” based on an easy selection of high-level options, values based on a mature data model, and supported by built-in ontology based on hundreds of years of manufacturing experience, with the flexibility to evolve to meet the myriad of individual customer needs, while also expanding and evolving the core value proposition in line with digital transformation expectations.

[Learn More](#)



**Email:** [info@aiscorp.com](mailto:info@aiscorp.com)

**Visit:** [www.aiscorp.com](http://www.aiscorp.com)

**@FactoryLogix**

**linkedin.com/company/aegis-industrial-software**

### Corporate Headquarters

220 Gibraltar Road, Suite 300  
Horsham, PA 19044

**Phone:** +1.215.773.3571

### European Headquarters

Wetterkreuz 27  
91058 Erlangen, Germany

**Phone:** +49.9131.7778.10

### Asia Headquarters

Rm. 809, Dahua Hucheng Business Center  
No 6, Lane 239, Dahua No. 1 Road

Putuo District, Shanghai, 200442, P.R. China  
**Phone:** +86 2 1 5882 4882