1

Why Interoperability R&D Work Should be Driven by Agile Integration and Message Standards Concerns?[1]

Smart Manufacturing assumes agile integration of manufacturing services and applications [IVE 17, KUL 16]. This implies that manufacturing message standards, which are key to the services integration, need to be quickly created, used, and managed. That is a far cry from the best practices today. The interoperability R&D community should take on the challenge of enabling new message standards life-cycle management (LCM) capabilities needed for rapid integration and reconfiguration of manufacturing systems.

1.1. Vision: Rapid Integration and Reconfiguration of Manufacturing Systems

Manufacturing is on the cusp of being massively transformed: static and monolithic manufacturing systems are changed into dynamic networks of distributed manufacturing services. This transformation is even more radical now with the application of the Industrial Internet of Things (IIoT) technologies in manufacturing.

[1]Chapter written by Nenad IVEZIC and Boonserm KULVATUNYOU

In the new state, the required functionalities will be provided by services, which could be much more flexible and cost-effective to use. Manufacturing, however, requires sure-proof transformation approaches for the new service-based networks, including efficient integration, configuration, and re-configuration of services and IIoT solutions. Despite the complexity, visions of agile, re-configurable, service-oriented manufacturing systems continue to drive industry investments.

1.1.1. *Example: Smart Supply Chain Scenario*

Manufacturing Operations Management (MOM) or Enterprise Resource Planning (ERP) services of a manufacturing company will communicate with Third-Party Logistics (3PL) services to set up a temperature monitoring service in containers to carry the manufacturer's sensitive products. The messages necessary for setting up such service may include information about (1) the temperature range that will invoke a notification if there is violation, (2) what and how much data to send, (3) where to send the data, etc. Upon receiving a violation notification from a 3PL service (based on a communicated status of containers equipped with smart sensors) the MOM or ERP services will react to rapidly reconfigure the existing supply chain network. This will include placing additional order for the materials; rerouting or recalling the truck; dumping, and picking up new lot; etc.

1.2. Key Enabler: Message Standards for Services Integration

A key to the vision of smart manufacturing systems is that manufacturing services may be efficiently integrated, configured, and reconfigured to meet rapidly changing user requirements and market forces. Yet, the service integration and configurations depend on effective and meaningful service communications.

To provide for such services communications, correct message specifications must be developed for each specific use case (i.e., target business process). Message standards are used to develop such specifications in a precise and efficient manner.

A message standard refers to a set of shared rules and constraints, commonly called schema, that allow correct design and implementation of message exchanges and processing. Well-constructed message standards can enable efficient implementation of communicating services. This is a pre-requisite for efficient integration, configuration, and re-configuration of smart manufacturing systems.

1.3. A Major Problem: Simplistic Practices for Message Standards Development, Use, and Management

Today, a message standard (as a schema) may be very large with hundreds of thousands of data elements. Also, message standards are typically designed and maintained in an implementation-specific manner (e.g., with specific expression syntax and integration patterns), making it difficult to reuse the standard for a new implementation. Additionally, message standards management is very inefficient, often taking months to bring necessary updates to the user community.

These challenges in the current practice exist because message standards management (i.e., development, use, and maintenance) is a very complex affair, yet very poorly supported, leading to simplistic standards management practices. One consequence of these simplistic practices is that message standards are used to represent many use cases. Following such practices, a message schema is incrementally developed in a manner that ignores the intended usage. When the time comes for its use, the message schema is again used in a fashion that ignores any prior usage and it is used for each integration use case independently.

In such simplistic message schema development, the main principle is not to affect backward compatibility. This means that schemas are monotonically growing with all previously added components remaining within the new releases of message schemas. At the same time, the specific usage situation of the message standard additions, which drove the previous incremental change, is discarded. Message standards architects increment message standards with little concern to the previous usage situations. This results in limited re-use of previously added components and extensive additions of new components, giving rise to bloated message standards.

The bloated standards with poor usage documentations extend the problem to the usage phase. The users typically adapt the message standards independently for each specific and isolated use case. Such approach views a usage situation as a separate, independent episode of a message standard customization. Consequently, there is a very limited chance for meaningful reuse of standard components and virtually no chance of quickly achieving interoperable services using the message standards.

These complexities are even more evident for the new generation of dynamic and flexible networks of manufacturing services, which are designed for increased agility. There, the customization of message standards needs to happen very efficiently, as user requirements and market forces rapidly change, while new services are introduced in the ecosystem of providers and users of these services.

1.4. Root Cause: Lack of Support for Message Standards Life-Cycle Management (LCM)

If the message standards are to support efficient reconfiguration of networks of manufacturing services, an unprecedented support of message standards life-cycle management (LCM) is required. Such support needs to address concerns not dealt by the message standards development organizations (SDOs) to any significant extent: traceability, role-based management, collaboration, and continuity of LCM.

Traceability of LCM. The message-standards LCM needs to keep track of the customizations to the message standard to an unprecedented degree of precision. These customizations result in adaptations of message standards for a specific usage situation. That may introduce variability at any level of (i) granularity, (ii) aggregation, (iii) contextualization, and (iv) parallel customizations of components of message standards. Granularity- and aggregation-related traceability needs to track changes done on every level – from simple data types to complex message schemas. Contextualization-related traceability needs to track changes done for any usage situation – from general contexts to very specific contexts. A general context is defined by, say, type of manufacturing process and geographic region. A specific context is defined by, say, role, application, and event types for the message usages. Parallel customization-related traceability needs to track changes for numerous parallel usages of the standard in an ecosystem of service providers and users.

Role-based LCM. First, there is a need to support concurrent and independent customizations of the message standards by individual companies, which take on specific trading partner roles within various business processes. Second, there is a need for timely, synchronized management of message standard updates and releases by responsible standards development organizations.

Collaboration in LCM. First, as concurrent and independent customizations of a message standard take place in a specific company, multiple participants within the company will collaborate to adapt the message standard to their needs. Second, when management of a message standard release takes place at a standards development organization, the organization will collaborate with companies to manage customizations as well as new integration requirements into the standard.

Continuity of LCM. When the customizations of the message standards take place in companies, they will need to consider both prior customizations as well as prior and current releases of the message standard for its impact on existing and future operations of the company. In addition, when maintenance of a message standard release takes place at a standards development organization (SDO), it will

need to consider prior releases, customizations, and current release of the standard from the perspective of impacts on the existing and future users of the standard.

Today, there is a lack of tools that can provide the required traceable, role-based, collaborative, and continuous message standards LCM. This prevents rapid integration and configuration of communicating services and IIoT components.

1.5. Impact of the Lacking Message Standards LCM

If the message standards LCM is not traceable, role-based, collaborative, and supporting continuous processes, there will continue to be many undesirable consequences, preventing the efficient reconfigurability of manufacturing services.

With respect to *traceability* of LCM, while customizations of message standards have been provided at any level of granularity or aggregation, such customizations, in some cases, could not be captured with required precision. For example, differences in the intended usage of data types or elements at multiple places in a message component were not captured precisely, causing integration problems.

The lack of approaches to formally and commonly document contextualized use of a message standard results in limited reuse of message customizations, difficult-to-achieve interoperability, and bloated (i.e., very large and with redundant components) message standards. This makes message standards use inefficient. Cost of a message development is much higher then what may be possible if contextualized use were harnessed. Also, the systems integrations are brittle, with small requirements changes leading to unwanted behavior of the integrated system.

The lack of approaches to capture changes in a message standard, as they occur in parallel customizations, results in a failed opportunity to manage message standards in a proactive manner. Since message standards are large, and standards users are concerned only with a specific task of integration, there is a need to capture and act on independent, parallel changes in a proactive and constructive manner. Without such support, the standards will continue to be very hard and costly to use, especially in the open ecosystems of services with large parallel uses of standards.

With respect to *role-based* LCM, there is no distinction between a standards user and developer. A user has no support to manage multiple revisions of a single component type for multiple use cases, while a developer has no support to work on multiple releases that consider various standards customizations and requirements. These limitations result in inefficient standards development and use processes.

Collaborative capabilities, both within companies, and between companies and SDOs, are very limited and in the form of out-of-band, free-form messages. That, however, causes limited precision of, and low efficiency in, collaborative activities between various roles in collaborating organizations.

With respect to *continuity* of LCM, little exists in the way of support. Publishing a new release is largely a manual affair. No explicit relationships between elements, components, and types of the new and prior releases or customizations are kept. That causes limited understanding of the impact of changes. An existing customization of a message standard that needed to be moved to a new standard release needs to be manually analyzed and updated. This, in turn, is very tedious to accomplish on a realistic scale. Lack of such support prevents any automation when implementing the changes in message standards. Ultimately, updates to a message standard used in existing service configurations may be a very costly proposition.

1.6. Conclusion

Today, message standards development, use, and management take place over many weeks and months. That prevents efficient standards use for service-oriented, modularized applications integrations at a realistic scale, such as in enterprise-wide rationalizations of application functionalities and in an open marketplace of apps and services. In addition, the maintenance process of message standards typically continues indefinitely, presenting even greater challenges to the best practice processes. To enable reconfigurable services, we need to deliver new message standards life-cycle management capabilities. Development of technologies necessary for these capabilities is a challenge that the interoperability R&D community could take on and deliver impactful results to the industry.

1.6. Disclaimer

Any mention of commercial products is for information only; it does not imply recommendation or endorsement by NIST.

1.7. References

[IVE 17] Ivezic N., Kulvatunyou B., Brandl, D., Cho H., Lu, Y., Noller D., Davis J., Wuest T., Ameri, F. “Drilling down on Smart Manufacturing–enabling composable apps”, *US Department of Commerce, National Institute of Standards and Technology*, https://doi.org/10.6028/NIST.AMS.100-8, 2017.

[KUL 16] KULVATUNYOU B., IVEZIC N., MORRIS, K.C., FRECHETTE, S., "Drilling down on Smart Manufacturing–enabling composable apps", *Manufacturing letters*, vol. 10, 2016, p. 14-17.