NIST Advanced Manufacturing Series 100-40

Agile for Model-Based-Standards Development

Brandon Sapp
Melissa Harvey
Marion Toussaint
Sylvere Krima
Allison Barnard Feeney
Herve Panetto

This publication is available free of charge from:
https://doi.org/10.6028/NIST.AMS.100-40

National Institute of Standards and Technology
U.S. Department of Commerce

NIST Advanced Manufacturing Series 100-40

Agile for Model-Based Standards Development

Brandon Sapp
Melissa Harvey
Engineering Strategy
The Boeing Company

Marion Toussaint
Allison Barnard Feeney
Systems Integration Division
Engineering Laboratory

Sylvere Krima
Engisis, LLC

Herve Panetto
University of Lorraine, CNRS, CRAN

This publication is available free of charge from:
https://doi.org/10.6028/NIST.AMS.100-40

March 2021

U.S. Department of Commerce
Gina M. Raimondo, Secretary

National Institute of Standards and Technology
James K. Olthoff, Performing the Non-Exclusive Functions and Duties of the Under Secretary of Commerce for Standards and Technology & Director, National Institute of Standards and Technology

National Institute of Standards and Technology Advanced Manufacturing Series 100-40
Natl. Inst. Stand. Technol. Adv. Man. Ser. 100-40, 31 pages (March 2021)

This publication is available free of charge from:
https://doi.org/10.6028/NIST.AMS.100-40

Abstract

Industries are undergoing a wide-scale digital revolution as they strive towards enabling their digital enterprises. This paradigm shift, from unstructured data sources and paper-based artifacts to Model-Based Enterprise strategies of Digital Threads and Digital Twins, places critical importance on the interoperability of the software applications and information systems involved. One of the primary responses to this new paradigm is the use of neutral, model-based data standards. As the digital revolution is causing rapid changes to systems and industry practices, foundational standards must be able to support rapid incremental change as well. Examination of the current standards development process points to two primary roadblocks inhibiting this advancement: (1) inflated standards development time scales and (2) quality issues in the published standards. An analysis of the key contributing factors to these roadblocks and of available optimization opportunities has resulted in a recommendation that standards development bodies adopt an agile framework and tool-chain. The proposed solution includes backlog management, program increment planning, agile release trains, and offers a means to shorten the development cycle and deliver usable standards to the industry more frequently. This report is illustrated with an analysis of ISO 10303, a complex and widely adopted standard in the manufacturing industry, to demonstrate how complex standards could benefit from a more Agile development lifecycle.

Keywords

model-based standards development; agile; iso 10303; information standard;

Table of Contents

List of Figures

Glossary[1]

[1] Model-Based Standard – a standard based on and published as a model that can be reused or implemented directly within other software
[2] STEP - Standard for the Exchange of Product model data
[3] ISO – International Standards Organization
[4] Epic Owner – Scaled Agile defines the epic owner as the role that is responsible for, defining the minimum viable product, business case, coordinating the portfolio through the Kanban system.
[5] Enterprise Architect – Scaled Agile defines the Enterprise Architect as the role that works with business stakeholders and Solution and System Architects to implement technology initiatives across Value Streams.
[6] Agile Release Train – Scaled Agile defines the agile release train as a set of agile teams.
[7] Product Manager/Owner – Scaled Agile defines product manager/owner as the role that own/represent the customer requirements.
[8] Release Train Engineer – Scaled Agile defines the release train engineer as the role that coordinates with the agile team scrum masters and coordinates the events of the program increment for the agile release train.
[9] System Architect/Engineer – Scaled Agile defines the system architect/engineer as the role that ensures the solution is fit for its intended purpose.
[10] Continuous Delivery – Scaled Agile defines continuous delivery as the framework for continuously releasing value to the end user.
[11] Continuous Exploration – Scaled Agile defines continuous exploration as the set of processes that foster innovation.
[12] Continuous Integration – Scaled Agile defines continuous integration as the set of processes that take a feature from the backlog, develop a solution, test and integrate it.
[13] Continuous Deployment – Scaled Agile defines continuous deployment as the set of processes that take a validated feature and package it for deployment to a production environment.
[14] Agile Team – Scaled Agile defines the agile team as the lowest level team responsible for development of the solution.
[15] Scrum Master – Scaled Agile defines scrum master as the role performing the management of the iteration and increments.
[16] Developer – Scaled Agile defines the developer as the role performing the development of the capability.
[17] Velocity – Scaled Agile defines velocity as the means to measure the capacity and speed of the agile team for an iteration.
[18] Program Increment – Scaled Agile defines the program increment as a set of iterations that are timeboxed.
[19] PI Planning – Scaled Agile defines PI Planning as the event to coordinate and plan the program increment.
[20] Iterations – Scaled Agile defines iterations as the lowest level of timebox, usually referred to as a sprint.

[1] [4]-[20] official definitions can be retrieved from the Scaled Agile glossary at https://www.scaledagileframework.com/glossary/

1. Introduction

1.1. Digitalization of Industry

Industries are undergoing a wide scale digital revolution as they strive towards achieving the promise of digital enterprises. Paper-based information artifacts are being replaced by their digital twins, and unstructured data sources are being replaced by structured data models and digital data.[1] This represents an opportunity to leverage modern computing techniques for improved speed, accuracy, and consistency in manufacturing as well as extending usage of AI, robotics, and other Smart Manufacturing concepts.

This paradigm shift, however, places key importance on information management and the interoperability of the systems involved throughout the product lifecycle (e.g., design, manufacturing, distribution, regulatory compliance)[1]. If not addressed, a lack of interoperability represents an increase in cost and time to industries as well as impedes organizational collaboration needed in a digital enterprise.[2]

One of the main responses to this paradigm shift is the use of neutral model-based data standards to enable interoperability in data exchange of these digital twins. Standards are a key to integrating, exchanging, and accurately interpreting the product data. Standards provide an internationally agreed-upon common language (data format, definitions, etc.) for information exchange between the systems consuming, processing, and generating product data.

1.2. Information Standards and How They Support Business Needs

1.2.1. What are Standards

Today, organizations are more aware of the importance of digital integration and the exchange of information assets. Hence, information standards are developed to ensure optimal interoperability and compatibility between information assets to exploit and process these assets in a consistent fashion. An information standard is a formal definition, agreed upon by a community of industry experts (e.g., healthcare, security, or manufacturing), on how to represent and process domain-specific information. This agreement can be represented as information models, a computer-interpretable representation to digitally represent, exchange, and integrate information across systems.

One of these standardization efforts is the ISO 10303, *Automation systems and integration – Product data representation and exchange* standard series, informally known as STEP[3]. The development of STEP started in the early 90s and is one of the most important information standards that enables product lifecycle collaboration. It includes a set of product data representations and implementation methods to exchange product data. Each of these product data representations is domain-specific and developed by industrial experts to enable standard-based product data exchange. To support its requirements, the STEP community developed its own data modelling language ISO 10303-11[4] and a file format to represent STEP data, called ISO 10303-21[5] *Part 21: Implementation methods: Clear text encoding of the exchange structure*.

As XML emerged as a common file format for data exchange, ISO 10303-28[6] was developed as another way to represent STEP data.

The STEP standards aim to provide a complete and unambiguous description of manufacturing products, usable throughout their life cycle, regardless of the IT support used. The scope of STEP is much broader than other existing computer-aided design (CAD) data exchange formats. It is intended to handle a wide range of product types (electrical, mechanical, composites, ships, architectural, process plant) and cover all life-cycle stages (design, analysis, planning, manufacturing)[3].

1.2.2. **Who Creates Standards**

Developing a standard involves different stakeholders of which there are four major types:

1. the Sponsor, who creates a request for standardization;
2. the Standards Development Organization (SDO), who supervises the standard development process and supports the publication and the maintenance of the standard;
3. the Standards Board, consisting of members of the SDO, who review and approve standards projects before their publication;
4. and the Working Group (WG), composed of domain experts, implementers, end users, standards experts, and technical solution experts.

The role of the WG is to support the development, maintenance, and implementation of the standard. For example, STEP is developed and maintained by ISO and more precisely, by the ISO technical committee TC 184 that deals with automation systems and their integration, and the sub-committee SC 4 that manages the industrial data (ISO TC 184 SC4). The STEP working groups are composed of domain experts from government agencies, the automotive industry, the aerospace industry and the defense industry. All these experts work together to ensure standards developed meet the specific requirements of their industry.

1.2.3. **Development Lifecycle of Model-Based Standards**

The standard development lifecycle is often complex, lengthy and a multi-stage process. In ISO, this lifecycle varies from eighteen to forty-eight months[7] and is composed of six stages (summarized in **Fig. 1**): proposal, preparatory, committee, enquiry, approval , and publication. These different stages tightly control the development process as well as the entry and exit criteria. The process begins at proposal stage when a new work item proposal is submitted by an individual or an entity, called Sponsor, to a Standards Development Organization (SDO). If the proposal is accepted, a collaborative team of experts, called the Working Group (WG), is assembled. This WG works on the development of a committee draft during the preparatory and committee stages. Once this draft is finalized, the enquiry stage begins, during which the draft is reviewed, changed if necessary and approved first by the WG and then, by a balloting group created by the Sponsor. After that, the final draft is submitted to the SDO Board for final approval. Finally, in the publication stage, the standard is published and maintained over the years. In parallel to the standard publication, members of the WG work on developing, testing, and implementing tools, methods, and models to support the standard application.

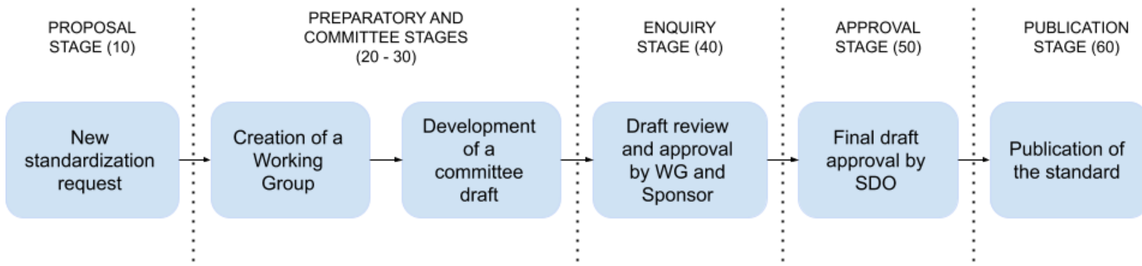

Fig. 1. ISO standard development lifecycle stages

1.2.4. **Who Uses Standards**

Information standards play an important role in businesses by facilitating trade and business interaction, and by supporting interoperability between new and existing technologies. Information standardization also helps to save time while reducing costs: interoperability of the different information assets reduces the need to adapt information formats, saving both time and money. The use of information standards in businesses increases in performance, competitiveness, and transparency because they facilitate the accessibility of information to all stakeholders. [8]

Standards also are a strategic tool to advance innovation. Indeed, the lack of standardization causes a multiplication of information formats that are not necessarily compatible with each other, which can prevent the exchange and sharing of information between stakeholders. For example, Original Equipment Manufacturers (OEMs) need to communicate the full content of Model-Based Definition (MBD) data with their suppliers.

Current limitations in the implementation of standard formats such as STEP result in critical information being lost. Part geometry is correctly exchanged but the tolerances and annotations are lost. This often requires that the native CAD model be sent to the supplier and the burden of interpretation is borne by the suppliers. Another approach is to use STEP to exchange the part geometry augmented by a lightweight geometry for viewing the annotations and notes. Add to this the number of OEM-to-supplier interfaces and supplier-to-supplier interfaces and the problem propagates exponentially throughout the supply chain. Furthermore, collaboration between design partners requires the exchange of geometry, materials, and functional interface data. Geometry needed for spatial analysis may not require the exact fidelity of native CAD geometry. Often a lightweight tessellated representation is better suited to spatial analysis methods such as interference detection and fit analysis. Additionally, the volume of data exchanged is often much larger than in the certification and supply chain use cases. Rather than exchanging parts and assemblies, design integration often exchanges collections of parts by spatial volume up to and including an entire aircraft.

1.2.5. **Why are Standards Complex**

The development of a standard is a relatively long process and it includes many people from different organizations working together. As mentioned before, the members of a standard WG work for different organizations and their contribution to its development is voluntary. Therefore, the resources available depend on the experts' schedules and their organizations' needs, which makes the standards development process long, irregular, and difficult to plan. Moreover, some standards are complex due to their architecture and application domains. In the case of STEP, its development is one of the largest projects that ISO has ever carried out. Six hundred people from many different countries have been involved for the last thirty-five years [3]. STEP is a product-centric standard that covers many industries, creating a large and multi-disciplinary community. requiring the collaboration of a multitude of subject-matter experts.

The STEP architecture had to evolve to support STEP development and growth, transitioning to a modular architecture "to enable the more efficient implementation and deployment of STEP standards without changing the fundamentals of the current technical architecture"[9]. This modular architecture creates new modules by reusing, integrating and extending existing domain-specific information models, known as Application Protocols (APs). Thus, APs are "more interoperable, easier to understand and manage, and quicker to develop" [10].

2. Issues with the Standards Development Lifecycle

We are implementing features and products and using technology that were not invented 18 months ago. No longer can we afford these large monolithic programs that go on for two to three years (24-36 months).

--Bronwyn Clere, Executive Director for Capital Planning & Delivery, Telstra Corporation.

As a part of this research report, an analysis of the duration of an ISO project was conducted[11] by the Planning and Policy Committee (PPC) and published as an official committee document. This study spanned a sample of eight ISO standards through the development lifecycle of publishing 16 cumulative editions of those standards. This study revealed the current average project duration from the approval of a new project to its publication of an International Standard is 43.5 months. Additionally, this study revealed there is an average project length growth of 10% (4.3 months) when comparing the release of edition 1 of a standard to that of edition 2.

Fig. 2 illustrates average project duration by edition (data based on ISO.org project metrics). Note that these metrics are based on a status range from stage 10.99 (new project approval) to 60.60 (international standard published). This is a truncated view and does not include the white paper authoring process which can add as much as an additional year to project duration.

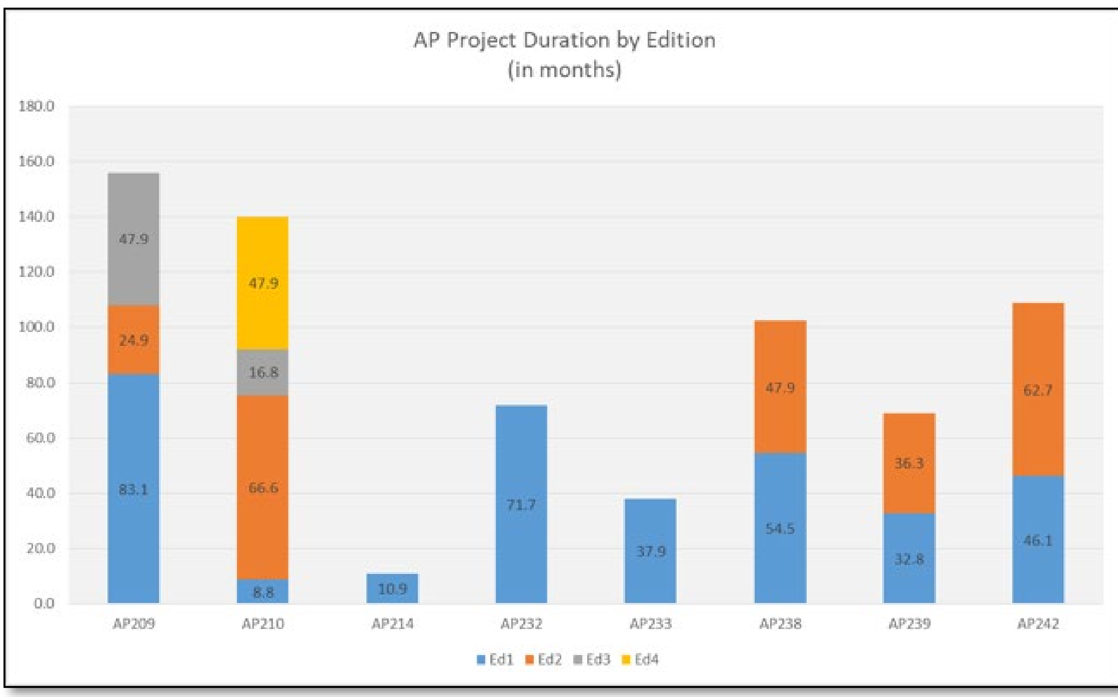

Fig. 2. AP project duration by edition.

This trend of project length growth for standards is at odds with the industry's need for faster releases of incremental functionality. ISO's 2016-2020 strategic directions[12] include six tenants, two of which are (1) "Develop high-quality standards through ISO's global membership", by ensuring we effectively (2) "Engage stakeholders and partners.". To support industry demand and maximize the benefit of standards the quality and length of development of standards must be addressed.

2.1. Development Time

The two primary contributing factors causing an extended development cycle are failure to gain consensus, and resource commitment and management.

For standards with long lifecycles, such as STEP, they carry legacy data and processes, and in some instances use of inadequate tooling. After a new work item is approved during the STEP development process or a defect raised, most teams use a system such as Bugzilla for coordination. Bugzilla was originally designed as a general-purpose bug-tracking tool, however, for STEP development it has been extended beyond that scope. For current STEP development, Bugzilla is used as a requirements management, collaboration /consensus tracking, issue management, and task management system in addition to for version control of source models. This leads to much confusion, maintenance, and misuse of Bugzilla. The result is extended development cycles and multiple rework cycles during the committee stage.

2.1.1. **Volunteer Staff**

Resource commitment for the performance of development work is also a primary obstacle. Factors that affect the commitment of resources include volunteer staff and a virtual distributed team environment. ISO development work is predicated primarily based on volunteer resources from participating companies and organizations. There are several innate challenges to managing volunteer staff including recruitment, retention, and availability.

Due to attrition of volunteer personnel and competing priorities, the available resources fluent in STEP development is declining. Recruitment efforts currently are informally conducted through national standards body nominations that yield a limited pool of resources. New resources face a daunting process to get up-to-speed and deliver value to the standards development process such as applying for ISO roles, installing development software frameworks, completing training, and integrating into the project operating cadence to contribute. Furthermore, as developer resources are shared among ISO project teams the pull on the availability of resources is greater than the supply of volunteer time. This is a constraint that limits the progress of standards development.

2.1.2. **Virtual Distributed Team**

Virtual teams in ISO allow for greater flexibility in executing development tasks despite geographical boundaries. However, this collaboration style has presented unique communication challenges that affect the team's efficiency. Communication obstacles in a virtually distributed team are differences in time zones, work cultures, communication styles, and tool preferences. The outcome is a lack of clear and universal understanding of the task at hand, a deficient sense of project ownership, and lack of trust among team members.

2.2. Quality and Completeness of Standard

There is often a disconnect between the different stakeholder communities involved, such as the standard developers and implementers. For instance, another issue facing the current development lifecycle within ISO TC 184 SC 4is that of quality/completeness concerns of the published standards. CAx-IF[13], which is a joint testing forum between AFNet, PDES, Inc. and prostep ivip tasked with testing MBx-IF STEP translator quality, has continually reported issues of implementability of the standards. The issues can be categorized as data quality, incomplete solutions, and overly complex and non-implementable solutions. Metrics from Bugzilla show 420 bugs (see **Fig. 3**) have been reported in these three categories by the CAx-IF related to ISO 10303 and its application protocols to date.

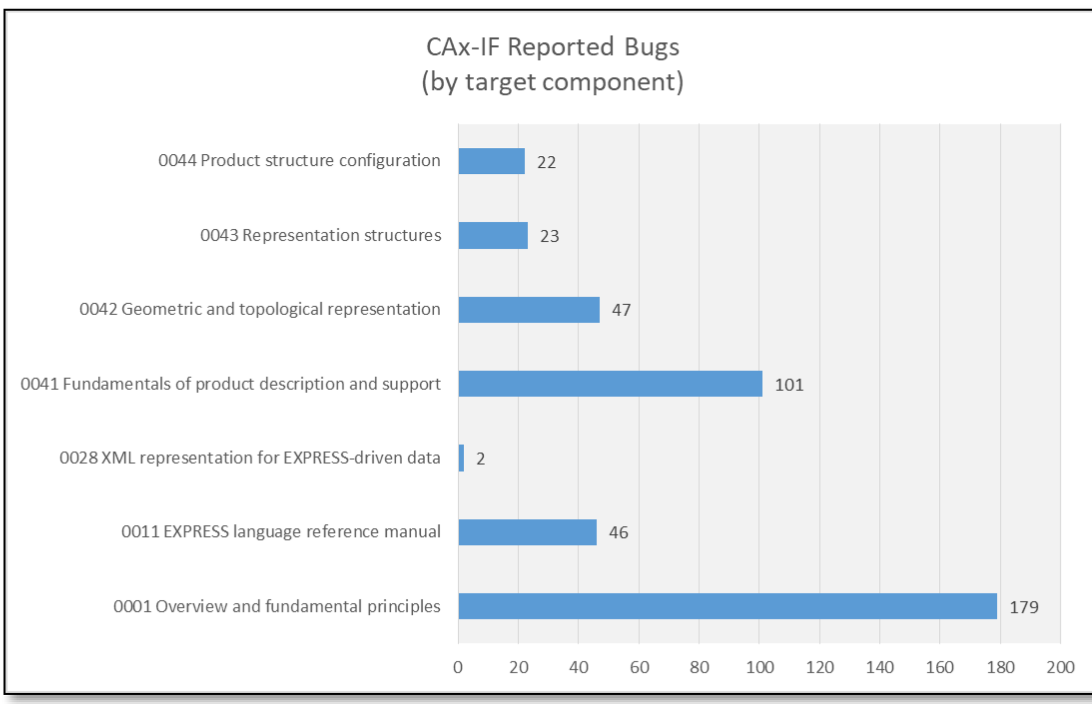

Fig. 3. Research results of analysis Bugzilla data.

While measures have been taken to combat complexity and streamline solutions, such as the transition from a monolithic to a modular architecture, the problem persists. The two primary root causes contributing to the quality/completeness issues of standards are lack of knowledge and lack of an adequate tool-chain.

A robust knowledge of the STEP data model, architecture, and industry domains are necessary to ensure enhancements and defect resolutions have a complete end-to-end solution. The integrated nature of the elements of the STEP data model (application reference model [ARM], module interpreted model [MIM], and integrated resource[IR])[3] mandate that parallel changes take place in each of these elements to ensure continuity of the model. A lack of understanding of integration points leads to inelegant solutions and data quality issues. While checks exist to catch data quality issues these checks are not infallible. Other issues, such as the completeness of solution concepts, require analysis by human interpretation and rely solely on the knowledge level of those performing the work. Understanding the development process and toolset are also factors that affect code quality. Results from a survey[11] of the standards development team reports that 89 % of developers had some rework during their last publication project due to lack of knowledge of the data model, development process, and/or tool-chain. Additionally, this survey revealed that in those instances 7% of cases resulted in more than 50% of rework for a task. As a part of the same survey, the development team identified their knowledge level by tool as either novice, moderate, or expert. These results are summarized in **Fig. 4** below. For all tools, the average number of self-identified experts is only 18% while the average of those at moderate knowledge level is 35% and novice is 47%. These statistics show the magnitude of the gap between knowledge levels of resources for developing standards.

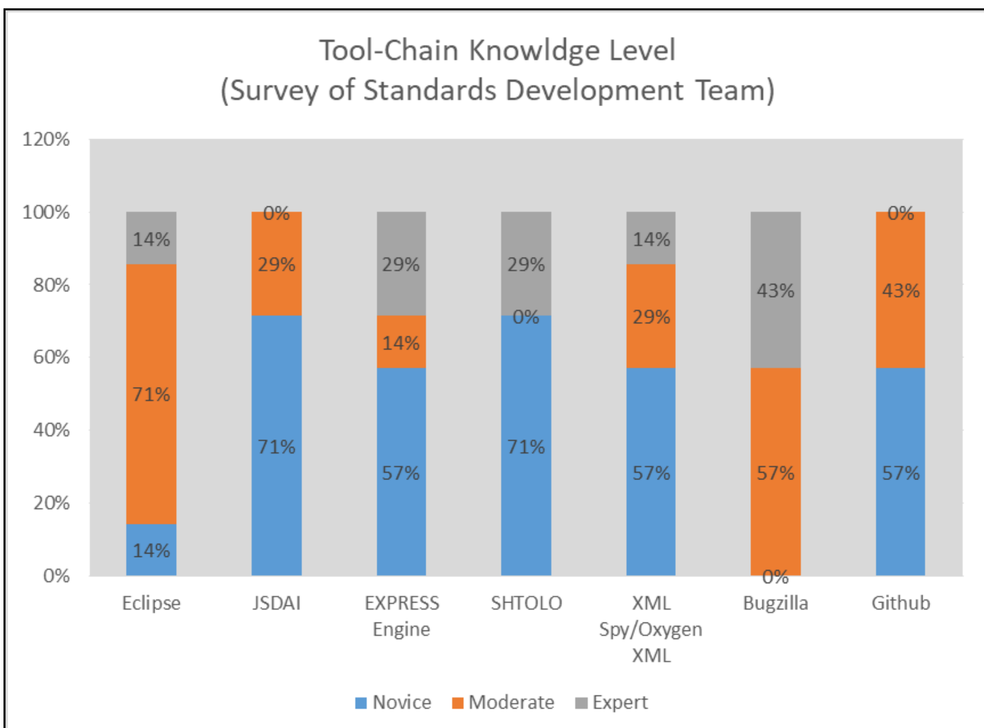

Fig. 4. Survey results of AP developer community.

Another primary cause contributing to quality/completeness concerns is an inadequate toolchain. The current toolchain for model-based standards is fragmented because of the incremental development over the last 30 years to adopt fit-for-use tools to specific tasks. While this has allowed development to go forward it has also resulted in a disjointed toolchain that relies heavily on process controls and manual integration to achieve end-to-end project management and development tasks. This leads to a temperamental development process, poor collaboration and version control, and manually introduced errors.

3. Solution Concepts using STEP as an Example

The previous section presented the most relevant issues in the current development lifecycle in the ISO 10303 community. These issues have a direct and significant impact on the development time length and both quality and completeness of the ISO 10303 community information standards. In this section, solutions are introduced to overcome these issues by improving both the processes and software infrastructure supporting the standard development lifecycle.

3.1. Adoption of Agile Framework

An alternative to traditional management methods (also known as predictive or waterfall management) is the use of adaptive (or Agile) management. Predictive projects rely on 1) all requirements being well defined before execution, 2) delivery of completed products, 3) avoiding changes, which are carefully and formally managed by an integrated control board. Agile projects are different and rely on 1) defining requirements before and during execution, 2) frequent and iterative deliveries of incremental products (i.e., not always fully completed), and 3) welcoming changes.

According to results of a study by the Institute of Electrical and Electronic Engineers, "a majority of respondents' organizational units are using agile and/or lean methods (58%). Furthermore, lean appears as a new player, being used by 24% of respondents, mainly in combination with agile (21%)" [14]. These statistics reinforce the trend to increase the development rate. It can be inferred that the industry requires rapid incremental development capabilities as they strive towards enabling the digital threads for their enterprises.

Standard development teams still use traditional methods to create their products. These traditional methods drive the teams to long phases of requirements documentation, product development, integration, review, and publication. Several organizations have adopted agile to shorten the development cycle and provide a usable product to the users faster. A study analyzed over 300 articles and related benefits showing that in the areas of Schedule, Productivity, Quality and overall ROI benefits of agile methods overshadowed that of traditional methods [15].

Agile is not a new concept. There are various examples of projects using agile concepts such as rapid application development, prototyping, and others. However, since the creation in 2001 of the Agile Manifesto, there have been numerous related implementations and development of new methods. The agile manifesto describes 12 principles – but there are three that hit home for the development of model-based standards [14]:

- "Deliver working software frequently.",
- "Working software is the primary measure of progress.",
- "At regular intervals, the team reflects on how to become more effective, then tunes and adjust its behavior accordingly.".

Note, the term "software" can be replaced with any product such as "data models" or "published data standards". Agile methods include, but are not limited to, practices such as Extreme Programming, Scrum, KANBAN, Backlog Management, and Continuous Delivery [16].

In addition to the specific methods used, there are overarching frameworks that help tie them all together to help large organizations implement at different scales. These frameworks include Scaled Agile (SAFe), Disciplined Agile Delivery (DAD) and Large-scale Scrum (LeSS). While some have criticized SAFe as being too prescriptive, it has seen double the implementations by industry over LeSS and DAD [17]

The FULL SAFe framework by Scaled Agile provides the most comprehensive configuration for deployment [18]and support of large teams. Each project team must analyze their needs and identify which component(s) of the framework will enable them to meet their goals. Scaled Agile has documented case studies that bring real business results including happier, more motivated employees, faster time-to-market, increases in productivity, and defect reductions[18].

While SAFe provides many methods to implement agile, only a few can bring benefit to the development teams of model-based standards: Backlog Management, Agile Release Trains, and Program Increment Planning.

3.1.1. **Backlog Management**

Having a backlog is not the same as managing the backlog. During STEP development, most teams use a system such as Bugzilla to store all the issues. Teams will assign, at bulk, issues to the next milestone and perform a quick reassessment a few times during the length of the project. There are several steps a team can take to actively manage a backlog such as establishing a prioritized ranking and defining a product owner/manager role. The person in this role will be primarily responsible for why, when and what of the product that the development team will deliver. Each team should have a person designated in this role that actively manages the backlog by reprioritizing, adjusting, grooming, and adding to the backlog. This will prevent the backlog from getting too big or out of date. It will also provide reliable work that is ready for the team to assign to a sprint.

3.1.2. **Agile Release Trains**

Using the Scaled Agile definition and framework, an Agile Release Train (ART) is used to group agile teams that operate to develop and deliver "one or more solutions in a value stream." [19]. The ART is a virtual organization that breakdowns the existing silos for development, testing, and publication. The ART is led by a Release Train Engineer (RTE) but has other important roles such as a Product Manager, System Arch, and Business owners/Customers.

For the development of Model-Based Standards like ISO 10303 Application Protocols, an ART can be used to create or revise an edition of an AP, such as a new edition to AP242. Then an agile team can be created for the different domains (see **Fig. 5**) that will deliver capabilities, such as Electrical Wire Interconnect Systems (EWIS), Product Manufacturing Information (PMI) or Additive Manufacturing (AM). These agile teams would each have a Scrum Master, Product Owner and a set of developers.

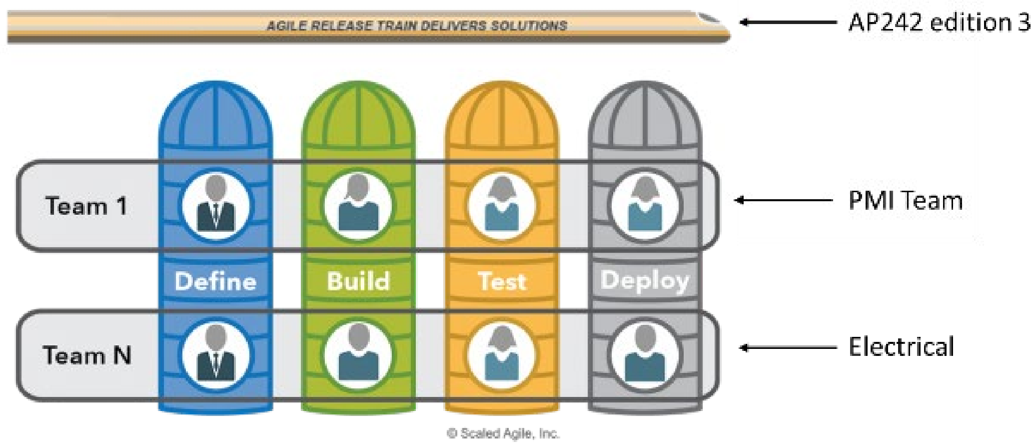

Fig. 5. Agile release train diagram.

An ART can address one of the integrations between teams. Recently, integration issues with AP242 edition 2 teams could have been avoided with synchronized and integrated development iterations. Each team may adopt agile but can and sometimes do operate with different velocities and do not sprint together. (See the Glossary for definitions of agile terminology). The ART addresses that problem by employing systems thinking and applying an operating cadence and synchronization that enables all the teams to sprint together while integrating. There is no limit to the number of trains that can operate together. The concurrent development of AP242e2, AP239e3 and AP243e1 could be managed as multiple trains (see **Fig. 6**).

Fig. 6. Agile release train example.

3.1.3. **Program Increment Planning**

Using the Scaled Agile framework, an ART delivers value in a fixed timebox called a Program Increment (PI). The planning of that Increment is critical to the synchronization of the teams on the train. This synchronization will facilitate planning and limit work in progress.

The RTE and team will decide on the number of iterations, sometimes called sprints, that will be performed in the Increment. All agile teams will follow the same schedule and operate harmoniously. At the beginning of each Increment, all the teams will have a planning event where they decide their velocity, estimate, and plan the work packages. For voluntary teams or teams with resources that are only available part-time, this planning event is critical to establishing the team resource availability and velocity. There are many estimation techniques. The team must avoid detailed analysis and estimation and instead adopt a method like Planning Poker, T-Shirt Sizes, Dot Voting or something similar where the process is quick and relative [20].

After each iteration there is a Plan, Do, Check, Adjust (PDCA) activity where the team can make changes to the plan. At the end of the Increment, a product is demonstrated, or shown, to the industrial customer. The customers are representatives of the industrial stakeholders. This aligns back to the principles of Agile from the Manifesto. For Model-Based Standards development, where the standard is the deliverable, a draft schema would be made available for review or even an example implementation by an implementor using one of the Interoperability Forums.

3.1.4. **Continuous Delivery Pipeline**

One of the most important principles of agile, and specifically the SAFe framework, is Continuous Delivery. Continuous Delivery can be considered as three independent yet related phases: Continuous Exploration, Continuous Integration and Continuous Deployment.

3.1.4.1.Continuous Exploration

Continuous Exploration (CE), as defined by Scaled Agile, is a "process that fosters innovation and builds alignment on what should be built." [21]. CE is when the customers and team members express new ideas that are refined and prioritized in the backlog. The final alignment comes during the PI Planning event.

Some ISO 10303 STEP standards (e.g., AP242, AP209, AP210) are developed in a second-generation version control system called CVS. Recently, the development community has migrated to a third-generation version control system called GIT and integrates with KANBANs and advanced communications tools like ChatOps. This can help agile teams rapidly explore new ideas, validating their ability to integrate while not disrupting the production system or branch line. GIT Branching is a key enabler as the previous generation of source code management did not provide collaboration or development areas.

3.1.4.2.Continuous Integration

Continuous Integration (CI), as defined by Scaled Agile, is a process of developing and integrating in a continuous flow. This will include tasks such as developing, testing, integrating and validating in an environment before production release[22]. CI is made possible with software development best practices that include version control, automated testing, and build automation. The CI process is traditionally managed by software tools such as Bitbucket/Bamboo, Jenkins, AWS CodePipeline, and Gitlab.

To take advantage of the CI capabilities, the development environment must move to a third-generation version control system based on GIT technology. The new tools will allow for continuous exploration as well as continuous integration via the decentralized and distributed architecture, commit before merge capabilities and integrated quality controls. A CI capability will allow standards developers to receive immediate feedback on the pass/failure of their commits by hooking in tools like EXPRESS Engine, JSDAI Compiles, Python scripts or ANT Builds. Immediate feedback will allow developers to fix the issue in the current iteration and not pass it to the end of the flow for someone else to address (in which case the resource may have moved on and not be available). Another feature of some of the CI tools, and really a requirement, is the integration with other issue/task management systems. Jobs can be triggered by lifecycle promotion of the issue/task and feedback so that everyone on the team can have a clear picture of the status of the project deliverables.

Note that the ISO 10303 Extended Architecture already makes use of GIT capabilities but has not developed a continuous integration pipeline for quality and integration automation.

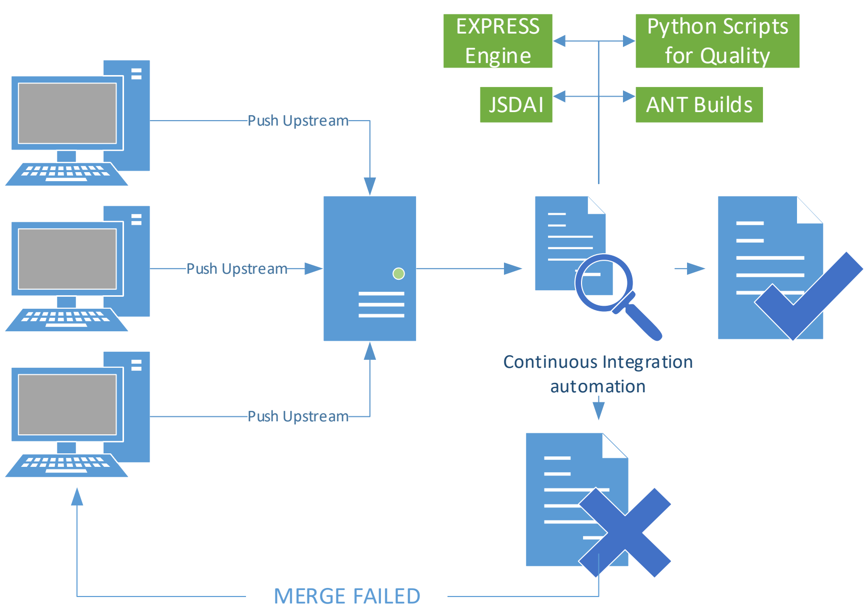

Fig. 7. Proposed development integration

3.1.4.3. Continuous Deployment

Again, Scaled Agile provides a framework for Continuous Deployment (CD). In this stage, the product is deployed, verified, monitored, and set up for responding to issues[23]. The CI process is traditionally managed by software tools such as Jenkins, AWS CodeDeploy and Bamboo.

Standards development teams can use these technologies to automate the deployment of standards to implementer forums, formal ISO Balloting and Publication processes and systems. If Continuous Integration is fully utilized, then the deployment or publication can be performed quickly by automation. Perhaps a better term for this activity in the context of a data standard is Continuous Publication.

3.2. Improved Tool-chain

3.2.1. Requirements Management and Traceability

Critical to the successful execution of a project are requirements, their proper management, and traceability. The needs and expectations of project stakeholders must be correctly captured, documented, implemented, verified, and validated. According to Wiegers [24], successful projects depend on a good understanding of the requirements and the implementation of a collaborative partnership between the stakeholders for requirements development and management. Moreover, Kumar [25] stated that ineffective requirement management is one of the main causes of project failure and that requirements issues can lead to design issues that "are more difficult and expensive to resolve" after the project development is well advanced.

3.2.2. Complete Solution View

In the standards development process, requirements come from different sources: each stakeholder has needs to meet using the standard. During the development process, requirements can change according to the evolution of the stakeholders' needs and new requirements can also be created from feedback on the implemented features for example. Consequently, requirements versioning and traceability should be integrated into this process to document the full lifecycle of each requirement, from its origin to its implementation. Thereby, each stakeholder can track the source of each requirement, the changes made to these requirements and link them to the features through which they are satisfied. Tracking requirements allows the stakeholders to know whether a requirement has been successfully implemented or if it needs to be reworked. Moreover, requirements management makes it easier to identify the person or group of people who issued a requirement, to get more information about it, but also offers a real-time overview of all the requirements to prioritize them.

The development of STEP began several decades ago and since that time, the stakeholders' requirements have evolved, following new business needs and the evolution of the information technologies available. In the STEP development process, ISO document can list two types of requirements: 1) technical requirements about the implementation form(s) of the standard, and 2) domain requirements about the environment in which the standard will be operated, for example, Product Manufacturing Information (PMI). However, information about the requirement type, issuer, or the objective behind each of them is not mandatory. Thus, in some cases, once the features are implemented, it is almost impossible to trace back to the concerned stakeholders to validate their requirements because of the lack of proper traceability.

Requirements traceability is a roadmap that defines where in the standard development process each requirement was implemented. Traceability can also be used to assess the impact of requirements change and expose dependencies between the requirements. On complex projects with multiple parts and different teams, like standard development, identifying these dependencies is a key but long and difficult process.

Additionally, the development of international standards includes many actors from different countries and organizations. This diversity of stakeholders necessitates efficient tools to make it possible for all the different actors to work together. Stakeholders need transparency to be able to understand the role and the activities of everyone on the project for efficient collaboration. For instance, members of the WG might need to know who is working on what and what tasks still need to be done.

3.2.3. **Current Tool Viability**

A key contributing factor to rework in standards development and overall extended duration of the development projects is a lack of clearly defined and traceable requirements. In the STEP community, requirements traditionally have been captured in spreadsheets and documents, and traceability managed through inappropriate tools, such as Bugzilla, which were not designed for this type of effort and lack integration with the documents where requirements are defined. These methods are limited and fragmented. The growing importance of requirements management has led to the development of dedicated requirements management and traceability tools that address these concerns and offer more complete solutions and offer the possibility to mitigate associated rework.

Several requirements management tools are available, and specialize in work management including tasks, user stories, bugs, versioning, and defects tracking within a single and collaborative environment. These are all improvements over the existing tool-chain capability and consistency, however, these tools often represent requirements in plain text and do not meet the STEP community's needs for more advanced requirements management. To correctly manage and validate them, the requirements need to be formally defined in a semantically computer interpretable way that enables the use of reasoning tools to automate consistency checking, validation, and logical prioritization of the requirements. A requirements management solution for information standards such as STEP must support traceability of the requirements against elements in the information models, documents, and deliverables to be able to verify that the requirements are correctly met.

3.2.4. **Challenges**

As previously mentioned, SAFe provides methods to help teams in implementing Agile in their projects (**Fig. 8**), including Backlog management and Agile Release Trains. SAFe also offers methods and processes for requirements management such as its Requirements Model and, Continuously Verify and Validate processes. The SAFe Requirements Model “provides a scalable model that demonstrates a way to express systems behaviors” [26]. The Continuously Verify and Validate processes ensure “that the system works as designed and it meets the needs of the user” [27] and these processes are supported by the Requirements Model. However, SAFe Requirements Model is only a conceptual model and lacks a formal implementable/implementation form that would enable SAFe-compliant tools interoperability.

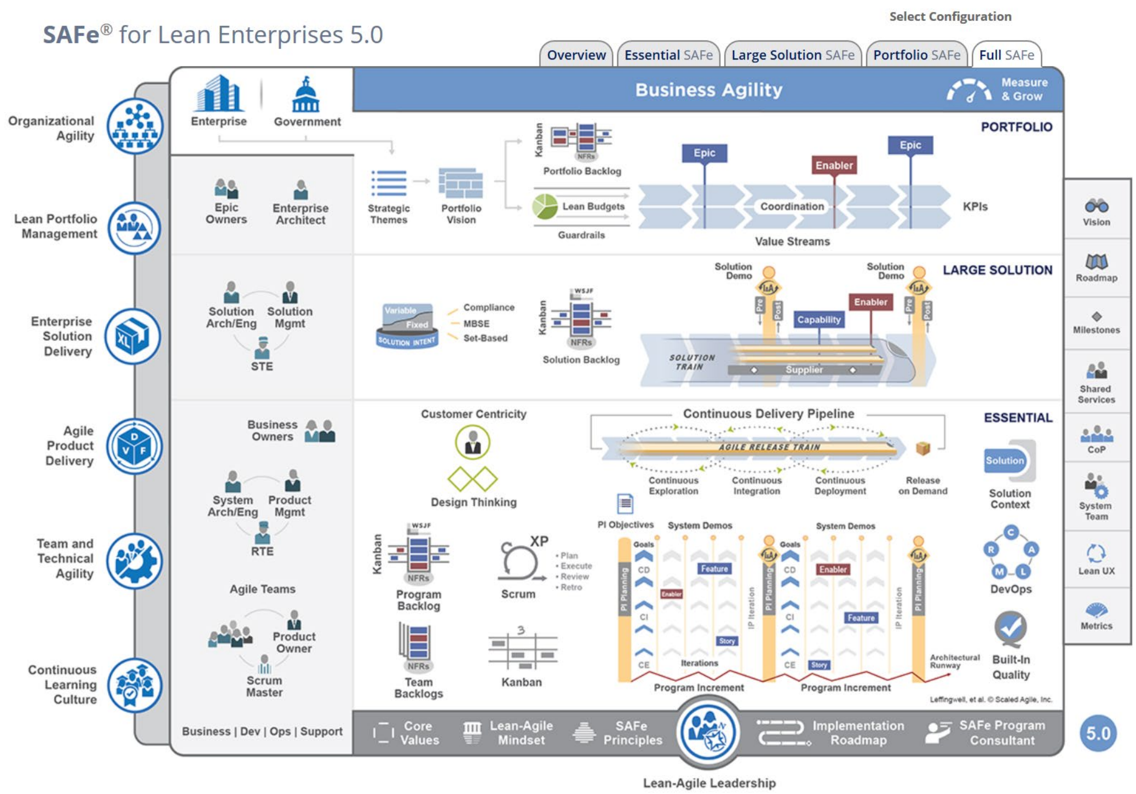

Fig. 8. SAFe framework

While SAFe provides guidelines to implement agile principles and requirements management, some best practices need to be integrated, such as meeting minutes management. Meeting minutes record relevant, important, and critical topics and decisions discussed and agreed upon during online and face-to-face meetings. The archived minutes help to ensure that every member of the development process knows what was discussed, decided, and agreed upon. In international standard development, the different actors are generally geographically dispersed, in different time zones, and working in different teams in parallel, making it challenging for the actors to keep track of all ongoing activities and decisions made. In this context, these minutes are a key communication, reporting and traceability tool, to keep people informed and up to date with the current state of the development process. The STEP development teams host several international meetings and technical workshops during which the stakeholders meet to discuss past, current, and future developments. Experts cannot always participate in all the international meetings and rely on meeting minutes shared with the participants, often by emails, to be and stay informed. Currently, minutes are not consistent and easily accessible by all of the STEP community. Because of the lack of consistency and formal template, it is impossible to link the decisions and the actions taken during the meetings to the task management software and issue log. This formal model should facilitate the understanding of the conduct of the meeting for the people unable to attend.

Furthermore, integrating agile principles with requirements and minutes management into the standard development process requires the use of multiple tools in addition to the many different tools used to develop, implement, and maintain an information standard. Working with all these different tools and technologies means that the development team needs to ensure that there is a proper integration model in place. In the STEP environment, the tools integration situation is even more difficult due to STEP complexity, lifespan, and the use of custom tools and technologies. The community is faced with two integration challenges: 1) the requirement to migrate legacy data, and 2) the need to integrate standard development tools and agile management tools, which should be able to seamlessly collaborate and exchange information.

4. Benefits

The development of standards takes too long and does not align to the pace of industry innovation. Increasing the speed at which a standard is created or revised is fundamental to the adoption of industry standards and associated benefits. The tools and methods will directly aid the developers of the standard by reducing their workflow and improving overall quality with the ultimate customer being the organizations in industry implementing capabilities sooner rather than later.

4.1. Benefits to MBS Developers

The Model-based Standard's developer is one of the biggest beneficiaries of the agile solutions presented in the previous section Agile methods and continuous deployment paired with enhanced requirements traceability will bring many benefits to developers, including:

- Immediate feedback loop to detect and fix issues early
- Increase transparency and visibility to other developers and team members
- Reduce tooling integration complexity
- Improve quality and testability

The feedback loop is probably the most important aspect to a developer. A NIST study on the Impacts of Inadequate Infrastructure for Software Testing shows that 45% of errors are found in the integration stage of development. A bug introduced in by bad requirements or poor coding can take double the cost to fix in the integration stage and triple the cost in the testing stage. If the bug continues to production, then it can cost up to six times to fix than in the stage it was introduced [28].

Teams with continuous delivery spend 50-70% less on problem resolution[29]. Another study of 34,000 open source projects found that teams that use CI, "release twice as often and have developers who are less worried about breaking the build" [30]. A developer will no longer struggle with not having clear and complete requirements – thus reducing rework and wasted time. They can have confidence that the solution they are designing meets the customer's requirements, due to clear traceability to the user story and requirements, and that they have designed a complete solution through the different layers of the data model. By automating the integration and publication processes, the developers will have immediate feedback on the quality of their work and can adjust seamlessly. The Product Increment Planning event and managed backlog will provide a clear statement of work so that the developers can better schedule their time supporting the iterations.

4.2. Benefit to Industry

Besides the Model-based Standards developers, other stakeholders (i.e., the industry) could benefit from a more agile, consistent, and integrated standard development lifecycle. To understand and identify the benefits to other stakeholders, it is important to remember their role(s) in the standard development lifecycle. Industry is both a contributor to and a user of information standards.

As a contributor, organizations satisfy the need for MBS developers (technical/information modeling and domain experts), through funding and/or resources (i.e., experts). Reducing the complexity of the development process will alleviate the involvedness and load of work performed by the MBS developers. Consequently, this will minimize the funding and resources required to support the design, development, publishing, and maintenance of MBSs. It offers industry the opportunity to reduce their interoperability support costs by optimizing efforts and (1) lowering their required contribution and/or (2) to expediting the delivery of MBSs while maintaining their level of contribution. Moreover, an improved planning capability at the developer's level will provide contributors with a more accurate level of contribution required, facilitating their own planning.

Industry users benefit from standards being widely adopted, which requires high-quality information models that are well-documented and designed to seamlessly support a wide range of implementation forms (in response to legacy, current, and future software engineering needs and trends). The benefits to the MBS developers (i.e., immediate feedback and improved quality and testability, as seen in 4.1) will lead to similar benefits at the industry level. Decluttering and streamlining the development process improves the overall quality of standards and increases adoption by reducing the cost of implementation and deployment in heterogeneous environments. Increased adoption of a standard will enable, promote, and unlock new and existing partnerships and opportunities by providing a strong, open, and neutral collaboration platform.

5. Conclusion

Information standards are a key enabler to any digital transformation. They provide a common language to the information systems required to support business and engineering operations in a digital world. However, their development is often a complex process. Developing information standards requires numerous, international, and heterogeneous stakeholders (e.g., users, technical experts, implementers) to collaborate and reach agreements, oftentimes on a volunteer basis. Due to this diversity and growing number of participants, the standard lifecycle is facing inefficiencies that slow down its development, publication, and adoption. In this report we evaluated the ISO 10303-242 standard, a key enabler to the digitization of Manufacturing, and identified key inefficiencies in its lifecycle.

First, due to the physically distributed and volunteer nature of the different teams involved, resource management comes at an extra cost and significantly impact the time to develop information standards (Section 2.1). Second, collaboration between the different communities involved in the standard lifecycle is built upon a legacy infrastructure and being challenged by the inadequacy of that infrastructure to face the complex interdependencies between these communities (Section 2.2). These inefficiencies are shared by all the ISO 10303 family of standards, one of the biggest ISO information standards.

In this report we present a first step towards using the Agile methodology to address the current inefficiencies in the information standards lifecycle (Section 4). Agile offers a way (through processes and tools) to improve the management of the available resources (i.e., volunteer staff) by reducing the need for long-term planning and commitment (see Section 3.1). Agile also documents infrastructure requirements to enable automated quality control and a proper and faster communication channel (Section 3.2.2) between the different communities involved, in order to improve collaboration between these communities and the information standard quality.

Finally, despite these numerous benefits (Section 4), we have identified a clear need for a better overall long-term management and traceability of requirements (see Section 3.2.1), which is currently not offered by either the existing infrastructure or the Agile methodology, and will be required to further improve the quality of information standards with long lifecycles.

References

[1] T. Hedberg, A. B. Feeney, M. Helu, and J. A. Camelio, “Toward a lifecycle information framework and technology in manufacturing,” *Journal of Computing and Information Science in Engineering*, vol. 17, no. 2, Jun. 2017, doi: 10.1115/1.4034132.

[2] M. P. Gallaher, A. C. O’Connor, J. L. Dettbarn, and L. T. Gilday, “Cost Analysis of Inadequate Interoperability in the U . S . Capital Facilities Industry,” NIST, 2004. [Online]. Available: http://fire.nist.gov/bfrlpubs/build04/art022.html.

[3] M. J. Pratt, “ISO 10303, the STEP standard for product data exchange, and its PLM capabilities,” *International Journal Of Product Lifecycle Management*, vol. 1, no. 1, p. 86, 2005, doi: 10.1504/IJPLM.2005.007347.

[4] ISO, “Industrial automation systems and integration -- Product data representation and exchange -- Part 11: Description methods: The EXPRESS language reference manual.” International Organization for Standardization, 1994.

[5] ISO, “10303-21:2002 Industrial automation systems and integration -- Product data representation and exchange -- Part 21: Implementation methods: Clear text encoding of the exchange structure.” International Organization for Standardization.

[6] ISO, “10303-28:2007 Industrial automation systems and integration -- Product data representation and exchange -- Part 28: Implementation methods: XML representations of EXPRESS schemas and data, using XML schemas.” 2007.

[7] ISO, “Target date planner,” 2017. Accessed: Jul. 05, 2020. [Online].

[8] Y. Lu, K. C. Morris, and S. P. Frechette, “Current Standards Landscape for Smart Manufacturing Systems.” 2016, doi: https://doi.org/10.6028/NIST.IR.8107.

[9] A. Barnard-Feeney and D. Price, “A Modular Architecture for STEP,” 2000.

[10] R. Jardim-Gocalves, R. Olavo, and A. Steiger-Garcao, “The Emerging ISO10303 Modular Architecture: In Search of an Agile Platform for Adoption by SMEs,” *International Journal of IT Standards and Standardization Research (IJITSR)*, vol. 3, no. 2, pp. 82–95, 2005, doi: 10.4018/jitsr.2005070107.

[11] M. K. Harvey, “SC4 Standards Development Metrics Analysis,” Nov. 2019. Accessed: Jul. 13, 2020. [Online]. Available: https://isotc.iso.org/livelink/livelink/open/tc184sc4ppc.

[12] ISO, “ISO Strategy 2016-2020,” 2015. Accessed: Jul. 13, 2020. [Online]. Available: https://www.iso.org/files/live/sites/isoorg/files/store/en/PUB100364.pdf.

[13] “MBx-Implementor Forum.” https://www.cax-if.org/ (accessed Jul. 05, 2020).

[14] P. Rodríguez, J. Markkula, M. Oivo, and K. Turula, “Survey on agile and lean usage in finnish software industry,” in *International Symposium on Empirical Software Engineering and Measurement*, 2012, pp. 139–148, doi: 10.1145/2372251.2372275.

[15] D. F. Rico, “What is the ROI of Agile vs. Traditional Methods?,” *TickIT International*, vol. 10, no. 4, pp. 9–18, 2008.

[16] Project Management Institute, *Agile Practice Guide*. Project Management Institute, 2017.

[17] "LeSS Vs SAFe: Which Certification Should You Choose?" https://www.knowledgehut.com/blog/agile/less-vs-safe-which-certification-should-you-choose-and-why (accessed Jul. 05, 2020).
[18] "SAFe 5.0 Framework ." https://www.scaledagileframework.com/ (accessed Jul. 05, 2020).
[19] "Agile Release Train - Scaled Agile Framework." https://www.scaledagileframework.com/agile-release-train/ (accessed Jul. 05, 2020).
[20] "7 Agile Estimation Techniques – beyond Planning Poker." https://technology.amis.nl/2016/03/23/8-agile-estimation-techniques-beyond-planning-poker/ (accessed Jul. 05, 2020).
[21] "Continuous Exploration - Scaled Agile Framework." https://www.scaledagileframework.com/continuous-exploration/ (accessed Jul. 05, 2020).
[22] "Continuous Integration - Scaled Agile Framework." https://www.scaledagileframework.com/continuous-integration/ (accessed Jul. 05, 2020).
[23] "Continuous Deployment - Scaled Agile Framework." https://www.scaledagileframework.com/continuous-deployment/ (accessed Jul. 05, 2020).
[24] K. E. Wiegers, "Karl Wiegers describes 10 requirements traps to avoid," *Software Testing \& Quality Engineering*, vol. 2, no. 1, 2000.
[25] V. S. Kumar, "Effective requirements management," 2006.
[26] "SAFe Requirements Model - Scaled Agile Framework." https://www.scaledagileframework.com/safe-requirements-model/ (accessed Jul. 05, 2020).
[27] "Compliance - Scaled Agile Framework." https://www.scaledagileframework.com/compliance/ (accessed Jul. 05, 2020).
[28] G. Tassey, "The Economic Impacts of Inadequate Infrastructure for Software Testing Final Report Prepared for," 2002. Accessed: Jul. 05, 2020. [Online].
[29] I. Benmoshe, "How to Calculate the ROI of Continuous Delivery Zend Blueprint for Continuous Delivery," 2014. Accessed: Jul. 05, 2020. [Online]. Available: www.zend.com.
[30] M. Hilton, T. Tunnell, K. Huang, D. Marinov, and D. Dig, "Usage, costs, and benefits of continuous integration in open-source projects," in *ASE 2016 - Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, Aug. 2016, pp. 426–437, doi: 10.1145/2970276.2970358.