

**NIST Advanced Manufacturing Series XXX-X**

**Agile for Model-Based-Standards Development**

Brandon Sapp

The Boeing Company

*Business Architecture Integration/  
Industry Data Standards*

Melissa Harvey

The Boeing Company

*Business Architecture Integration/  
Industry Data Standards*

Sylvere Krima

Engisis, LLC

Bethesda, MD

*Office of XXXXXXXXXX*

*First Operating Unit*

Marion Toussaint

Systems Integration Division

Engineering Laboratory

This publication is available free of charge from:

https://doi.org/10.6028/NIST.AMS.XXX-X

September 2019



U.S. Department of Commerce

*Wilbur L. Ross, Jr., Secretary*

National Institute of Standards and Technology

*Walter Copan, NIST Director and Undersecretary of Commerce for Standards and Technology*

Certain commercial entities, equipment, or materials may be identified in this

document in order to describe an experimental procedure or concept adequately.

Such identification is not intended to imply recommendation or endorsement by the

National Institute of Standards and Technology, nor is it intended to imply that the

entities, materials, or equipment are necessarily the best available for the purpose.

**National Institute of Standards and Technology Advanced Manufacturing Series XXX-XX**

**Natl. Inst. Stand. Technol. Adv. Man. Ser. XXX-XX, NNN pages (Month YYYY)**

**This publication is available free of charge from:   
https://doi.org/10.6028/NIST.AMS.XXX-X**

Abstract

Industry is 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 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. However, as this digital strategy represents an increased development rate for industry it must be ensured that as their foundation standards can support rapid incremental development as well. Examination of the current standards development process points to two primary roadblocks inhibiting this advancement: (1) inflated standards development time lengths and (2) quality escapements in the published standards. An analysis of the key contributing factors to these roadblocks and of available optimization opportunities has resulted in a recommendation to pursue the adoption of an agile framework and toolchain by standards development bodies. This proposed solution includes backlog management, program increment planning, and agile release trains and offers a means to shorten the development cycle and provide a usable product to the industry faster.

Keywords

model-based standards development; agile

Table of Contents

[1. Introduction 1](#_Toc25051635)

[1.1. Statement of Industry 1](#_Toc25051636)

[1.2. Information Standards and How They Support Business Needs 1](#_Toc25051637)

[1.2.1. What are Standards 1](#_Toc25051638)

[1.2.2. Who Creates Them 2](#_Toc25051639)

[1.2.3. Development Lifecycle of Model-Based Standards 2](#_Toc25051640)

[1.2.4. Who Uses Them 3](#_Toc25051641)

[1.2.5. Why are they complex 3](#_Toc25051642)

[2. Issues in Current Development Lifecycle 5](#_Toc25051643)

[2.1. Development Time Length 6](#_Toc25051644)

[2.1.1. Volunteer staff 6](#_Toc25051645)

[2.1.2. Virtual Distributed Team 7](#_Toc25051646)

[2.2. Quality/Completeness of Standard 7](#_Toc25051647)

[3. Solution Concepts [using STEP as an example] 10](#_Toc25051648)

[3.1. Adoption of Agile Framework 10](#_Toc25051649)

[3.1.1. Backlog Management 11](#_Toc25051650)

[3.1.2. Agile Release Trains 11](#_Toc25051651)

[3.1.3. Program Increment Planning 12](#_Toc25051652)

[3.2. Improved Tool-Chain 13](#_Toc25051653)

[3.2.1. Requirements Management and Traceability 13](#_Toc25051654)

[3.2.2. Solutions 16](#_Toc25051655)

[4. Benefits 19](#_Toc25051656)

[4.1. Benefits to MBS Developer 19](#_Toc25051657)

[4.2. Benefit to Industry/Enterprise 19](#_Toc25051658)

[5. Conclusion 21](#_Toc25051659)

[References 22](#_Toc25051660)

[Appendix A: Supplemental Materials 23](#_Toc25051661)

[Appendix B: Term Bank 24](#_Toc25051662)

[Appendix C: Change Log 25](#_Toc25051663)

List of Figures

[**Fig. 1.** AP project duration by edition. [1]. 5](#_Toc25051625)

[**Fig. 2.** Research results of analysis Bugzilla data. [2]. 8](#_Toc25051626)

[**Fig. 3.** Survey results of AP developer community. [3]. 9](#_Toc25051627)

[**Fig. 4.** Agile release train diagram. [4]. 12](#_Toc25051628)

[**Fig. 5.** Agile release train example. [5]. 12](#_Toc25051629)

[**Fig. 6.** Proposed development integration. [6]. 18](#_Toc25051630)

# Introduction

## Statement of Industry

Industry is undergoing a wide scale digital revolution as they strive towards enabling their digital enterprises. Paper-based information artifacts are being replaced by their digital twins, and unstructured data sources are being replaced by structured data models. 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). If not addressed a lack of interoperability represents an increase in cost and time to industry [ref to NIST report] as well as impedes organizational collaboration needed in a digital enterprise.

One of the main responses to this 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.

## Information Standards and How They Support Business Needs

### What are Standards

Today, organizations are more aware of the importance of the digital integration and exchange of information assets. Hence, information standards are developed to ensure an optimal interoperability and compatibility between information assets in order to exploit and process these assets in a consistent fashion. An information standard is a formal definition, agreed upon by a community of experts, on how to represent and process domain-specific information. This agreement can be represented as information models, a computer-interpretable representation. Thus, information from different providers can be represented, exchanged and integrated together, seamlessly. Information standards are used in many domains, such as healthcare, security, or manufacturing.

One of these standardization efforts is the ISO 10303 *Automation systems and integration – Product data representation and exchange* standard series, informally known as STEP[ref].  STEP started in the early 90s and is one of the 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 *Part 11: Description methods: The EXPRESS language reference manual* and a file format to represent STEP data, called ISO 10303-21 *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 *Part 28: Implementation methods: XML representation of EXPRESS schemas and data, using XML schemas* was developed as another way to represent STEP data.

It aims 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 (electronic, mechanical, fiber composites, ships, architectural, process plant, furniture) and cover all life-cycle stages (design, analysis, planning, manufacture). [1]

### Who Creates Them

Developing a standard involve different stakeholders. There are four major types of stakeholders : the Sponsor, who is the entity that creates a request for standardization; the Standards Development Organization (SDO) that supervises the standard development process and, supports the publication and the maintenance of the standard; the Standard Board, which consists of members of the SDO, reviews and approves standards projects before their publication; and finally the Working Group that is composed of domain experts, implementers, end users, standards experts and technical solution experts. The role of the WG is to support the development of the standard by writing the standard draft, working on the standard maintenance and developing products that implement the standard. For example, STEP is developed and maintained by ISO and more precisely, by the ISO technical committee TC184 that deals with automation systems and their integration, and the sub-committee SC4 that manages the industrial data. The STEP working groups are composed of experts from government agencies, the automotive industry, the aerospace industry and the nautical industry for instance. All of these experts work together to meet the specific needs of their industry.

### Development Lifecycle of Model-Based Standards

The model-based standard development process consists of six stages: proposal (10), preparatory (20), committee (30), enquiry (40), approval (50), and publication (60). multiple stages. These different stages tightly control the development process as well as the entry and exit criteria. The process begins at proposal stage (10) 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 (20) and committee (30) stages. Once this draft finalized, the enquiry (40) stage began 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 at stage 50. Finally, in publication (60) stage, the standard is published and maintained over the years. In parallel of the standard publication, members of the WG work on developing, testing and implementing tools, methods, and models to support the standard application.

### Who Uses Them

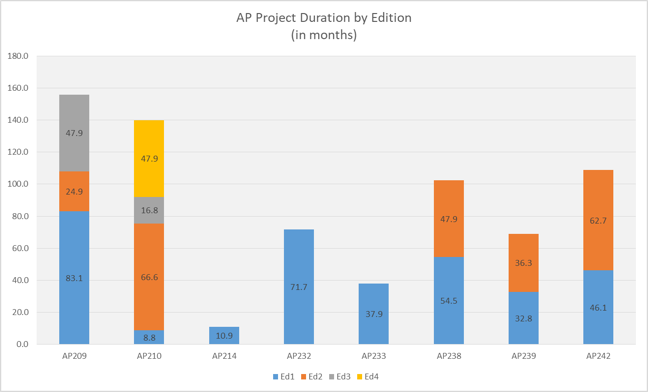
Information standards play an important role in businesses by facilitating trade and business interaction, and by supporting interoperability between new and existing technologies. Information standardization also helps to save time while reducing costs: indeed, because of the interoperability of the different information assets, there is no need to adapt the information formats, which saves 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. Standards also appear as strategic tools 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 the entire aircraft.

### Why are they complex

The development of standard is a relatively long process and it includes a lot of 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 and six hundred people from many different countries have been involved for the last thirty-five years [1]. STEP is a product-centric standard that covers so many areas which has created a big and multi-disciplinary community and needs. STEP covers a lot of domains that need to collaborate and a lot of subjects that need to be combined. In course of its development, the STEP architecture has changed. The objective of this new modular architecture is “to enable the more efficient implementation and deployment of STEP standards without changing the fundamentals of the current technical architecture” [2], i.e. to create new modules by reusing, integrating and extending existing Application Protocols (APs). Thus, APs are “more interoperable, easier to understand and manage, and quicker to develop” [3].

# Issues in Current 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, at Telstra Corporation. As a part of this research report an analysis of ISO project duration was conducted; ISO/TC 184/SC4/PPC N152 contains this full report. This study spanned a sample of 8 standards and 16 edition publishes. This revealed the current average project duration is 43.5 months for the release of an edition of a standard. This calculates the time from stage 10.99 of New Project Approval to stage 60.60 International Standard Published. Additional study reveals this is an average project length growth of 10% (4.3 months) from edition 1 lifecycle lengths to that of edition 2 lifecycles.



**Fig. 1.** AP project duration by edition. [1].

Average project duration by edition (data based on ISO.org project metrics). Note 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.

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 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.” In order to support industry demand and maximize the benefit of standards the quality and length of development of standards must be addressed.

## Development Time Length

There are two primary contributing factors causing an extended development cycle:

1. Failure to gain consensus [International Agreements]
2. Resource commitment/management

Long lifecycle standards carry legacy data and processes, such as the use of inadequate tooling. For instance, in the course of STEP development once a new work item is approved or a defect raised, most teams use a system such as Bugzilla for coordination. Bugzilla was originally designed as a general purpose bug tacking and testing tool (this is from wiki, find a better source) however in STEP development it has been stretch beyond that scope. In existing STEP development application, it 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 many times rework during the committee stage 20.

Resource commitment for the performance of development work is also a primary obstacle. Factors that affect commitment of resources include:

1. Volunteer staff
2. Virtual distributed team

### Volunteer staff

ISO development work is predicated primarily on the basis of volunteer resources from participating member companies and organizations. Innately there are several challenges in managing volunteer staff:

* Recruitment
* Retention
* Availability

Due to factors of attrition and competing priorities the available amount of resources fluent in STEP development is declining. Recruitment efforts currently are informally conducted through member company nomination which yields a limited pool resources. Among these resources there is difficulty in the completion of indoctrinating volunteers through the process of applying for ISO roles, installing development software frameworks, completing training, and getting on a regular schedule 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 speed of development.

### Virtual Distributed Team

Virtual teams allow for greater flexibility in ISO executing development tasks despite geographical boundaries, however this collaboration style has presented unique communication challenges that effect the team’s efficiency. Communication obstacles in a virtually distributed team take many forms from time zone differences, cultural differences, and communication style/tool preference differences. The outcome is a lack of clear and universal understanding of the task at hand, deficient sense of project ownership, and reduced team trust.

## Quality/Completeness of Standard

There is often a disconnect between the communities involved, such as the standard developers and implementers. For instance, another issue facing the current development lifecycle is that of quality/completeness concerns of the published standards. CAx-IF, which is a joint testing forum between AFNet, PDES, Inc. and prostep ivip tasked with testing CAx STEP translators’ quality, has continually reported issues of implement-ability of the standards. These issues fall into three areas and can be categorized as data quality issues, incomplete solutions, and overtly complexities/non implementable solutions. Metrics from Bugzilla show 420 bugs have been reported in these three categories by the CAx-IF against STEP to date.



**Fig. 2.** Research results of analysis Bugzilla data. [2].

While measures have been taken combat complexity and streamline solutions, such as the transition from monolithic to modular architecture, the problem persists. There are two primary root causes contributing to the quality/completeness issues of standards:

1. Lack of knowledge
2. Lack of adequate toolchain

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 (ARMS, MIMs, and IRs) 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 (1) inelegant solutions and (2) data quality issues. While quality checks exist to catch data quality issues these checks are not infallible. Other issues, such as the elegant or clumsy nature of solution concepts, require analysis by human interpretation and rely solely on the knowledge level of the human resource performing the work. Understanding of the development process and toolset are also factors that affect code quality. Results from a survey of the standards development team reports that 89% of developers incurred some amount of rework during their last publication project due to lack of knowledge of the data model, development process, and/or tool-chain. Of that 89% rework 7% reported more than 50% of rework was required. For a view of the survey results in their entirety reference appendix B. See the below figure for survey results of perceived knowledge level by tool-chain application.

****

**Fig. 3.** Survey results of AP developer community. [3].

The second primary root causes contributing to quality/completeness concerns is an inadequate tool-chain. The current tool-chain for model-based standards is fragmented, which is the result of its 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 tool-chain 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.

# Solution Concepts [using STEP as an example]

In the previous section we have 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 we introduce solutions to overcome these issues by improving both the processes and software infrastructure supporting the standard development lifecycle.

## Adoption of Agile Framework

According to survey 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%)” [4] (Rodriguez, 2013). These statistics reinforce the increased development rate of industry towards a trend of rapid incremental development 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 as a means to shorten the development cycle and provide a usable product to the users faster. A study by Dr. David Rico was performed by analyzing over 300 articles and related benefits. His analysis showed that in the areas of Schedule, Productivity, Quality and over all ROI benefits of agile methods overshadowed that of traditional methods [5].(Rico, David F., 2008)

Agile, itself, 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 manifesto describes 12 principles – but there are three that hit home for the development of model-based standards: 1) “Deliver working software frequently.”, 2) “Working software is the primary measure of progress.", 3) “At regular intervals, the team reflects on how to become more effective, then tunes and adjust its behavior accordingly.”. Agile Manifesto [6] (Beck et al., 2001). 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.

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 [7]. (KnowledgeHut, 2018)

The FULL SAFe framework by Scaled Agile provides the most comprehensive configuration for deployment [8]. (Scaled Agile, 2018a) Each project team must analyze their needs and identify which component(s) of the framework that 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 [9]. (Scaled Agile, 2018b)

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

### Backlog Management

Having a backlog isn’t the same as managing the backlog. In the course of 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 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 and 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. Several software tools exist to manage backlogs in an agile framework, such as Atlassian’s JIRA, Micrsoft’s TFS, VersionOne or PivotalTracker.

### 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.” [10] (“Agile Release Train – Scaled Agile Framework,” n.d.). 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, Business owners/Customers.

For the development of Model-Based Standards like ISO 10303 Application Protocols, an Agile Release Train can be used to create/revise an edition of an AP, such as a new edition to AP242. Then an agile team can be created for the different domains that will deliver capabilities, such as Electrical Wire Harness [EWH], 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. 4.** Agile release train diagram. [4].

An ART can address one of the biggest problems with multiple teams. That is a regular integration. Recently integration issues with AP242 e2 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. 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 number of trains that can operate together. The concurrent development of AP242e2, AP239e3 and AP243e1 could be managed as multiple trains.



**Fig. 5.** Agile release train example. [5].

### Program Increment Planning

Using the Scaled Agile framework, an ART delivers value in a fixed timebox called a Program Increment. The planning 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 amount 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 of 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 [11].(“7 Agile Estimation Techniques – beyond Planning Pokerâ”AMIS Oracle and Java Blog,” n.d.)

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 available to the customer. This aligns back to the principles of Agile from the Manifesto. For Model-Based Standards this could be a draft standard or schema of the AP that is made available for testing in an Implementer Forum.

## Improved Tool-Chain

### Requirements Management and Traceability

Key elements of a successful project are requirements and, their proper management and traceability. The goal is to ensure that the needs and expectations of the project stakeholders are correctly captured, documented, implemented, verified and validated. Indeed, Wiegers [12] wrote that 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 [13] 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.

#### Complete Solution View

In the standard development process, requirements come from different sources: each stakeholder has needs to meet using this 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 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 because of the change of the business needs and the evolution of the information technologies available. In the STEP development process, requirements can be listed in ISO documents, and it happens sometimes that there is no information about the requirement issuer or the objective behind each of them. Thus, in some cases, once the features are implemented, it is almost impossible to get back to the concerned stakeholders to validate their requirements because of the lack of traceability. Besides, in STEP, there are two different types of requirements: technical requirements, which are the requirements about the implementation of the standard, and domain requirements, which are the requirements about the environment in which the standard will be operated, for example, Product Manufacturing Information (PMI), Mechanical and Electrical Wire Harness (EWH).

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 of the different actors to work together. Indeed, the stakeholders need to be able to understand the role and the activities of everyone on the project for a good collaboration. In the same way, it can be useful for all members of the WG to know who is working on what and what tasks still need to be done.

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. Indeed, on complex projects with multiple parts and different teams working on it like standard development, it can be pretty long and difficult to manually determine what part and who is affected by the change.

#### 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. Requirements traditionally have been captured in spreadsheets, documents, and traceability managed through tools like Bugzilla. These methods are limited and fragmented approaches. The growing importance of requirements management has led to the development of dedicated requirements management and traceability tools which address these concerns and offer more complete solutions and offer the possibility to mitigate associated rework.

There are several requirements management tools on the market. Jira is one of many very popular software in Agile projects. Jira is a tool specialized in work management including bug/defects/task/user stories tracking. These are all improvements over those in the existing tool-chain, however, when requirements are completed they are removed from the backlog making their traceability a less than intuitive operation. Another software solution is Modern Requirements. It provides a collaborative requirements management platform. This software also offers requirements traceability and impact analysis. Besides, it can be easily integrated with bug issues tools like JIRA and backlog tools like Microsoft TFS. ReqView is another popular requirement management tool that allows to capture structured requirements and trace these requirements between requirements elicitation, design, and tests.

These different tools, in which requirements are often represented in plain text, are not necessarily adapted to our needs for advanced requirements management.

Indeed, to both correctly manage and validate the requirements, these need to be formally defined in a semantically and structurally computer interpretable way. This computer interpretable and formal definition of the requirements will allow us to leverage reasoning tools to automate consistency checking, validation and logical prioritization of the requirements. Moreover, the solution should support traceability of the requirements against elements in the information models, documents and deliverables to be able to verify that the requirements are correctly meet.

#### Future Research Needed

As previously mentioned, SAFe provides methods to help teams in implementing Agile in their projects, including Backlog management and Agile Release Trains. SAFe also offers methods and processes for requirements management such as the SAFe Requirements Model and, Continuously Verify and Validate processes. The SAFe Requirements Model “provides a scalable model that demonstrates a way to express systems behaviors” [14], like features (*user account, notifications, keywords search)*, stories (*As a user, I want to be able to run the software on Windows and on Mac.*), and non-functional requirements (*The software should be reliable in order to be able to resist attacks and handle system errors.*). The Continuously Verify and Validate processes ensure “that the system works as designed and it meets the needs of the user” [15] 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.

Moreover, while SAFe provides guidelines to implement Agile principles and requirements management, there are still some practices that are missing and need to be integrated such as meetings’ minutes.  By definition, meeting minutes’ record relevant, important, and critical topics and decisions discussed and agreed upon during meetings (online and face-to-face). These notes, archived and available to the community, helps 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 geographical dispersed, in different time zones, and working in different teams in parallel, making it challenging for the different actors to keep track of all ongoing activities and decisions made. In this context, these minutes are a key communication, reporting and traceability tool, in order to keep people informed and up to date with the current state of the development process. The STEP development team(s) hosts several international meetings such as the ISO TC184/SC4 or PDES workshops during which the different stakeholders meet face-to-face to discuss past, current, and future developments.  These meetings are held twice a year and STEP experts cannot always participate to all the international meetings. Meetings minutes are taken during both the ISO and PDES meetings, and they are shared with the participants, often by emails. These minutes are not necessarily very detailed and easily accessible by all the STEP community. Moreover, the meeting minutes are not written according to a formal template. Therefore, implementing a formal model for taking minutes would make it possible to link the decisions and the actions taken during the meetings to the tasks or bugs in Jira. This formal model should also 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 on top of the many different tools used to develop, implement, and maintain an information standard. Working with all of these different tools and technologies means that the development team needs to ensure that there is a proper integration model in place. Indeed, there is often no formal integration model to ensure perfect interoperability and integration between all of these technologies and tools. In the STEP environment, the tools integration situation is even more complex due to STEP complexity, lifespan, and the use of bespoke tools and technologies. STEP development began decades ago, and its range has expanded a lot over the past few years. With the continuous emergence of new technologies, the tools used for STEP development have changed since its creation. There are two types of integration to implement for STEP: on the one hand, the integration between the old and the current technologies. Regarding the first integration, some legacy data need to be migrated in the technologies currently used. On the other hand, the integration between the tools used to implement STEP and the Agile management tools. The tools chosen to implement the Agile method in the STEP development should be able to easily collaborate with the tools currently used.

Finally, the standard development process is not the same as usual IT projects process. Indeed, the development of a standard relies on the voluntary contributions of the members of the WG. The people and resources available vary, which makes the development process more irregular than in standard Agile projects, in which the development team is dedicated to the project full-time. Besides, standard experts are geographically dispersed and with the time difference, everybody is working according to their time zones, which can make it difficult to adopt some of the Agile practices. Without a full-time development team, the Agile sprint can’t be as regular and intense as normal Agile projects sprints, which means that in the case of the standards development, there are challenges that we still need to work on to ensure that the Agile method can meet all their needs and constraints.

### Solutions

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

#### *Continuous Exploration*

Continuous Exploration [CE], as defined by Scaled Agile, is a “process that fosters innovation and builds alignment on what should be built.” [16] (“Continuous Exploration – Scaled Agile Framework,” 2018) CE is when the customers and team members express new ideas, 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.

#### *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 [17]. (“Continuous Integration – Scaled Agile Framework,” 2018)

CI is made possible with software development best practices that include version control, automated testing, and build automation. There are many choices in the industry 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 available]. Another feature of some of the CI tools, and really a requirement, is the connection to 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 status of the project deliverables. NOTE: 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. 6.** Proposed development integration. [6].

#### *Continuous Deployment*

Again, Scaled Agile provides a framework for Continuous Deployment [CD]. In this stage, the product is deployed, verified, monitored and setup for responding to issues [18]. (“Continuous Deployment – Scaled Agile Framework,” 2018) There are many tools for this 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.

# Benefits

As noted in the beginning of this paper – 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 enable capabilities like the Digital Twin. The tools and methods mentioned in this paper will directly aid the developer of the standard by reducing their flow and improving overall quality with the ultimate customer being the organizations in industry implementing capabilities sooner rather than later.

## Benefits to MBS Developer

The Model-Based Standard’s developer is one of the biggest beneficiaries of the solutions identified in this paper. Agile methods and continuous deployment paired with enhanced requirements traceability will bring many benefits to the developer, including:

* Immediate feedback loop to detect and fix issues early
* Increase transparency and visibility to other developers and team members
* Avoid “integration hell”
* 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 then in the stage it was introduced [19]. (Tassey, Gregory, 2002).

Teams with continuous delivery spend 50-70% less on problem resolution [20]. (Benmoshe, n.d.) 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.” [21] (Hilton, Tunnell, Huang, Marinov, & Dig, 2016) 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 developer will have immediate feedback on the quality of their work and can make adjustments straightaway. The Product Increment Planning event and managed backlog will provide clear statement of work so that the developer can schedule their time supporting the iterations.

## Benefit to Industry/Enterprise

Besides the MBS developers, other stakeholders The industry from a more agile, consistent, and integrated standard development lifecycle. In order 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 (as seen in 4.1). 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.

As a user, industry benefits 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. As shown in 4.1, decluttering and streamlining the development process improves the overall quality of the standards, facilitating and increasing its adoption by cheapening and fastening its implementation and deployment in heterogeneous environment. 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.

# 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 very complex process. Developing information standards requires numerous, international and heterogeneous stakeholders (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 Smart Manufacturing vision, 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 (see X). 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 (see X). 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 (see 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 (see 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 (see Section 4), we have identified clear needs for a better overall management and traceability of requirements (see Section 3.2.1), which is currently not offered by neither the existing infrastructure or Agile, and will be required to further improve the quality of the information standards.

# References

1. Michael J. Pratt. Introduction to ISO 10303 – the STEP standard for product data exchange. Journal of Computing and Information in Engineering, 2001.
2. Allison Barnard Feeney, David M. Price. A modular architecture for STEP. Journal of Computing and Information Science in Engineering, 2002.
3. Ricardo Jardim-Gonçalves, Ricardo Olavo, Adolfo Steiger-Garcao. The Emerging ISO 10303 Modular Architecture : In search of an Agile Platform for Adoption by SMEs. International Journal of IT Standards and Standardization Research (IJITSR), 2005.
4. Rodriguez, P., Markkula, J., Oivo, M., & Turla, K. (2013, March 7). Survey on agile and lean usage in finnish software industry. Retrieved August 8, 2019, from https://ieeexplore.ieee.org/abstract/document/6475408.
5. Rico, David F. (2008). What if the ROI of Agile vs Traditional Methods.
6. Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001). Manifesto for Agile Software
7. KnowledgeHut. (2018, May 11). LeSS Vs SAFe: Which Certification Should You Choose And Why? Retrieved July 18, 2019, from https://www.knowledgehut.com/blog/agile/less-vs-safe-which-certification-should-you-choose-and-why
8. Scaled Agile. (2018a). Introducing SAFe 4.6.
9. Scaled Agile. (2018b, November). SAFe 4.6 Introductionâ”Overview of the Scaled Agile Framework for Lean Enterprises.
10. Agile Release Train – Scaled Agile Framework. (n.d.). Retrieved July 19, 2019, from /agile-release-train/
11. 7 Agile Estimation Techniques – beyond Planning Pokerâ”AMIS Oracle and Java Blog. (n.d.). Retrieved July 19, 2019, from https://technology.amis.nl/2016/03/23/8-agile-estimation-techniques-beyond-planning-poker/
12. Wiegers, K. Karl Wiegers Describes 10 Requirements Traps to Avoid. Software Testing & Quality Engineering Journal, 2000.
13. Kumar, V. S. (2006). Effective requirements management. Paper presented at PMI Global Congress 2006—EMEA, Madrid, Spain.
14. SAFe Requirement model - Scaled Agile Framework (2018). Retrieved July 25, 2019, from https://www.scaledagileframework.com/safe-requirements-model
15. Compliance - Scaled Agile Framework (2018). Retrieved July 25, 2019, from https://www.scaledagileframework.com/compliance
16. Continuous Deployment – Scaled Agile Framework. (2018). Retrieved July 19, 2019, from https://www.scaledagileframework.com/continuous-deployment/
17. Continuous Exploration – Scaled Agile Framework. (2018). Retrieved July 19, 2019, from https://www.scaledagileframework.com/continuous-exploration/
18. Continuous Integration – Scaled Agile Framework. (2018). Retrieved July 19, 2019, from <https://www.scaledagileframework.com/continuous-integration/>
19. Tassey, Gregory. (2002, May). The Economic Impacts of Inadequate Infrastructure for Software Testing. NIST.
20. Benmoshe, I. (n.d.). How to calculate the ROI of Continuous Delivery. 18.
21. Hilton, M., Tunnell, T., Huang, K., Marinov, D., & Dig, D. (2016). Usage, costs, and benefits of continuous integration in open-source projects. Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering - ASE 2016, 426–437. https://doi.org/10.1145/2970276.2970358
22. Development. Retrieved July 18, 2019, from https://agilemanifesto.org/
23. <https://www.iso.org/files/live/sites/isoorg/files/archive/pdf/en/iso_strategy_2016-2020.pdf>
24. <https://www.iso.org/files/live/sites/isoorg/files/developing_standards/docs/en/Target_date_planner_4_ISO_standards_development_tracks_2017.pdf>
25. <https://www.cax-if.org/>
26. <https://www.cio.com/article/3304276/speed-adaptation-and-the-pace-of-change.html>

# Appendix A: Supplemental Materials

Brief description of supplemental files – delete if not applicable.

1. To add materials on standards development time-line analysis
2. To add survey on standards development process
3. To add survey on standards development tool-chain

# Appendix B: Term Bank

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 Arch/Eng – Scaled Agile defines the system architect/engineer as the role that ensures the solution is fit for it’s 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 deploymen 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 refered to as a sprint.

# Appendix C: Change Log

If updating document with errata, detail changes made to document – delete if not applicable.