Introduction

An annotated bibliography of papers, reports and books regarding ISO/IEC/IEEE 42010:2011 (revision of the former IEEE Std 1471:2000). Originally prepared for ISO/IEC JTC1/SC7 WG42, the Architecture Working Group of the Systems and Software Engineering Subcommittee of ISO. The bibliography includes 1) items which were inspirations for the Standard; 2) items citing or about the Standard or its development; and 3) items inspired by or built on the Standard and its concepts.

Note: with version 4.x, we switch to producing the bibliography using the \texttt{biblatex} package. There may be errors.

Please send corrections and additions to r.hilliard@computer.org.

References

	extit{ArchiMate 2.0 Specification}. Jan. 2012. URL: \url{http://www.opengroup.org/archimate/}.

Annotations: ArchiMate provides definitions of a number of architecture viewpoints, and provides a useful classification scheme for viewpoints.


Annotations: An eloquent argument for the need for specialized viewpoints in architectural description: “Unfortunately, in contrast to building architectures, we have yet to agree on what the appropriate software structures and views should be and how to represent them. One of the reasons for the lack of consensus on structures, views, and representations is that software quality attributes have matured (or are maturing) within separate communities, each with their own vernacular and points of view.”  

1
Annotations: Reasoning frameworks have several properties similar to architecture viewpoints.

Abstract: Determining whether a system will satisfy critical quality attribute requirements in areas such as performance, modifiability, and reliability is a complicated task that often requires the use of many complex theories and tools to arrive at reliable answers. This report describes a vehicle for encapsulating the quality attribute knowledge needed to understand a system’s quality behavior as a reasoning framework that can be used by nonexperts. A reasoning framework includes the mechanisms needed to use sound analytic theories to analyze the behavior of a system with respect to some quality attribute. This report defines the elements of a reasoning framework and illustrates the reasoning framework concept by describing several reasoning frameworks and how they realize these elements.


Abstract: As military and civil software-intensive information systems grow and become more and more complex, structured approaches, called architecture frameworks (AF), were developed to support their engineering. The concepts of these approaches were standardised under ISO/IEC 42010, Systems and software engineering — Architecture description. An Architecture Description is composed of Views, where each View addresses one or more engineering concerns. As mentioned in the standard, a multi-viewpoint approach requires the capacity to capture the different views, and maintain their mutual consistency. This paper addresses primarily the problem of integrating a model-based security risk assessment view to the mainstream system engineering view(s) and, to a lesser extent, the problem of maintaining the overall consistency of the views. Both business stakes and technical means are studied. We present two specific approaches, namely CORAS and Rinforzando. Both come with techniques and tool support to facilitate security risk assessment of complex and evolving critical infrastructures, such as ATM systems. The former approach offers static import/export relationships between artefacts, whereas the latter offers dynamic relationships. The pros and cons of each technical approach are discussed.


Annotations: Proposes to extend the IEEE 1471 conceptual model with “semantic models” and architecture signatures to bridge the gap between business process models and enterprise architectures.

Jan Bosch. “Software Architecture: The Next Step”. In: *Proceedings First European Workshop Software Architecture (EWSA 2004)*. Ed. by Flavio Oquendo, Brian Warboys, and Ron Morri-
Abstract: This position paper makes the following claims that, in our opinion, are worthwhile to discuss at the workshop. 1) The first phase of software architecture research, where the key concepts are components and connectors, has matured the technology to a level where industry adoption is wide-spread and few fundamental issues remain. 2) The traditional view on software architecture suffers from a number of key problems that cannot be solved without changing our perspective on the notion of software architecture. These problems include the lack of first-class representation of design decisions, the fact that these design decisions are cross-cutting and intertwined, that these problems lead to high maintenance cost, because of which design rules and constraints are easily violated and obsolete design decisions are not removed. 3) As a community, we need to take the next step and adopt the perspective that a software architecture is, fundamentally, a composition of architectural design decisions. These design decisions should be represented as first-class entities in the software architecture and it should, at least before system deployment, be possible to add, remove and change architectural design decisions against limited effort.


Annotations: Describes a framework and formalization of relations and compositions between architectural models (and views).


Annotations: Presents a taxonomy of mechanisms for view relations.


Abstract: Architectural descriptions representing and modeling the architecture of a system or parts thereof are typically used in the engineering disciplines to plan, develop, maintain, and manage complex systems. Primarily originating from construction engineering, the means of architecting and architectural descriptions have been successfully transferred to related disciplines like
software engineering. While a rich and formal theory on conceptual modeling exists as well as frameworks on how to approach architectural descriptions, e.g. the ISO standard 42010, only few attempts have yet been made to integrate the prescriptions and guidelines from these sources into a formal architectural description framework. In this paper, we establish such a framework against the background provided by the ISO standard 42010 by formally defining the terms concern, view, viewpoint, and architectural description. Further, an outlook discusses potential application areas of the framework.


Annotations: Describes an integrated development environment (IDE) for physical system architecting using concepts of IEEE 1471.


Abstract: Architectures come about through forces and needs other than those captured in traditional requirements documents. A business goal expresses why a system is being developed and what stakeholders in the developing organization, the customer organization, and beyond aspire to achieve through its production and use. Business goals can provide the rationale for requirements and help identify missing or superfluous requirements. Business goals can also influence architectures directly, even without affecting requirements at all. A business goals viewpoint can help architects and organizations capture their business goals in a precise and unambiguous form, which in turn will help architects design systems that are more responsive to organizational needs.


Abstract: Rwanda, one of the smallest and most densely populated countries in Africa, has made rapid and substantial progress towards designing and deploying a national health information system. One of the more challenging aspects of the system is the design of an architecture to support: interoperability between existing health information systems already in use in the country; incremental extension into a fully integrated national health information system without substantial re-engineering; and scaling, from a single district in the initial phase, to national level without requiring a fundamental change in technology or design paradigm. This paper describes the key requirements and the design of the current architecture using the ISO/IEC/IEEE 42010 standard architecture descriptions. The architecture takes an Enterprise Service Bus approach. A partial implementation and preliminary analysis of the architecture is given. Since these challenges are experienced by other developing African countries, the next steps involves creating a generic architecture that can be reused for health information exchange in other developing African countries.


Annotations: Uses IEEE 1471 concepts to conduct knowledge engineering on multidisciplinary course and curriculum design.


Annotations: Tackles issues of inter-view consistency via unification in a multiple viewpoint setting based on RM-ODP.

Abstract: There have been a number of proposals to split the specification of large and complex systems into a number of inter-related specifications, called viewpoints. Such a model of multiple viewpoints forms the cornerstone of the Open Distributed Processing (ODP) standardisation initiative. We address two of the technical problems concerning the use of formal techniques within multiple viewpoint models: these are unification and consistency checking. We discuss the software engineering implications of using viewpoints, and show that object encapsulation provides the necessary support for such a model. We then consider how this might be supported by using object-oriented variants of Z.


Annotations: Symphony is a viewpoint-driven approach to reconstruction of software architectures.


Annotations: Uses IEEE 1471 conceptual framework as starting point for an ontology of complex communications and quality of service.


Annotations: This paper proposes the use of a basic viewpoint as a basis for defining and relating viewpoints for distributed application design.

Abstract: The architectural design of distributed enterprise applications from the viewpoints of different stakeholders has been proposed for some time, for example, as part of RM-ODP and IEEE 1471, and seems now-a-days to gain acceptance in practice. However, much work remains to be done on the relationships between different viewpoints. Failing to relate viewpoints may lead to a collection of viewpoint models that is inconsistent and may, therefore, lead to an incorrect implementation. This paper defines an approach that helps designers to relate different viewpoints to each other. Thereby, it helps to enforce the consistency of the overall design. The results of this paper are expected to be particularly interesting for Model Driven Architecture (MDA) projects, since the proposed approach can be used for the explicit definition of the models and relationships between models in an MDA trajectory.


Annotations: The use of concerns in IEEE 1471 derives from the phrase separation of concerns in software engineering. The earliest use of this phrase appears to be in this 1974 paper by Dijkstra: “Let me try to explain to you, what to my taste is characteristic for all intelligent thinking. It is, that one is willing to study in depth an aspect of one’s subject matter in isolation for the sake of its own consistency, all the time knowing that one is occupying oneself only with one of the aspects. We know that a program must be correct and we can study it from that viewpoint only; we also know that it should be efficient and we can study its efficiency on another day, so to speak. In another mood we may ask ourselves whether, and if so: why, the program is desirable. But nothing is gained—on the contrary!—by tackling these various aspects simultaneously. It is what I sometimes have called “the separation of concerns”, which, even if not perfectly possible, is yet the only available technique for effective ordering of one’s thoughts, that I know of. This is what I mean by ‘focussing one’s attention upon some aspect’: it does not mean ignoring the other aspects, it is just doing justice to the fact that from this aspect’s point of view, the other is irrelevant. It is being one- and multiple-track minded simultaneously.”


Annotations: Describes ArchiMate’s approach to the definition and presentation of enterprise architecture viewpoints, a classification of viewpoints; based upon the IEEE 1471 frame of reference.


Annotations: Defines a process for software architects, using the IEEE 1471 model as a foundation. Provides a viewpoint template and viewpoint catalog including: Requirements, Functional,
Deployment, Validation, Application, Infrastructure, Systems Management, Availability, Performance, Security; and the work products (model kinds) used in each.


Annotations: First account of the goals and requirements for IEEE 1471.


Abstract: This paper describes experiences with several architectural frameworks. An “architectural framework” specifies what is included in the description of an architecture, independent of the specific system being described. The three frameworks are the U.S. DoD C4ISR Architecture Framework, the associated Core Architecture Data Model and the emerging IEEE Recommended Practice on Architecture Description. From these experiences, we speculate on the further evolution of architecture frameworks and architectural descriptions.


Annotations: One of the architectural methods motivating the development of IEEE 1471.


Annotations: Overview of the joint IEEE and ISO revision.

Rik Farenhorst and Remco C. de Boer. Architectural knowledge management: supporting architects and auditors. VU University, 2009.

Annotations: Two dissertations on architectural knowledge, built on the IEEE 1471 ontology. Yields useful insights into architectural decisions incorporated into ISO/IEC 42010 revision.


Annotations: First introduction of stakeholder into management thinking.


Annotations: One of the sources motivating the introduction of the notion of stakeholder into IEEE 1471.


Annotations: Defines fourteen architecture viewpoints for use with UML.


Abstract: Separation of concerns represents an important principle for managing complexity in the design and architecture of large component-based software systems. The fundamental approach is to develop local solutions for individual concerns first, and combine them later into an overall solution for the complete system. However, comprehensive support for the integration of interdependent, possibly conflicting concerns related to synchronization behavior is still missing. In our work, we propose a sound solution for this complex type of composition, employing well-known UML description techniques as well as a rigorous formal model of component synchronization behavior. Based on this foundation, we describe a constructive synthesis algorithm which reliably detects conflicting concerns or generates a maximal synchronization behavior for software components with multiple interactions. An optimized implementation of the algorithm has been integrated into a CASE tool to illustrate feasibility and scalability of the presented technique to the example of a moderately large case study.


Annotations: Suggests constructs distinct from process modeling toward the definition of a “business” or “commerce” viewpoint.

J. Gordijn, H. de Bruin, and J.M. Akkermans, “Scenario Methods for Viewpoint Integration in e-Business Requirements Engineering”. In: Proceedings of the 34th Hawaii International Confer-
Annotations: Multiple viewpoint modeling for commerce-related architectural concerns.


Annotations: Surveys 23 architecture frameworks and proposes 9 dimensions for classifying frameworks: Type of information, Scope, Detail level, Stakeholder, Transformation, Quality attribute, Meta level, Nature and Representation.


Annotations: Introduces three viewpoints: Decision, Degree and Data for framing service automation concerns in architecting service-based applications.


Abstract: Software architecture is the result of a set of architecture decisions. Unfortunately, there is currently no commonly accepted approach to architecture decision modeling. Existing approaches do not satisfy all stakeholder concerns in decision description; they do not optimally support the architecting process, and they do not integrate well with the rest of the architecture documentation, which is usually arranged in multiple architectural views. This dissertation reports on multiple empirical studies conducted to understand better the decision making process in practice. The core contribution is a framework for architecture decisions, following the conventions of the international architecture description standard ISO/IEC/IEEE 42010. The framework consists of five interrelated viewpoints, each of which being dedicated to satisfying different stakeholder concerns in architecture decisions. The viewpoints of the framework can be used individually, or in combination, to describe the architecture decisions made in a software project. To find out if decision viewpoints can support designers in making rational decisions, we conducted a comparative multiple-case study with four groups of senior software engineering students. The results confirm
that students who create decision views according to the viewpoint definition explore and evaluate candidate architectural solutions more systematically than student groups who do not use the decision framework. Finally, this dissertation reports on a lightweight decision-centric architecture evaluation method, which uses viewpoints from the decision framework. The method uncovers and evaluates the rationale behind the most important architecture decisions made in a software project, considering all relevant forces that must be addressed by the decisions.


Abstract: In this paper, we introduce a documentation framework for architecture decisions. This framework consists of four viewpoint definitions using the conventions of ISO/IEC/IEEE 42010, the new international standard for the description of system and software architectures. The four viewpoints, a Decision Detail viewpoint, a Decision Relationship viewpoint, a Decision Chronology viewpoint, and a Decision Stakeholder Involvement viewpoint satisfy several stakeholder concerns related to architecture decision management.

With the exception of the Decision Stakeholder Involvement viewpoint, the framework was evaluated in an industrial case study. The results are promising, as they show that decision views can be created with reasonable effort while satisfying many of the stakeholder concerns in decision documentation.


Annotations: Discussion of some open issues with respect to the use of IEEE 1471, after its standardization.


Annotations: Discusses impact of adopting IEEE 1471 on The Open Group’s Architecture Framework (TOGAF).


Abstract: This paper sketches an approach to using aspects for architectural description within the conceptual framework of IEEE 1471. I propose a definition of architectural aspect within that framework and examine its consequences and motivations. I show that architectural aspects can be accommodated within the current conceptual framework of IEEE 1471 without modification; and outline extensions to the framework which could be candidates for further standardization work, or incorporated into aspect-oriented architectural methods.


Annotations: The C4ISR Architecture Framework was a proposed approach to documenting architectures for the DoD. This memo provides detailed comments on version 1.0 of the framework. Subsequent versions of the framework are now known as the DoD Architecture Framework (DoDAF). The latest version has not fixed the problems cited in this memo.


Annotations: An early attempt to apply some of the ideas of IEEE 1471 to systems engineering.


Annotations: Describes tools to support definition of architecture frameworks and their viewpoints based on 42010 model.


Annotations: One of the architecture methods motivating IEEE 1471’s approach.


Annotations: Builds upon the IEEE 1471 ontology to develop a set of 158 guidelines for improving the readability of IT architectures. Proposes a method to define IEEE 1471 viewpoints. Also surveys 23 architecture frameworks and presents 9 base dimensions that structure architecture descriptions: Type of information, Scope, Detail level, Stakeholder, Transformation, Quality attribute, Meta level, Nature and Representation.


Annotations: Key inspiration for the IEEE 1471 conceptual model and its documentation as a UML class diagram.


Annotations: Leading example of a multiple view-based software architectural method, and a motivating case for IEEE 1471.


Philippe Kruchten, Rafael Capilla, and Juan Carlos Dueñas. “The Decision View’s Role in Software Architecture Practice”. In: IEEE Software 26.2 (March–April 2009), pp. 36–42. DOI: 10.1109/MS.2009.52.

Annotations: Traces the historical evolution of thinking about software architecture representation and advocates a decision viewpoint cross-cutting other architectural views.


Annotations: Gartner builds on the IEEE 1471 definition of architecture to its relevance to Enterprise Architecture.


Annotations: Overview of Gartner’s Enterprise Architecture Framework in which they "adopted an aspect-oriented approach to our framework, deliberately compatible with IEEE 1471... [defining] three interdependent viewpoints: a business viewpoint, which is concerned with the processes and organization of the business; an information viewpoint, which is concerned with the information that runs the enterprise; and a technology viewpoint, which is concerned with the hardware and software components that support the enterprise. The aspect-oriented approach allows for the articulation of additional viewpoints, should the organization require them.”.


Annotations: A technical overview of IEEE 1471 and discussion of its applicability to systems architecture.


Annotations: Overview of IEEE 1471 after its publication.


Abstract: The use of parallelism enhances the performance of a software system. Its excessive use, however, can degrade the performance. In this paper we propose a parallelism viewpoint to optimize the use of parallelism by eliminating unnecessarily used threads in legacy systems. The viewpoint describes the parallelism behaviour of the system, which can be used to analyze for overheads associated with threads. We illustrate the proposed viewpoint with the help of an industrial case, a parallelism-intensive electron microscope software system. We use the viewpoint to analyze threads suitable to be replaced with a small sized thread pool in this system. Results show that the viewpoint provides a profound insight into the thread-model of the system that is required to reduce the parallelism. In the thread pool analysis, we found that more than 50% threads are underused. They were replaceable with a pool of approximately 11% of these threads.


Annotations: Shows how relational calculus can be very powerful means for cross-view analysis.


Annotations: “SysML has extended the concept of view and viewpoint from UML to be consistent with the IEEE 1471 standard. In particular, a viewpoint is a specification of rules for constructing a view to address a set of stakeholder concerns, and the view is intended to represent the system from this viewpoint. This enables stakeholders to specify aspects of the system model that are important to them from their viewpoint, and then represent those aspects of the system in a specific view. Typical examples may include an operational, manufacturing, or security view/viewpoint.”


Abstract: The conceptual modelling of software architectures is of central importance for the quality of a software system. A rich modelling language is required to integrate the different aspects of architecture modelling, such as architectural styles, structural and behavioural modelling, into a coherent framework. We propose an ontological approach for architectural style modelling based on description logic as an abstract, meta-level modelling instrument. Architectural styles are often neglected in software architectures. We introduce a framework for style definition and style combination. The link between quality requirements and conceptual modelling of architectural styles is investigated. The application of the ontological framework in the form of an integration into existing architectural description notations such as ACME and UML-based approaches, and also service ontologies is illustrated.


Annotations: Published version of their underground classic “Software Architectures”. Early motivation for use of multiple views in architecture description.


Annotations: Surveys the principles of architectural modeling from three angles: Modeling, Utility and Communication and works through case studies of two viewpoint frameworks: Kruchten’s 4+1 and RM–ODP, using consideration of concerns adapted from IEEE 1471.


Annotations: Earliest reference to first-class viewpoints in software engineering literature.


Annotations: Uses IEEE 1471 as part of its foundations.


Annotations: Adopts IEEE 1471 as a starting point. Defines a number of viewpoints and perspectives (cross-cutting viewpoints).


Annotations: Adopts IEEE 1471 as a starting point. Defines a number of viewpoints and perspectives (cross-cutting viewpoints).


Annotations: Extends the IEEE 1471 conceptual framework to Enterprise Architecture.


Annotations: Follow-on to Zachman, “A framework for information systems architecture”, and key paper for enterprise architecture frameworks.


Annotations: This paper describes an explicit viewpoint/style for recovery concern.


Annotations: This paper discusses the decomposition of an architecture based on the recovery style as well as the automatic generation of the code based on the selected architectural decomposition.


Abstract: View/Viewpoint approaches like IEEE 1471-2000, or Kruchten’s 4+1-view model are used to structure software architectures at a high level of granularity. While research has focused on architectural languages and with consistency between multiple views, practical questions such as the structuring at a lower level of detail have not been dealt with. This paper aims at filling this gap by reporting personal experiences from a very large scale industrial domain modeling project. There, structuring the logical view turned out to be a critical success factor. We explain the project and its setting, analyze the role and repercussions of model structuring, and examine the implications model structuring decisions have on other parts of the project. We then explain the model structure abstracted from a very large scale industrial modeling project. Finally, we discuss lessons learned.


Annotations: Building on the definition of concern in IEEE 1471, the authors argue concerns must be first-class entities and concern modeling must be an explicit part of Aspect-Oriented Software Development.

Damien A. Tamburri, Patricia Lago, and Henry Muccini. “Leveraging Software Architectures through the ISO/IEC 42010 Standard: A Feasibility Study”. In: *Trends in Enterprise Architec-
Abstract: The state of the practice in enterprise and software architecture learnt that relevant architectural aspects should be illustrated in multiple views, targeting the various concerns of different stakeholders. This has been expressed a.o. in the ISO/IEC 42010 Standard on architecture descriptions. In the same vein, the research community observed that Architecture Description Languages, or ADLs, should be developed to address stakeholders concerns concentrating on the use of viewpoints for their description. This notwithstanding, we notice today a proliferation of ADLs impervious to these guidelines. This imperviousness creates a gap between what the IT industry requires and what ADLs can provide. This gap makes it impossible for practitioners to choose and use the best-fit ADL for his/her requirements. To fill this gap, we must analyze the existing ADLs, and mine and make explicit their addressed concerns, views, viewpoints, and stakeholders. Such an explicit overview can provide practitioners with pragmatic information for selecting the most suitable ADL, and hence support them in the architecting process. This paper reports on initial results in this direction. Given a specific ADL (namely, DARWIN/FSP), it presents a feasibility study on the methodology mapping the concepts of the ISO/IEC 42010 on the DARWIN/FSP ADL.


Annotations: The Open Group’s enterprise architecture framework.


Abstract: An algebraic approach to the view consistency problem in software development is provided. A view is formalised as a sentence of a viewpoint language; a viewpoint is given by a language and its semantics. Views in possibly different viewpoints are compared over a common view for consistency by a heterogeneous pull-back construction. This general notion of view consistency is illustrated by several examples from viewpoints used in object-oriented software development.


Annotations: Paper motivates introduction of concept of architectural perspectives, in contrast to IEEE 1471-style viewpoints.


Annotations: Proposes an approach to consistency between views using correspondence rules.


Annotations: A key paper underlying much work on enterprise architecture, and establishing an initial foundation for same.