IEEE Standards

IEEE Std 1175.1™-2002
(Revision of
IEEE Std 1175.1-1991)

1175.1™

IEEE Guide for CASE Tool Interconnections—Classification and Description

IEEE Computer Society

Sponsored by the
Software Engineering Standards Committee

IEEE

Published by
The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

11 February 2003

Print: SH95059
PDF: SS95059

IEEE Std 1175.1™-2002 (R2007)
(Revision of
IEEE Std 1175™-1991)

IEEE Guide for CASE Tool Interconnections—Classification and Description

Sponsor

Software Engineering Standards Committee
of the
IEEE Computer Society

Reaffirmed 26 September 2007
Approved 11 November 2002

IEEE-SA Standards Board

Abstract: This guide describes the scope of application and interrelationships for the members of the IEEE 1175 family of standards. It points the reader to the appropriate member standard that addresses issues involved in effectively integrating computing system tools into a productive engineering environment.
Keywords: CASE tool integration, CASE tool interconnection, CASE tool interfaces, CASE tool organization issues, CASE tool platform issues, CASE tool user issues

---

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Print: ISBN 0-7381-3483-X SH95059
PDF: ISBN 0-7381-3484-8 SS95059

Introduction

(This introduction to the 1175 family of IEEE standards is not part of IEEE Std 1175.1-2002, IEEE Guide for CASE Tool Interconnections—Classification and Description.)

NOTE—At the time of the final approval of this standard and the other members of this family of IEEE standards, references to "IEEE P1175.x" will be changed to read "IEEE Std 1175.x" etc. The reader of this document should make such an interpretation for any "P1175.x" text entries denoting members of this family that have already been approved by IEEE.

This guide is a member of the 1175 family of IEEE standards. This family of standards replaces IEEE Std 1175-1991, IEEE Standard Reference Model for Computing System Tool Interconnections.[a] IEEE Std 1175-1991 was advanced to a full-use standard in 1994. It covered a number of closely related subjects, and the scope of material contained was able to serve a number of divergent interests. This revision restructures and substantially augments the material in IEEE Std 1175-1991. It has been divided into several individually useful guides, recommended practices, and standards in order to facilitate its use by different communities of interest. The members of this family are:

Standard number	Title
IEEE Std 1175.1-2002	IEEE Guide for CASE Tool Interconnections—Classification and Description
IEEE P1175.2™	Draft Recommended Practice for CASE Tool Interconnections—Characterization of Interconnections
IEEE P1175.3™	Draft Standard for CASE Tool Interconnections—Reference Model for Specifying Software Behavior
IEEE P1175.4™	Draft Standard for CASE Tool Interconnections—Reference Model for Specifying System Behavior
IEEE P1175.5™	Draft Standard for CASE Tool Interconnections—Syntax for Transferring Behavior Specifications

NOTE—The products listed in this table are trademarks of the Institute of Electrical and Electronics Engineers, Incorporated.

These guides, recommended practices, and standards generally address issues involved in characterizing the kinds of interconnections that exist between a computing system tool and its environment. Four kinds of interconnections with a computing system tool are addressed: interconnections with organizations, users, platforms, and other computing system tools.

This guide is an expansion of Part 1 of IEEE Std 1175-1991. It provides an integrated overview of the remaining members of the 1175 family of standards, including the fundamental concepts that provide a basis for organizing the material.

[a]Although approved in 1991, IEEE Std 1175-1991 was actually published in 1992, and is sometimes found referenced as IEEE Std 1175-1992. It appears in the standards listing on the IEEE Xplore Web site (http://ieeexplore.ieee.org/) as "IEEE Std 1175, 17 Aug. 1992," with the title "IEEE trial-use standard reference model for computing system tool interconnections." In 1994 the term "trial-use" was removed from the title when the standard was approved for full-use status. The 1994 version, which was identical to the 1992 publication except for the title and minor editorial corrections, is not available on the IEEE Web site.

Participants

At the time this standard was completed, the working group had the following membership:

Carl A. Singer, *Chair*

Peter L. Eirich
Álvaro Francisco de Castro Medieros
Paul C. Jorgensen
Dwayne L. Knirk
Robert M. Poston
Robert Wessely

The following members of the balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Bakul Banerjee
Mitchell Bonnett
Juris Borzovs
Edward Byrne
Antonio M. Cicu
Paul Croll
Guru Dutt Dhingra
Geoffrey Darnton
David Dikel
Sourav Dutta
John Fendrich
Yaacov Fenster
Ronald Hofer
Vladimir Ivanov
Dwayne L. Knirk
Susumu Ohno
Lou Pinto
Alexander Polack
James Ruggieri
Carl A. Singer
Richard Thayer
Booker Thomas
Oren Yuen
Janusz Zalewski

When the IEEE-SA Standards Board approved this standard on 11 November 2002, it had the following membership:

James T. Carlo, *Chair*
James H. Gurney, *Vice Chair*
Judith Gorman, *Secretary*

Sid Bennett
H. Stephen Berger
Clyde R. Camp
Richard DeBlasio
Harold E. Epstein
Julian Forster*
Howard M. Frazier
Toshio Fukuda
Arnold M. Greenspan
Raymond Hapeman
Donald M. Heirman
Richard H. Hulett
Lowell G. Johnson
Joseph L. Koepfinger*
Peter H. Lips
Nader Mehravari
Daleep C. Mohla
William J. Moylan
Malcolm V. Thaden
Geoffrey O. Thompson
Howard L. Wolfman
Don Wright

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Alan Cookson, *NIST Representative*
Satish K. Aggarwal, *NRC Representative*

Catherine Berger
IEEE Standards Project Editor

Contents

IEEE Guide for CASE Tool Interconnections—Classification and Description

1. Overview

1.1 Scope

This guide describes the scope of application and interrelationships for the members of the IEEE 1175 family of standards, and it points the reader to the appropriate standards for clarifying issues involved in effectively integrating computing system tools into a productive engineering environment. The other members of this family are:

— IEEE P1175.2™, Draft Recommended Practice for CASE Tool Interconnections—Characterization of Interconnections
— IEEE P1175.3™, Draft Standard for CASE Tool Interconnections—Reference Model for Specifying Software Behavior
— IEEE P1175.4™, Draft Standard for CASE Tool Interconnections—Reference Model for Specifying System Behavior
— IEEE P1175.5™, Draft Standard for CASE Tool Interconnections—Syntax for Transferring Behavior Specifications

1.2 Purpose

To assist the users in the effective application of the IEEE 1175 family of standards, this guide provides an overview of the remaining members of this family. It describes the organization of this family, the focus of the individual members, and the logical interrelationships among the members. By addressing the fundamental concepts of computing system tool interconnections, this guide also establishes a framework for applying the recommended practice and the standards in this family.

The discussion of interconnections in this family actually has wider applicability to computing system tools in general, beyond only CASE tools. Most computing system tools have interconnections with organizations, users, platforms, and other tools, so consideration of these interconnections is important to them. Also, while most computing system tools do not need to communicate behavior descriptions of subject systems, their creators need to develop such descriptions for them.

In these standards, the word interconnection has an abstract connotation: it includes all ways in which successful operation of a computing system tool depends on its environment. Thus, an interconnection is an association between a computing system tool and something in the environment. This family of standards recognizes two forms of interconnections. A passive interconnection is an interoperability agreement. An active interconnection is an interaction mechanism. These standards address the characteristics of interconnections between computing system tools and their organizational context, their user context, their platform context, and the context provided by other tools. Interconnections between computing system tools are the principal focus of these standards.

1.3 Audience

This guide is intended to help buyers, builders, testers, and users of professional tools to use this family of standards more effectively, and to help managers responsible for the implementation of computing system tools to reach sound decisions on an implementation strategy.

The IEEE 1175 family of standards may be used for various purposes in software and system engineering.

Process managers, designers, and implementers in supplier organizations. These standards provide an organizational, managerial, and technical framework for evaluating, adopting, and integrating commercial and custom computing system tools, and for extending automated support of development processes.

Computing system tool producers. These standards define a general interface for transferring system behavior specifications between toolset elements in an integrated development environment, to include GUI user interfaces, repositories, analysis tools, evaluation tools, visualization tools, and documentation tools.

System and software analysts, architects, and designers. These standards define a reference model for capturing behavior specifications and architectural structures of systems in machine-processible formats to which tools may be applied for analysis, design, development, verification, and validation.

Since its original publication in IEEE Std 1175™-1991 [B2],[1] much of the material in this family of standards has been used in teaching software engineering, designing specification tools and databases, integrating commercial CASE toolsets, developing industrial data management and control systems, developing and testing safety-related software, and designing a logical data model for Simulation Based Acquisition environments intended for prototyping military systems. It has also been used as the language specification for the input to a commercial test design tool. This original material has been edited and updated to prepare the first three documents in this family: this guide, IEEE P1175.2, and IEEE P1175.3. In addition, new material with a similar purpose has been developed for the final two documents in this family, IEEE P1175.4 and IEEE P1175.5, in order to provide similar benefits within a broader scope of application.

1.4 Organization of this guide

This guide contains four clauses and one annex. The following annotations provide a road map to the contents.

Clause 1: Overview—This clause describes the purpose and scope of this guide to the members of the 1175 family of standards.

Clause 2: References—This clause lists the references necessary for implementing this standard.

Clause 3: Definitions—This clause provides definitions of terms.

[1]The numbers in brackets correspond to those of the bibliography in Annex A.

Clause 4: Fundamental tool concepts employed in the IEEE 1175 family of standards—This clause describes several fundamental concepts underlying the content of this family of standards and provides some basic terminology used to describe the application and use of computing system tools.

Clause 5: Description of the IEEE 1175 family of standards—This clause describes the nature of the revisions made to the initial publication of IEEE Std 1175-1991 in order to develop this family of standards, and provides an overview for each of the recommended practices and standards in the family.

Annex A: Bibliography—This annex provides references for other standards whose potential usage is identified within this guide.

2. References

This standard shall be used in conjunction with the following publication. If the following publication is superseded by an approved revision, the revision shall apply. The following glossary standard shall be used, when applicable, for software behavior concept terms not defined in this guide.

IEEE Std 610.12™-1990, IEEE Standard Glossary of Software Engineering Terminology.[2,3]

3. Definitions

The following definitions describe specific terms as used within the context of the standard recommended practices. Additional relevant terms defined in IEEE Std 610.12-1990 are as follows: CASE, interface, semantics, software tool, and syntax.

3.1 behavior: Observable activity of a system, measurable in terms of quantifiable effects on the *environment* whether arising from internal or external stimulus; also, the peculiar reaction of a thing under given circumstances.

3.2 behavior specification: A structured collection of data that describes the potential variety of *behavior* possible from a system.

3.3 CASE tool: A software tool used for computer-aided software engineering (CASE).

3.4 computing system specification concepts: Visible and quantifiable abstractions of computing system characteristics having attributes in isolation and relationships in context.

3.5 computing system tool: A computer-based tool used by a developer or maintainer *organization* for creating and evolving dynamic systems; computing system tools include not only traditional CASE tools, but also, for example, requirements tools, verification and validation tools, design tools, and documentation tools.

3.6 environment: Anything affecting a *subject system* or affected by a *subject system* through interactions with it, or anything sharing an interpretation of interactions with a *subject system*.

3.7 interconnection: An association between a computing system tool and something in the *environment* that affects both endpoints, though not necessarily in the same way.

3.8 meta-model: A logical information model that specifies the modeling elements used within another (or the same) modeling notation.

[2]The IEEE standard referred to in Clause 2 is a trademark owned by the Institute of Electrical and Electronics Engineers, Incorporated.

[3]IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA (http://standards.ieee.org/).

3.9 ontology: A logical structure of the terms used to describe a domain of knowledge, including both the definitions of the applicable terms and their relationships.

3.10 subject system: A computing system (existing or to be created) about which descriptive information is being developed in a computing system tool.

3.11 subject tool: A particular computing system tool that is the focus for a description of interconnections and tool content.

4. Fundamental tool concepts employed in the IEEE 1175 family of standards

Everything presented in this family of standards with respect to a computing system tool is from a fixed point of view relative to the *subject tool*. The *subject tool* is a particular tool, singled out from the computing system tool population, whose interconnections are to be described. Four contexts are the focus for interconnections of the *subject tool,* as shown in Figure 1.

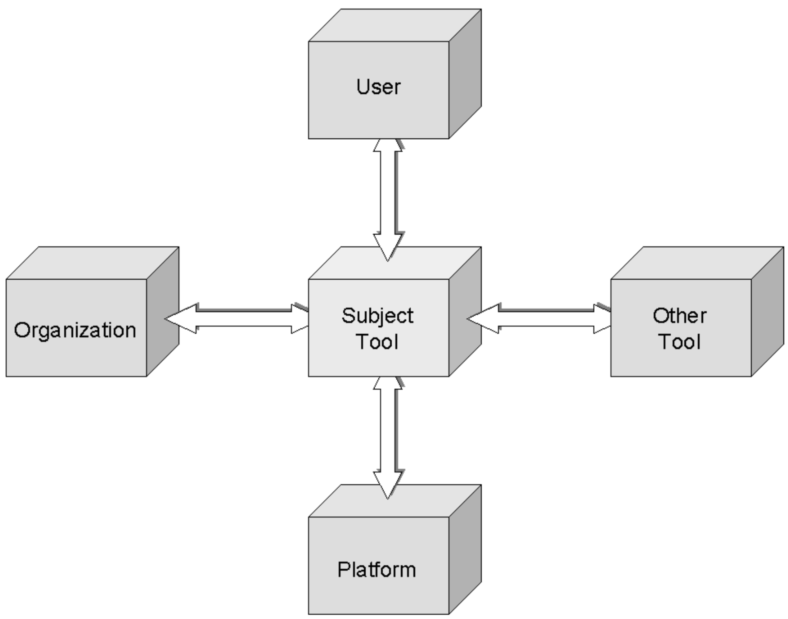

Figure 1—Context for tool usage and information exchange

These contexts are considered separately.

Organization. One context is the *organization* and its processes in which the *subject tool* is used. The issues of interconnection with the *organization* center on how the *subject tool* supports the activities and work products in the *organization's* engineering processes.

User. Another context is the user who interacts with the *subject tool*. The issues of interconnection with the User center on data and control interactions at the computer-human interface. A special subset of these issues surrounds the representation and interpretation of data about some external *subject system*. This is a distinguishing feature of computing system tool interconnection with *users*.

Platform. Another context is the hardware and software *platform* (infrastructure) on which the *subject tool* is hosted. The issues of interconnection with the *platform* center on the data and control interactions by which the *subject tool* acquires services and resources such as communications, process control, data storage, and input and output channels. None of these interactions involve interpretation of data with respect to any *subject system*.

Other tool. The final context is any other computing system tool with which the *subject tool* must exchange data. The issues of interconnection with *other tool* center on data control, data format, data content, and shared semantic interpretation.

When the *user* in the *organization* employs the *subject tool* on the *platform*, it is for the purpose of acquiring, developing, or maintaining information about a particular system. That particular referent system is the subject of considerations, descriptions, queries, and work. Many tools can manipulate data (record, store, search, collect, emit), but they do it with little interpretation beyond values and structures.

The computing system tools to which this family of standards applies not only can manipulate data but also can interpret data, finding meaning in the data with respect to some external *subject system*. See Figure 2.

A *user* perceives or conceives the behavioral characteristics of a computing system. The *user* represents those perceptions or conceptions as data that can be communicated to other humans or to computing system tools. Computing system tools differ from one another in terms of the form of communication they have with *users* and the extent to which they make inferences based on the user data about the *subject system*.

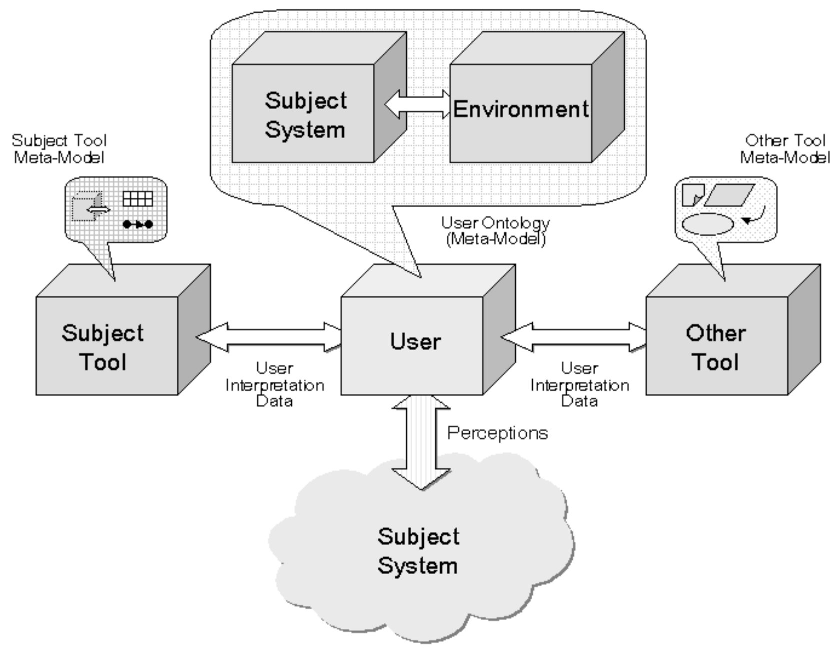

Figure 2—User interpretation of tool data

Each computing system tool has its own meta-model of the data it exchanges with it *users*. It is only to the extent that a tool's meta-model corresponds with the user's (human) ontology that the tool is able to make useful inferences about the *subject system* based on data reflecting the user's interpretation of the system. Figure 2 illustrates how the *user* interprets *behavior* according to an ontology of interactions. Each *tool* is shown with its own particular meta-model of the user's interpretation data. Because each tool meta-model is unique, only some of the user understanding is usable to it, and only some of its interpretations may be useful to the *user*.

Some computing system tools accept user interpretation data in a graphical form and limit themselves to interpreting the data as drawings. They make almost no inferences about the *subject system*. Other computing system tools provide varying degrees of interpretation of user data as facts about a referent software system.

Some computing system tools offer a limited interpretation of *user* data. They extend their internal concepts of data value and structure to include graphical constructs used in various modeling methods. For example, they identify some data with graphical elements and some data with graphical element relationships. They interpret these data with reference to rules of particular graphical models such as allowed and disallowed connectivities, compositions, and correspondences.

Other computing system tools offer extensive interpretation according to sophisticated models of software behavior and structure. Examples of this are reverse engineering tools that uncover the design and architecture basis for code and domain frameworks that combine an application specification with a standard architecture to produce code.

5. Description of the IEEE 1175 family of standards

5.1 IEEE Std 1175.1-2002 classification and description

5.1.1 Overview

IEEE Std 1175.1-2002 is a guide to the IEEE 1175 family of standards. It describes how these standards are intended to be used to accomplish the effective integration of computing system tools into a productive engineering environment and sets forth the fundamental concepts on which these standards are based. These concepts establish the integrating framework for the other members of this family of standards. IEEE Std 1175.1-2002 describes the scope of application of each member standard, the various issues addressed in that standard, and the interrelationships between the family members.

5.1.2 Relationship to IEEE Std 1175-1991

This guide is an expansion of Part 1 of IEEE Std 1175-1991. The expanded contents include a discussion of the new material in this family of standards, as well as a summary of the concepts that unify the members of this family.

5.2 IEEE P1175.2 characterization of interconnections

5.2.1 Overview

IEEE P1175.2 presents four perspectives on a computing system tool's interconnections that offer insight into the operational problems of interconnecting computing system tools with their environment. IEEE P1175.2 establishes recommended collections of standard contextual attributes describing relationships between a computing system tool and its organizational deployment, its human user, its executable *platform*, and its peer tools. These contextual attributes are of the "news-story" form: who, what, when, where, why. The values of these contextual attributes are references to organizational, industrial, and professional standards. By assisting *users* to reach a clear understanding of the context of operation for a computing system tool, IEEE P1175.2 contributes to the effective implementation and application of computing system tools.

5.2.2 Relationship to IEEE Std 1175-1991

This draft recommended practice is an expansion of Part 2 of IEEE Std 1175-1991. It has been updated to reflect new technologies, practices, and standards that have emerged in recent years and has been augmented with additional material for various issues that have taken on increased importance since 1992. One significant addition is a *tool–user* interconnection profile, added in response to the increased current emphasis on user interface issues. The *tool–user* profile also responds to the development of new software techniques for interface programming in conjunction with the emergence of the World Wide Web. Updates have been made throughout the document in order to clarify the interconnection framework.

5.2.3 Design principles

IEEE P1175.2 is a recommended practice that describes four relationship contexts that must exist around a computing system tool for its effective use: *tool–organization, tool–user, tool–platform,* and *tool–tool*. This focus is illustrated in Figure 1. IEEE P1175.2 identifies the significant factors affecting the successful interconnection of the tool within those contexts and assigns attributes for each context that help to characterize the nature of the computing system tool's relationships. The attributes in each context are the major factors affecting successful interconnection of the tool with that context. These are multi-dimensional attributes whose values may be standards or practices that are project-specific, organization-specific, professional, national, and/or international in nature.

5.2.4 Issues addressed

Standardizing interfaces among *users*, *tools,* and *platforms* enables transfers of data between them, but properly communicating the meaning of the data requires compatibility between the sender's and receiver's conceptual models as well, and also requires at least some contextual similarity. Each contextual interface has different compatibility requirements that are influenced in part by the mechanisms of interconnection and in part by contents of the interactions.

Tool–organization. Data exchanges between the *subject tool* and the *organization* are interpreted with respect to the *organization's* software process.

Tool–user. Data exchanges between the *subject tool* and the *user* are interpreted with respect to the Tool's internal concepts of computing system descriptions. These concepts form a fixed modeling viewpoint into which the *user* maps his understanding of the *subject system*.

Tool–platform. Data exchanges between the *subject tool* and the *platform* are interpreted with respect to storage and communication mechanisms: types, values, and internal associations. There is no interpretation of information about the *subject system*.

Tool–tool. Data exchanges between the *subject tool* and another tool or user are dependent not only upon a common syntax and semantics, but also upon the availability of compatible transfer mechanisms and transfer control processes, along with compatibility of the purposes (or intended uses) for the information being transferred.

5.3 IEEE P1175.3 reference model for specifying software behavior

5.3.1 Overview

A unique feature of computing system tools is that the purpose of their primary interactions, both with *users* and with *other tools,* is to describe other computing systems. IEEE P1175.3 focuses specifically on a common set of modeling concepts found in commercial CASE tools for describing the operational behavior of a software product, and provides a formal, logical model for describing this behavior. IEEE P1175.3 also defines a Semantic Transfer Language (STL) for communicating software behavior descriptions from one tool to another. A notable feature of the STL is its design for human readability, which makes STL text files suitable for use in software design reviews by *users* unfamiliar with computing system tool diagramming notations. In addition, the design of the STL syntax readily permits analysts to prepare and edit STL descriptions using a text editor or word processor.

5.3.2 Relationship to IEEE Std 1175-1991

This draft standard is a revision of Part 3 of IEEE Std 1175-1991. The original material has been restructured to be consistent with the logical presentation approach used for structuring IEEE P1175.4 and IEEE

P1175.5, namely: the concept definition material is presented first, followed by the presentation of the corresponding formal syntax. This has been done to enhance understandability. In the case of IEEE P1175.3, this can never be a complete and distinct separation because the normative meta-model in IEEE P1175.3 is presented as structured text corresponding to the transfer syntax. However, there is less interleaving of concepts and syntax in IEEE P1175.3 than in Part 3 of IEEE Std 1175-1991. Because Part 3 of IEEE Std 1175-1991 remains useful within its defined scope and purpose, it is included in this family of standards as IEEE P1175.3. IEEE P1175.3 also incorporates minor editorial corrections in addition to the restructuring as indicated above. To preserve backward compatibility, this revision does not change any of the syntax or the conformance aspects of the STL. The primary emphasis for this revision has been to improve the clarity and usability of the material for its original intended purposes, rather than to revise the content to incorporate the software methodology developments of the 1990s.

5.3.3 Design principles

IEEE P1175.3 identifies a common set of modeling concepts found in commercial computing system tools for describing the operational behavior of a software product. It establishes a uniform, integrated model and a textual syntax for expressing the common properties (attributes and relationships) of those concepts as they have been used to model software behavior.

If two different computing system tools are being used to describe a system, each may maintain its own information meta-model. However, to share information about a *subject system*, at least one of the computing system tools must map its individual meta-model against the meta-model of the other.

As an alternative to establishing direct *tool–tool* mappings as the basis for information sharing among computing system tools, IEEE P1175.3 identifies a common set of “tool-neutral” modeling concepts. By mapping their internal data into the common STL syntax provided by IEEE P1175.3 and exchanging data files constructed using the STL, it becomes possible for two different computing system tools to exchange descriptions of a *subject system* reliably. This is illustrated in Figure 3.

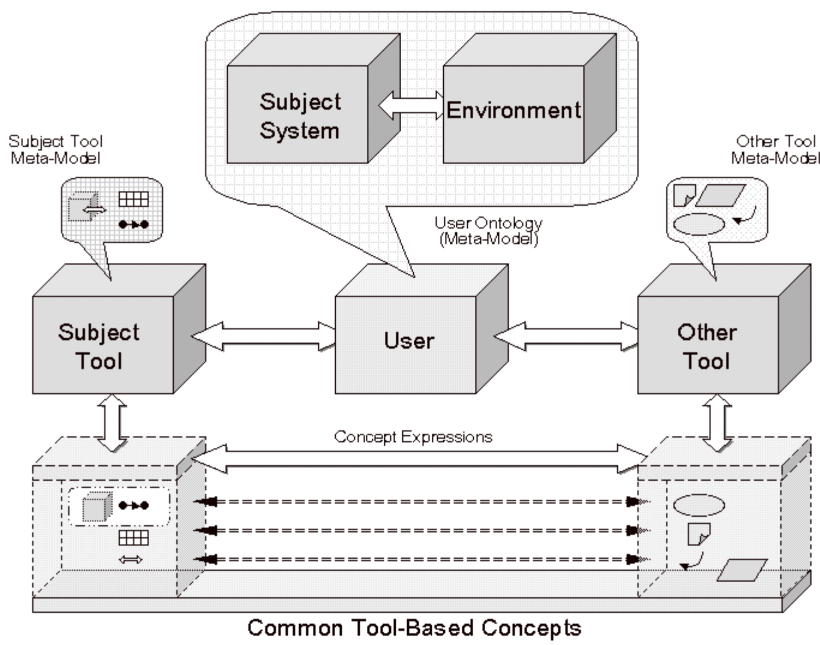

Figure 3—*Tool–tool* mapping of individual tool meta-models

Since each tool can map its own meta-model constructs against the standard IEEE P1175.3 model, it is not necessary for any one tool to understand the internal structures of another tool. This also means that a number of tools can exchange descriptions on the basis of one concept mapping per tool, a substantial reduction of effort from the pair-wise mapping approach.

The exchange of *subject system* descriptions between tools requires a closure constraint on the meta-model mappings. Closure requires that one user's understanding of a *subject system*, encoded into one tool's meta-model, can be transferred through common concept expression mappings into another tool's meta-model, such that a second user receives an essentially equivalent understanding of the *subject system*. "Essentially equivalent" means that both *users* can make similar inferences about the *subject system*. Similarity between each user's inferences is the test for achieving closure in the exchange.

5.3.4 Issues addressed

The syntax provided by IEEE P1175.3 is designed and optimized for the exchange of software system descriptions prepared according to traditional software engineering methodologies, i.e., those typically involving data flow diagrams, entity-relationship diagrams, state-transition diagrams, and real-time diagramming constructs. There is a capability to create rudimentary object descriptions from the STL elements typically used in such diagrams in order to indicate how objects can aggregate descriptive features.

The IEEE P1175.3 exchange syntax is both machine-processible and human-readable. It is suitable for both *tool–tool* data exchange purposes and for user reviews of application behavior descriptions, and it has been successfully applied for requirements reviews by customer and user representatives.

The STL defined in IEEE P1175.3 complements the Unified Modeling Language™ (UML™) [B4], the Extensible Markup Language (XML) [B1], the specification for XML Metadata Interchange (XMI®) format [B5], and the OMG™ Meta-Object Facility (MOF™) [B3]. Those standards and specifications together define how UML models and other MOF-compliant models can be passed from tool to tool, using the format specified by XMI. Those standards and specifications do not apply for models defined using methodologies that are not MOF-compliant. However, the concepts defined in IEEE P1175.3 STL do encompass the meaning of the elements found in a variety of common modeling techniques, and those elements can be mapped to the STL concepts to create a basis for *tool–tool* exchange of models, as Figure 3 illustrates. In this manner, the STL can be used to transfer models that are outside the scope of XMI.

While the IEEE P1175.3 STL provides a basic capability to represent objects, it does not encompass many of the specific features incorporated into UML and cannot replace XMI for its intended uses. In summary, STL and XMI have limited overlap of their respective scopes of application, they serve significant application areas of different kinds, and each applies within its own intended domain of usage.

5.4 IEEE P1175.4 reference model for specifying system behavior

5.4.1 Overview

IEEE P1175.4 encompasses the description of other computing systems supported by IEEE P1175.3, but goes further, providing a basis for representing a wider variety of computing systems. Specifically, IEEE P1175.4 provides the necessary semantic elements for describing general hardware/software systems, including hardware-only, software-only, and/or mixed system components, and allows these different types of components to be treated in a consistent manner.

5.4.2 Relationship to IEEE Std 1175-1991

The meta-model in this draft standard is conceptually expanded over the original meta-model for software behavior in Part 3 of IEEE Std 1175-1991. It encompasses all of the observable behavior concepts provided by the meta-model there and has been further refined in terms of more basic systems fundamentals. The description of interactions is no longer limited to only discrete message-passing forms.

Many inter-tool data transfer standards deal with protocol and syntax of the transfer, assuming a shared semantic basis. Others, such as Part 3 of IEEE Std 1175-1991 (and therefore IEEE P1175.3), provide an explicit meta-model to establish a shared semantic basis. However, the IEEE P1175.3 meta-model is based on popular software development methodologies and must therefore necessarily reflect a compromise among elements from methodologies that were developed independently. IEEE P1175.4 goes beyond IEEE P1175.3 by providing an explicitly defined meta-model for specifying fundamental, mechanistic behaviors for broad classes of systems, of which computing system and software behaviors are special cases. Many of the software concepts originally presented in IEEE P1175.3 have been reduced into simpler, more directly observable qualities of system behavior.

5.4.3 Design principles

IEEE P1175.4 specifies a semantic basis for describing the behavior of a *subject system* in terms of observations that can be made at the boundary of that system. Conventional system engineering model elements are reduced into simpler, directly observable fact statements about system behavior. This allows individual tools, whatever their own internal ontologies, to be as precise as their own internal ontology will allow in communicating an understanding of behavior of a *subject system*. This approach also permits improved traceability from requirements to design, and provides advantages for system testing and evaluation as well.

IEEE P1175.4 includes representations for extended-time interactions, conservation of physical quantities, and a wide range of physical phenomena. Although targeted for computing system descriptions, the treatment is sufficiently general that a variety of computing systems, mechanical systems, and hybrid systems can be represented using the IEEE P1175.4 concepts.

When multiple tools are being used to describe a system, each may maintain its own information meta-model. However, as illustrated in Figure 4, to share information about *subject system*, each tool must map its individual meta-model into the reference model defined in IEEE P1175.4. The reference model in IEEE P1175.4 establishes a meta-model for system behavior to support compatible interpretation by multiple tools. This extends the practices of identifying and managing interconnections, as addressed in IEEE P1175.2, to the standardization of content in *tool–tool* interconnections. The universe of discourse is the specification of behavior for computing systems and applications. The purpose of IEEE P1175.4 is to identify the significant elements, relations, and attributes characterizing the executable behavior of a computing system or application. To the extent that various system and software modeling methods can express these facts, models created within those methods can be unambiguously transferred between computing system tools.

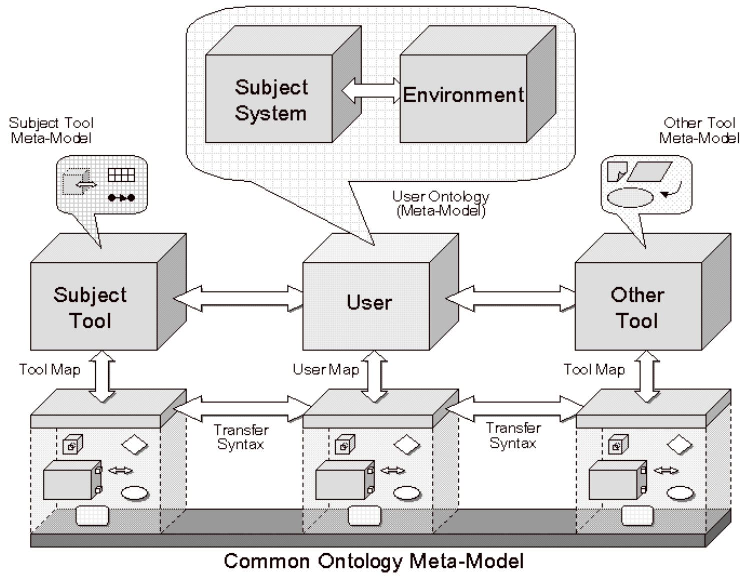

Figure 4—Mapping of tool meta-models against standard reference meta-model

5.4.4 Issues addressed

IEEE P1175.4 establishes a standard meta-model to provide common semantics for interpreting different tool descriptions of a *subject system*. In addition to the general *tool–tool* compatibility items covered in IEEE P1175.2, data exchanges between the *subject tool* and *other tool* or between the *subject tool* and *user* are interpreted with respect to each tool's meta-model of the *subject system*. The amount of information in a transfer of data between tools is proportional to the amount of overlap in the concepts and representations of their meta-models. If the meta-models for a pair of tools are incompatible, there can be no mutual interpretation.

Since it is not generally the case that two tools will have a 100% identical scope and identical semantic content, a reference meta-model can be used to establish where there is, and where there is not, a correspondence. Such a reference meta-model provides the means to precisely answer the question of the degree of commonality for semantics, a question identified within the *tool–tool* reference model in IEEE P1175.2.

In the IEEE P1175.4 reference meta-model, concepts arise from fundamental characteristics, such as the controllability, observability, and measurability of the behavior of the *subject system*. This does not require the communicating tools to share a modeling concept with an inherent restricted application domain, such as the data flow diagramming paradigm. Rather, two tools can interpret their internal concepts in terms of the observable occurrence of an interaction during a time interval, with an identified content. Tools that can do this can extract at least a partial understanding of a *subject system* from a transfer description of a data flow —when expressed in those terms—*even if* the concept of a data flow is not part of the receiving tool's basic vocabulary.

The meta-model defined in IEEE P1175.4 permits a more diverse set of tools (in comparison to IEEE Std 1175-1991) to effectively use 1175 to exchange information. Different types of tools, further apart in the life style of software engineering, will be able to meaningfully interchange data. Also, software engineering tools will be able to exchange data with systems engineering tools, and with tools from other engineering disciplines (e.g., electronics design), to a degree not possible with the original IEEE Std 1175-1991.

5.5 IEEE P1175.5 syntax for transferring behavior specifications

5.5.1 Overview

To help with the integration of tools into productive environments, IEEE P1175.5 defines a set of data elements based on the meta-model in IEEE P1175.4, and a corresponding syntax for transferring the description of a *subject system* from one tool to another. The Extensible Markup Language (XML) is used as the basis for a syntax to provide a lossless, textual, human-readable form of such data for communication, and to facilitate exchanges of data with web-based applications.[4]

5.5.2 Relationship to IEEE Std 1175-1991

This draft standard provides a general syntax for transferring the models and meta-models defined in IEEE P1175.4. IEEE P1175.5 relates to IEEE P1175.4 in the same way that the STL, defined in Part 3 of IEEE Std 1175-1991, relates to the meta-model for software behavior in Part 3 of IEEE Std 1175-1991.

5.5.3 Design principles

IEEE P1175.5 provides a structured syntax and a specific set of data elements that can be exchanged as a text stream or text file with *users* or *other tools*. This syntax permits describing a system's operational behavior as an instance of the reference meta-model in IEEE P1175.4. Figure 5 illustrates the relationship between the IEEE P1175.4 meta-model and the IEEE P1175.5 XML syntax in the context of the *tool–tool* exchange of a computing system description.

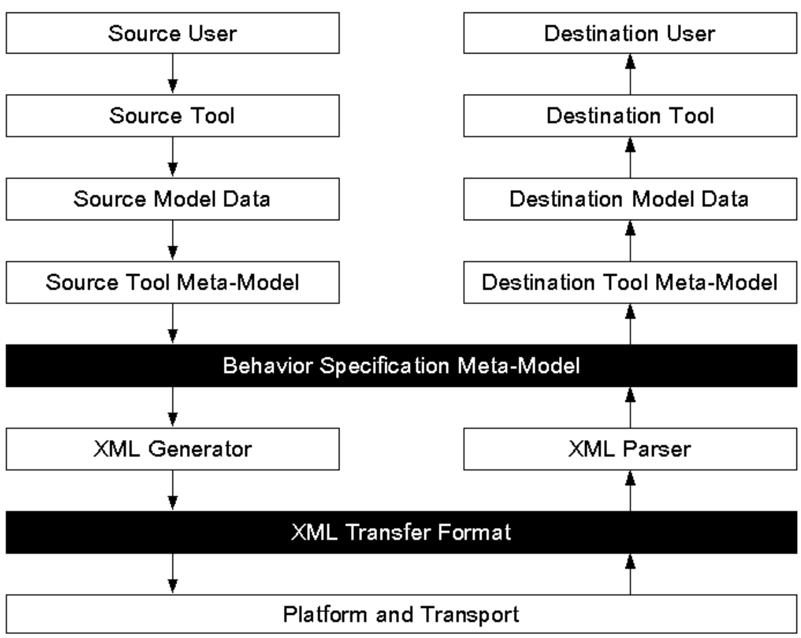

Figure 5—Layered view of behavior specification exchanges

[4]Although this is an "understandable" text file to a reader, it is not "human readable" to the same degree as is the STL in IEEE P1175.3. The STL is essentially a machine parsable form of structured English.

5.5.4 Issues addressed

IEEE P1175.5 addresses the exchange between two tools of behavior specifications for a *subject system*. The purpose of IEEE P1175.5 is to define a standard information transfer. A transfer language has to encode both meta-model and model descriptions, but there is considerable flexibility in how that is done. *Tool–tool* exchanges are interpreted with the standard meta-model provided in IEEE P1175.4. IEEE P1175.5 specifies a syntax for physically transferring behavior specification data from one computing system tool to another, using a sequential text format.

The original 1991 version of IEEE Std 1175 provided a verbose, symmetric syntax for encoding its reference model. Similarly, IEEE P1175.5 provides a verbose, symmetric XML-based syntax for directly encoding the IEEE P1175.4 reference model data.

Annex A

(informative)

Bibliography

[B1] Extensible Markup Language (XML) 1.0, World Wide Web Consortium (W3C®).[5]

[B2] IEEE Std 1175-1991, IEEE Standard Reference Model for Computing System Tool Interconnections (Withdrawn).[6]

[B3] Meta-Object Facility (MOF™), Object Management Group™ (OMG™).[7]

[B4] Unified Modeling Language™ (UML™), Object Management Group (OMG).[8]

[B5] XML Metadata Interchange (XMI®), Object Management Group (OMG).[9]

[5]Available from the World Wide Web Consortium (W3C) Web site, http://www.w3.org/XML/. The most current available version is the intended reference.

[6]Although approved in 1991, IEEE Std 1175-1991 was actually published in 1992, and is sometimes found referenced as IEEE Std 1175-1992. It appears in the standards listing on the IEEE Xplore Web site (http://ieeexplore.ieee.org/) as "IEEE Std 1175, 17 Aug. 1992," with the title "IEEE trial-use standard reference model for computing system tool interconnections." In 1994 the term "trial-use" was removed from the title when the standard was approved for full-use status. The 1994 version, which was identical to the 1992 publication except for the title and minor editorial corrections, is not available on the IEEE Web site.

[7]Available from the Object Management Group (OMG) Web site, http://www.omg.org/. The most current available version is the intended reference.

[8]Available from the Object Management Group (OMG) Web site, http://www.omg.org/. The most current available version is the intended reference.

[9]Available from the Object Management Group (OMG) Web site, http://www.omg.org/. The most current available version is the intended reference.