Service-Aware Interoperability Framework - Canonical Definition

This is the informative edition of the Service-Aware Interoperability Framework - Canonical Definition (SAIF-CD).

NOTE to Readers: This document contains the *informative ballot* content of the SAIF-CD. Every effort was made to incorporate all of the comments received in the ballot into this document.. A future release of this document will defined specific requirements that a given SAIF IG must meet in order to be viewed as a SAIF-CD-compliant SAIF IG. The document containing those requirements will be submitted for *normative DSTU ballot.*

Chair	Charlie Mead National Cancer Institute, Center for Biomedical Informatics and Information Technology
Vice Chair	Ron Parker Canada Infoway
Secretary	Anthony Julian Mayo Clinic
Technical Editor	Ann Wiley
Sponsoring Work Group	Architecture and Review Board
List Server	arb@lists.hl7.org

In addition, the ArB wishes to acknowledge the contributions of the following persons:

Andy Bond	NEHTA
Jane Curry	Health Information Strategies
Grahame Grieve	Health Intersections Pty Ltd
Steve Hufnagel	U.S. Department of Defense, Military Health System
John Koisch	Guidewire Architecture
Patrick Loyd	Icode Solutions
Cecil Lynch	Accenture
Zoran Milosevic	NEHTA
Wendell Ocasio	Agilex Technologies
John Quinn	Health Level Seven, Inc.
Abdul Malik Shakir	Shakir Consulting
D. Mead Walker	Health Data and Interoperability Inc.

Table of Contents

Table of Figures

1 Introduction

1.1 Background

The development of the SAIF Canonical Definition (SAIF-CD) – which began in early 2008 – was motivated and directed by a high-level set of requirements communicated to the Health Level Seven International (HL7) Architecture Board (ArB) by the HL7 Chief Technology Officer (CTO) and senior representatives of several large national programs whose representatives participate in various HL7 activities. In particular, the ArB was asked to specify an "enterprise architecture approach" to the development of HL7 specifications. In particular, the ArB was asked to provide a coherent, enterprise-architecture-aware approach that would enable the explicit description of technology components – including but not necessarily limited to HL7-specified components – from the perspective of the interactions between those components as they were involved in scenarios whose purpose was to achieve an agreed-upon goal based on "cross-organizational-boundary shared purpose." The scope of the components themselves was not specified, i.e. a "component" could be defined as a system, a service, an enterprise, or a generic party. The notion of "interactions to achieve an agreed upon goal based on cross-organizational-boundary shared purpose" was assumed to mean – at a technical level – some degree of technical interoperability between the involved components that itself was a manifestation of a non-technical agreement and definition of a joint (i.e. cross-organizational-boundary) shared purpose.

NOTE: From this point forward, this document will use the term "cross-boundary" to indicate scenarios which involve interactions/interoperability across one of a number of possible boundaries, e.g. departmental/disciplinary, organizational, enterprise, jurisdictional, etc. A common – but not required – characteristic of cross-boundary interactions is the fact that not all of the components/systems/technologies/required resources required for the interaction are under the under the control of a single resource.

As the ArB began considering its task from the perspective of the collective experience of its members, the core effort soon became focused on standardizing a set of languages that could be used to explicitly define various factors that enable interoperability between the components. In particular, the ArB focused on defining a set of *canonical frameworks* that could then be instantiated in organization-specific Implementation Guides (IG) as specific grammars. The distinct between the *languages* defined by the SAIF-CD and an organization-specific IG's *grammars* is explicated in the Wikipedia definitions of the two terms:

Language: *When described as a system of symbolic communication, language is traditionally seen as consisting of three parts: signs, meanings and a code connecting signs with their meanings. The study of how signs and meanings are combined, used and interpreted is called semiotics. Signs can be composed of sounds, gestures, letters or symbols, depending on whether the language is spoken, signed or written, and they can be combined into complex signs such as words and phrases. When used in communication a sign is encoded and transmitted by a sender through a channel to a receiver who decodes it (a signal).*

Language (SAIF-CD): The concepts and relationships defined in the SAIF-CD. Many are taken from the Enterprise Viewpoint and Computational Viewpoint languages of RM-ODP (ISO RM-ODP).

Grammar: The study of how meaningful elements (morphemes) within a language can be combined into utterances. Morphemes can either be free or bound. If they are free to be moved around within an utterance, they are usually called words, and if they are bound to other words or morphemes, they are called affixes. The way in which meaningful elements can be combined within a language is governed by rules. In standard linguistic theory the rules of the internal structure of words is called morphology. The rules of the internal structure of the phrases and sentences is called syntax.[17] In the generativist tradition of Chomsky morphology is seen as a part of syntax.

Grammar (SAIF-CD): The adoption or adaption, optimization, realization, and/or contextualization of the languages specified in the SAIF-CD for use in *organization-specific* SAIF Implementation Guides(SAIF IG).

The need for the separation of a single common SAIF *language* – as defined in the SAIF Canonical Definition specification, as opposed to the use of this language in any number as Implementation Guide-specific *grammars* – grew out of the recognition by the ArB that no single framework could – or should – be dictated by the ArB (or any

other body, for that matter). However, both the HL7 CTO and the ArB felt strongly that there was value in having a common language/collection of languages that could be used to define and discuss the various aspects of component-to-component interoperability.

In addition, it was also recognized that, in addition to language needed to discuss the technical aspects of shared purpose interoperability scenarios, a formal governance language which allowed the clear expression of the formal linkages between organization-level definition of shared purpose and its technical realization in specific run-time components was also required, i.e. technical component interoperability is, in fact, a manifestation of a "higher level" of cross-organization/cross-boundary (in the jurisdictional or administrative sense) agreements between human beings and/or the organizations they represent. These requirements were repeatedly reinforced to the ArB on numerous occasions over the past three years through dialogues with various external stakeholders including, but not limited to, representatives from large/national programs.

Thus, the SAIF-CD defines a minimal set of common concepts and relationships from which compliant SAIF IG models can be defined that, in turn, support a number of different technical approaches – e.g. messages, documents, or services – which enable the successful realization of shared purpose scenarios. A SAIF IG thus adopts and defines modeling languages and document artifact templates compliant with the concepts and properties defined in the SAIF-CD. In terms of the separation between *language* and *grammar* mentioned above, the SAIF-CD defines a *language* – or, more correctly a set of inter-linked languages – that a particular organization can use to specify organization-specific *grammars* – documented in the organization's SAIF Implementation Guide – which define how an organization documents the various interoperability aspects of components involved in shared purpose scenarios. As such, *IG-specific grammars adopt, adapt, organize, realize, and contextualize the SAIF-CD languages in ways suitable for the organization's own interoperability requirements and goals using that organization's adopted (or adapted) modeling conventions and specific grammars, reference models, technology choices, etc.*

It should also be noted that the concept of *interoperability* in the context of the SAIF-CD is rather broad-based. In particular, it is ultimately based on the basic notion of *shared purpose* resulting in defined value for the various parties involved in interoperability scenarios. Specifically, interoperability at a technical level may be characterized as one of several interoperability types, involving simply the exchange of structure (syntax) versus the more difficult exchange of meaning (semantics) between humans (e.g. browser-compatible documents) versus machines. Thus, defining and achieving shared purpose between two organizations, via an implementation involving various software components designed, developed, and deployed by the organizations, includes a context-specific discussion of human-to-human, human-to-machine, or machine-to-machine interactions. Experience has repeatedly shown that semantic interoperability between machines – known as *computable semantic interoperability* (CSI) – is by far the most difficult and expensive type of interoperability to achieve in a scalable, tractable manner, particularly when the interoperability scenarios cross one or more organizational boundaries (a construct that the SAIF-CD refers to as the "deployment context" of the scenario. See the Governance Framework and the Appendix for more discussion on Interoperability Type versus Deployment Context.)

Given the fact that an enterprise architecture should support the business of the enterprise that defines and develops that enterprise architecture, it is important to note that the SAIF-CD was is specifically meant to function not as a replacement for, but rather as an adjunct to, existing enterprise-centric architecture frameworks including RM-ODP (ISO RM-ODP), Zachman2 (Zachman), TOGAF (The Open Group), DoDAF (US Department of Defense Architecture Framework), Lopez/Blobel's description of a healthcare-specific architecture (Lopez, 2009), etc. Specifically, the SAIF-CD defines the languages necessary for focusing component specification on cross-boundary (e.g. cross-enterprise) interoperability that is itself focused on achieving a mutually beneficial shared purpose.

1.1.1.1 *Overview of the SAIF-CD*

The purpose of the HL7 Service-Aware Interoperability Framework Canonical Definition (SAIF-CD) is to provide the "top-level" specification of SAIF. As such, the SAIF-CD is written for persons or organizations that are interested in implanting SAIF as an adjunct to existing (or planned) enterprise architecture frameworks because of SAIF's singular focus on the various dimensions and perspectives associated not with enterprise architecture *per se*, but rather with achieving predictable, scalable, and effective *interoperability* between the various software components that collectively populate *one or more* enterprise architectures. Such implementation is most effectively

done through the development of an organization-specific SAIF Implementation Guide (SAIF IG). Examples of some of the specific steps and end results of using the SAIF-CD to define a specific SAIF IG are collected in the Appendices of this document. The following concept map provides a high-level overview of the SAIF-CD:

- Blue concepts are defined in the SAIF-CD
- Yellow concepts in an organization specific IG
- Green concepts are instance specifications developed using definitions supplied by a specific SAIF-IG
- Terra-cotta concepts identify external resource information, e.g. The RM-ODP standard.
- Purple – not present in Figure 1 – is used to indicate run-time instances of specification instances (colored green)

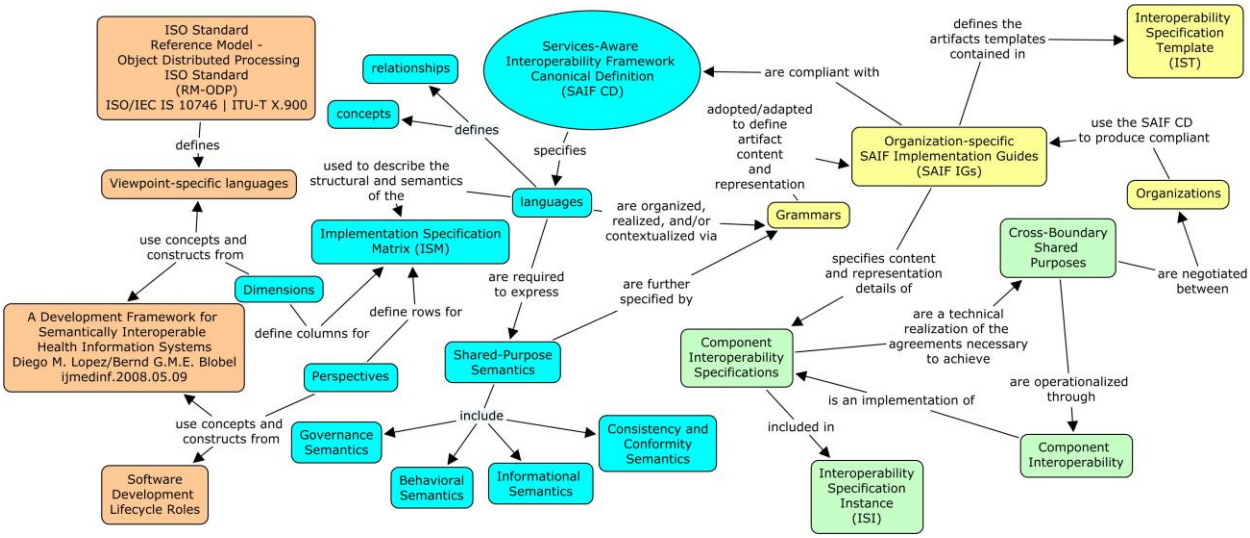


Figure 1 SAIF-CD organization and structure

The SAIF-CD uses core concepts and constructs of the ISO standard Reference Model for Open Distributed Processing (RM-ODP) (ISO RM-ODP). As explained in Section 6, the columns of the SAIF-CD Interoperability Specification Matrix (ISM) are related to – but *not isomorphic* to – the like-named ODP Viewpoints. As defined by the ISM, Dimensions intersect with role-based Perspectives to form the Interoperability Specification Matrix, supporting explicit, layered, multi-factorial component analysis and design with a focus on component interoperability. Perspectives are roughly equivalent to levels-of-abstraction, but are more correctly viewed as role-based Perspectives, that is, views of a particular Dimension from the perspective of SMEs and "outward-facing analysts," (Conceptual Perspective), architects and "inward-facing analysts" (Logical Perspective), and developers and designers (Implementable Perspective). SAIF-CD Perspectives provide the opportunity to represent Dimension-specific views of subject matter experts and component users as well as analysts, architects, designers, implementers and testers. This approach is in distinct contrast to that of ODP, which has an implied rather than explicit layering of perspectives. The ArB feels that the explicit representation of role-based perspectives in the SAIF-CD is critical to achieving predictable and tractable success in complex interoperability scenarios. In particular, the explicit separation and representation of Perspectives versus Dimensions allows for the co-existence, where appropriate, of multiple – but ultimately coherent and consistent – Perspectives within a single SAIF Dimension. This is a manifestation of the need to directly support the many uses of SAIF-complaint specifications which can then be made by different stakeholders within one or more interoperable communities.

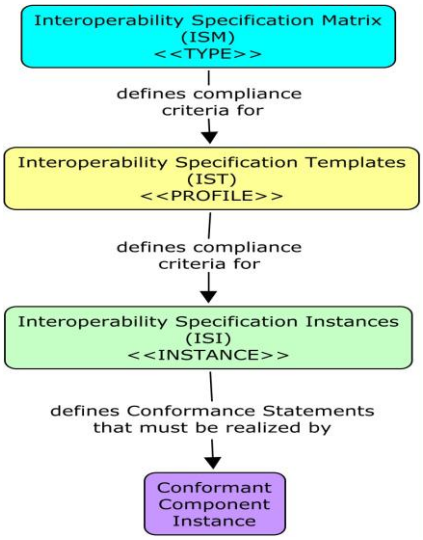
NOTE: Use of concepts taken from the ODP Viewpoints in combination with SAIF Perspectives provides SAIF the basis for addressing issues that directly emerge from focusing on interoperability scenarios. In particular, the SAIF-CD leverages the core intent of the ODP standards, to provide a technology-independent framework for specifying enterprise distributed systems, while explicitly providing mechanisms for addressing various organizational modeling issues. Examples are organizational and legislative polices defined by the administrative boundaries, and regional and state jurisdictions – issues which are explicitly addressed in the SAIF-CD through the use of Perspectives.

1.1.2 The SAIF-CD, SAIF IGs, and IG-compliant artifacts

Critical to understanding the operationalization of SAIF is the distinction of what is defined where, i.e. what is defined in the SAIF Canonical Definition, a particular enterprise's SAIF Implementation Guide (e.g. the HL7 SAIF IG), and the instantiation of component interoperability specifications and implementations that are, in turn, compliant (specifications) or conformant (implementations) with the artifact content and representation constructs defined by the governing SAIF IG. The HL7 SAIF-CD is intended to be used primarily by the authors of an enterprise's SAIF IG and therefore its value to an enterprise's analysts, architects, developers, or other enterprise architecture stakeholders is more as reference material, since they would be more directly utilizing the enterprise's SAIF IG.

The "SAIF stack" consists of four levels which can be conceptually viewed as representing a Type, Profile, and Instance *specification* hierarchy and an associated implementation instance of a given specification instance:

- The SAIF Canonical Definition (SAIF-CD)
- Enterprise-specific and SAIF-CD-compliant SAIF Implementation Guides (SAIF IGs)
- SAIF IG-compliant component specification instances
- Conformant component implementations having component-specific static and dynamic aspects related to the component's participation in cross-boundary shared purpose interoperability scenarios.
- In the following concept map, this most visible vestige of the "SAIF stack" – the Interoperability Specification Matrix and its derivatives – is shown. In particular, it is important to note that the SAIF-CD defines a *single* Interoperability Specification Matrix (ISM) as a *type.* One-to-many SAIF Implementation Guides (SAIF IGs) can then be defined as *profiles* on that type. A substantive portion of a SAIF IG is, in fact, the specification of the content, representation, and specific cell location(s) for each artifact in the SAIF IG-specific Interoperability Specification Template (IST). Finally, as a given SAIF IG is operationalized, any number of specification *instances* are produced, each referred to as an Interoperability Specification Instance (ISI). Following specification, one or more *implementation instances* of a given specification instance may be developed and – if so desired – subject to conformity testing. These concepts and relationships are discussed in more detail in the remainder of this document.
- Figure 2 depicts the Relationship between SAIF-CD as a Type, compliant SAIF Implementation Guides (IGs) as profiles on that type, instances of component specifications as instances, and Conformant Component Instances. See Section 6 and Appendix for more detailed discussion



-
-

Figure 2 Relationship between SAIF-CD as a Type, compliant SAIF Implementation Guides (IGs)

The SAIF-CD defines the essential concepts and constructs necessary for an organization to define its own SAIF Implementation Guide (SAIF IG) in such a manner that that IG will be compliant with the SAIF-CD. The basic structure of the SAIF-CD as well as its high-level relationship to enterprises and their architectures and SAIF IGs is shown in the following concept map.

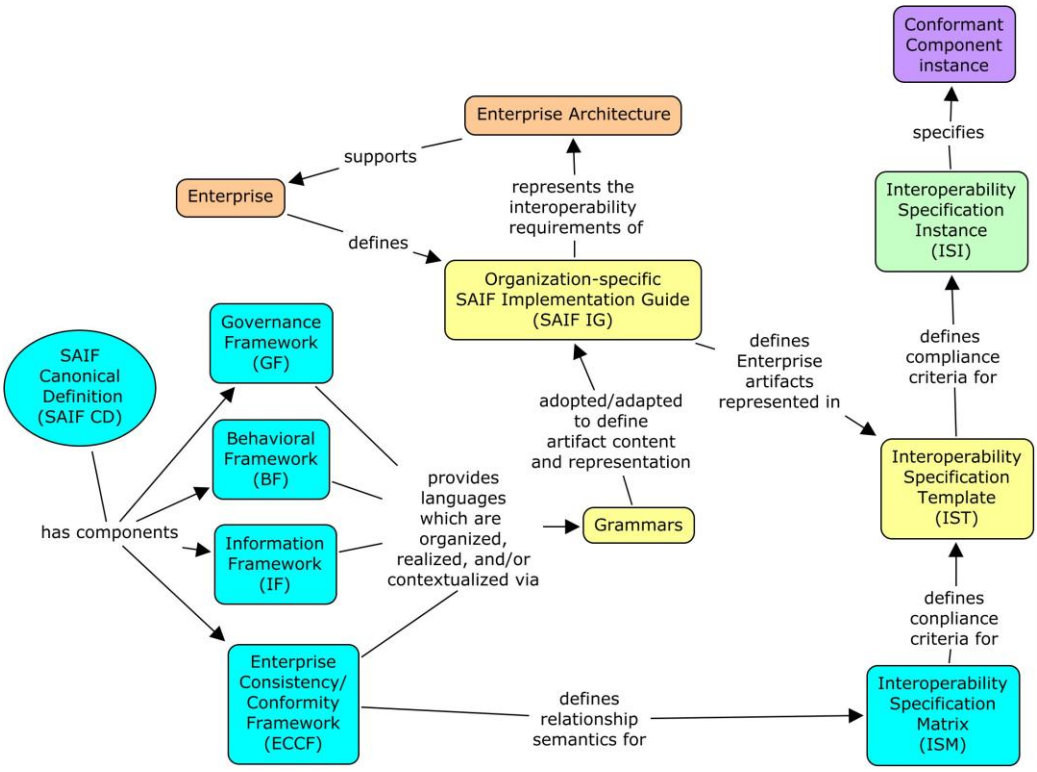


Figure 3 – SAIF-CD: basic structure. (See Figure 1 notes for meaning of colors).

1.1.3 The SAIF Value Proposition

The SAIF-CD defines a specification that can be used by multiple organizations to build organization-specific, SAIF-CD-compliant SAIF Implementation Guides (SAIF IGs). An organization interested solely in intra-enterprise

component interoperability could certainly define a "SAIF-like" set of requirements for the artifacts needed to collectively specify a given software component to interoperate with other components without the use of the SAIF-CD per se. However, achieving inter-organization, i.e. cross-boundary, interoperability presents greater challenges since it is necessary to ensure that the "expectations" of each party involved in a given interoperability scenario, as manifested in a particular software component developed by one of the participating parties, have been quantitatively assessed for completeness and correctness

If both organizations have specified their respective components using their own SAIF-CD-conformant SAIF IG, the task of component specification comparison and (if necessary, refactoring) becomes considerably more tractable because the framework within which the comparison is done, the SAIF-CD-compliant SAIF IGs, eliminates or minimizes many of the operational differences between the two organizations' ways of defining component semantics and their representations. The development of SAIF-CD compliant SAIF IGs enables organizations to explicitly discuss and negotiate their *cross-boundary shared purposes* as operationalized in component interoperability.

It should be noted, however, that independently designed components may still not be interoperable due to incompatible requirements. However, if specifications are explicit and expressed using the language provided by the SAIF IG, targeted harmonization, alignment, and refactoring can more effectively and efficiently take place. In summary, negotiations between various information exchange communities can lead to explicit agreements that can result in components participating in a truly distributed, interoperable ecosystem. SAIF thus enables cross-boundary risk reduction in the context of interoperability scenarios requirements.

The SAIF-CD explicitly defines the languages for explicitly specifying informational (static) and behavioral (dynamic) semantics at the level of a software component (for example, services, messages, and documents). In addition, it provides direction as to how Conformance Statements may be included in a given specification instance. Specification-specific Conformance Statements can then be associated with pair-wise, implementation-instance-specific Conformance Assertions to assess the conformity of a given run-time Component Implementation.

1.1.4 The Four SAIF-CD Frameworks

1.1.4.1 *Governance Framework (GF)*

The Governance Framework (GF) language enables an enterprise implementing SAIF to define explicit, organization-specific policies, standards and roles to artifact-specific content and representational choices that use the languages specified in the Behavior and Information Frameworks. The overall management of the life cycle of each SAIF artifact, including the correctness and completeness and any IG-specified RACI relationships, is defined by the Governance Framework language. As such, the GF aides an organization in risk management by providing a language that can be used to apply governance at specific high-risk operational points.

The GF uses a documentation framework adopted from a recent publication (Thomas Erl, 2011). As explained in detail in the GF discussion in this document, the framework includes Precepts – further defined in terms of Objectives, Policies, Standards, Guidelines – People (and their associated Roles and including both organizations and systems), Processes, and Metrics. A SAIF-IG operationalizes the GF language in an organization-specific SAIF IG grammar, to explicitly cover concepts like expectations, granting of authority and resources, verifying performance, managing configuration baselines and related concerns.

Cross-boundary shared purpose as it is achieved through technical interoperability represents a set of agreements between the human and organizational owners of the components that are ultimately deployed and interact to achieve a defined set of shared objectives. In particular, technical, component-specific contracts are specified as a means of providing technical realizations of formal (or informal) contracts between human beings and enterprises. As such, readers of the SAIF-CD will note this intersection of the human and organizational and technical perspectives on interoperability in many of the terms used in both the Behavioral Framework and Governance Framework chapters of the SAIF-CD.

NOTE: The language describing certain targeted types of governance -- e.g. artifact and Interoperability Specification Template well-formed-ness, and conformance and compliance testing and certification of

specification-specific implementations – is defined in a separate SAIF-CD chapter, i.e. the Enterprise Consistency and Conformity Framework (ECCF).

Note to SAIF IG Developers: ***It is not necessarily true that a given SAIF IG will cover the complete scope of the GF lanaguage. In addition, it is not the case that only a single grammar will be required to cover all three of the Interoperability Specification Matrix (ISM) Perspectives with respect to governance semantics involved in organization-specific specification content, syntax and representation. In fact, different Perspectivesmay naturally give rise to different grammars (and representations) in the context of a given conformant SAIF IG. In addition, the GF language has application outside of the ISM because of its role as a "bridge" between organizational agreements stating and technical implementations realizing cross-boundary shared purpose.***

1.1.4.2 *Behavioral Framework (BF)*

The language of the Behavioral Framework (BF) defines constructs to specify the dynamic semantics of interactions in a shared purpose interoperability scenario. The BF focuses on the languages necessary to define the semantics of *contracts*, *operations*, and *processes* that collectively define shared purpose scenarios *at a technical level*. Collectively, the BF languages – and their IG-specific grammars – describe "*who does what when and how*." In particular, contracts are expressed as implicit or explicit agreements at a number of jurisdictional boundaries including those between business objects, components, applications, systems and/or enterprises/organizations. The BF language specifies constructs describing various system role relationships expected by various stakeholders, system components, and/or applications. These relationships involve information exchanges and behavioral interactions in support of shared purpose scenarios.

The other SAIF-CD frameworks work with – and in support of – the BF. In particular, the GF provides the language to both define the non-technical constructs of shared purpose, as well as to bind organizational and technical risk management to component development and use. The IF and BF languages enable the explicit specification of business objects, components and their services, capabilities, applications, systems and their respective roles, responsibilities and interactions such as information exchanges. The ISM and the ECCF provide the structure and language for documenting and managing technical component specifications.

Note to SAIF IG Developers: ***It is not necessarily true that a given SAIF IG will cover the complete scope of the BF lanaguage. In addition, it is not the case that only a single grammar will be required to cover all three of the Interoperability Specification Matrix (ISM) Perspectives with respect to behavioral semantics involved in organization-specific specification content, syntax and representation. In fact, different Perspective may naturally give rise to different grammars (and representations) in the context of a given conformant SAIF IG.***

1.1.4.3 *Information Framework (IF)*

The Information Framework (IF) defines the language required for discussing and defining the static/informational semantics relevant to interoperability scenarios including concepts such as information and terminology models, metadata, vocabulary bindings, value sets, executable models, etc. that collectively specify the static semantics of interactions. This includes the language to describe patterns of structured and unstructured data, documents, messages and services, quality measures and transformations.

The IF also defines the language necessary to explicitly describe how these various information/static semantic constructs are related to each other in a composite static semantic "whole" in the context of a shared purpose interoperability scenario.

Note to SAIF IG Developers: *It is* ***not necessarily true that a given SAIF IG will cover the complete scope of the IF lanaguage. In addition, it is not the case that only a single grammar will be required to cover all three of the Interoperability Specification Matrix (ISM) Perspectives with respect to informational semantics involved in organization-specific specification content, syntax and representation. In fact, different Perspective may naturally give rise to different grammars (and representations) in the context of a given conformant SAIF IG.***

1.1.4.4 *The Enterprise Consistency and Conformity Framework and the Interoperability Specification Matrix*

The Enterprise Consistency and Conformity Framework (ECCF) defines the language necessary to describe the various *relationships* – e.g. conformance, compliance, consistency, traceability, compatibility, etc. – between the

artifacts that collectively define a given specification, including how a given specification relates to both derived implementations of the specification, and other specifications that use one or more of the artifacts as part of their artifact collection. In contrast, the ISM itself defines the structure – a 5 x 3 *non-normalized* matrix – that is used to collect the various artifacts that collectively specify information exchange and interaction details that define a component's capabilities and accountabilities. IG-specific instances of the ISM – referred to as Interoperability Specification Templates (ISTs) – actually collect the various artifacts and artifact-specific Conformance Statements that can be used to evaluate the conformance of a given application instance to a given specification. Thus, the IF and BF formally define the essential concepts and relationships necessary to define within a given SAIF-IG, i.e. *what* can be specified, the ISM defines how artifacts can be sorted and collected based on their particular Dimension and Perspective, while the ECCF defines the relationships between artifacts.

Note to SAIF IG Developers: ***It is not necessarily true that a given SAIF IG will cover the complete scope of the ECCF lanaguage. In addition, it is not the case that only a single grammar will be required to cover all three of the Interoperability Specification Matrix (ISM) Perspectives with respect to consistency and conformity semantics involved in organization-specific specification content, syntax and representation. In fact, different Perspective may naturally give rise to different grammars (and representations) in the context of a given conformant SAIF IG.***

1.1.4.5 *Inter-relationships among the four SAIF-CD Languages*

The four languages of the SAIF-CD – i.e. the GF, BF, IF, and ECCF – should not be viewed as siblings. Rather, they have a number of inter-relationships that, when understood, provide a layered, multi-dimensional view of the SAIF-CD as a specification for SAIF IGs. In particular, three relationships and their unifying concepts are of primary importance:

- GF and BF – related through the concepts of Shared Purpose and Objectives, and Role-based Communities and the subtype Governance-based communities
- GF and ECCF – related through the concept of Artifact Governance
- ECCF, BF and IF – related through the concepts of artifact syntax and semantics, and well-formed-ness.

The following concept map provides a graphical view of these pivotal SAIF-CD inter-relationships:

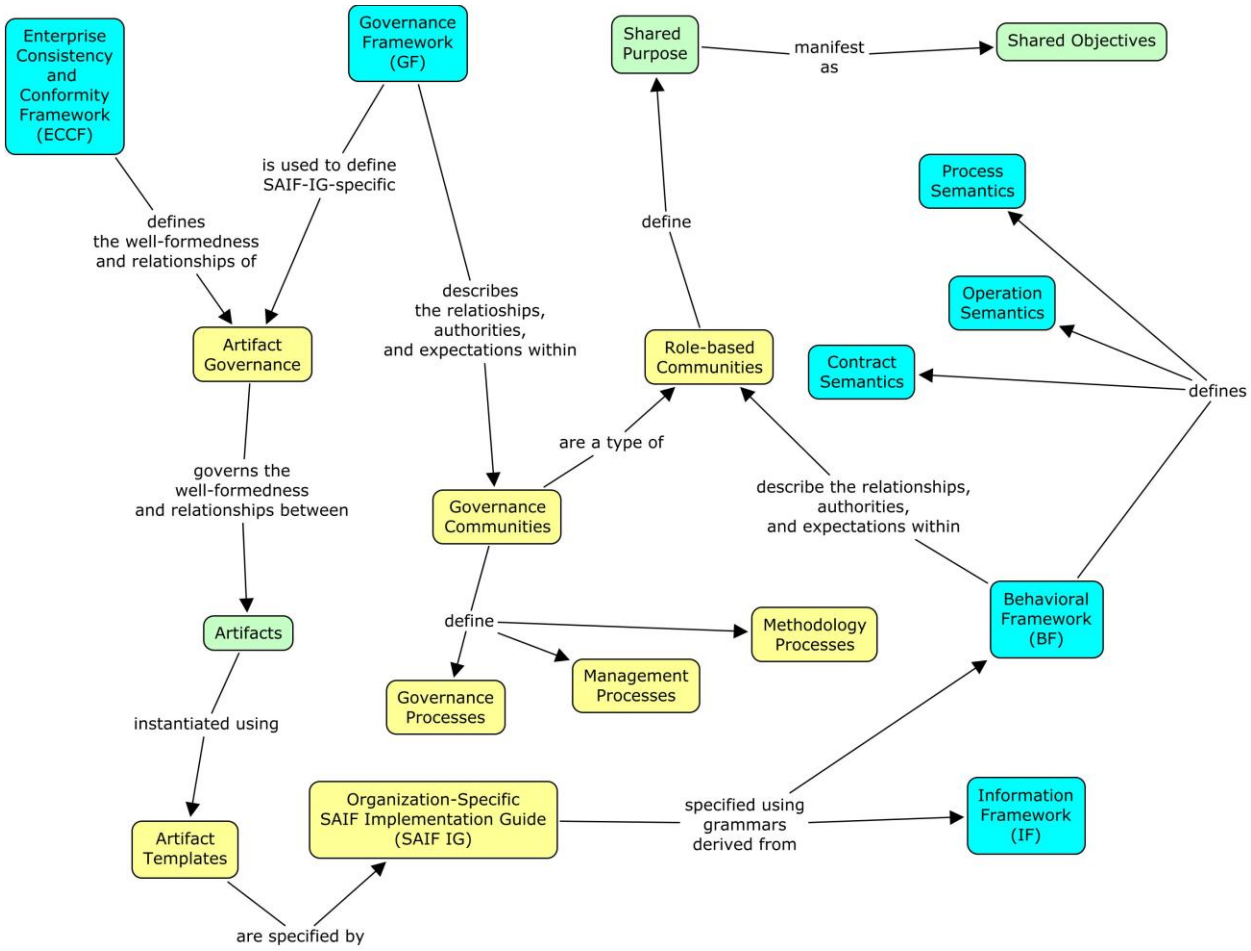


Figure 4 Inter-relationships of four SAIF-CD languages

1.1.4.6 *SAIF-CD Adoption and Adaption of existing and/or related work*

With respect to the criticism voiced by several members of the community that the SAIF-CD specification is not sufficiently aware of existing work, it is important to understand that the SAIF Canonical Definition defines common concepts and patterns that will subsequently be instantiated through the concrete artifact specification definitions in the various IGs. The reuse of existing work is thus – for the most part – an IG-level and not a Canonical Definition-level issue.

The ArB does not agree with statements that suggest that SAIF is not aware of work in other groups, for example, OASIS, UML/OMG, and TOG. SAIF makes considerable use of the ODP's Enterprise and Computational languages. In particular, the development of the UML profile for ODP and other UML specifications, for example, SoaML, MOF, and certain aspects of UML 2.x, have been directly influenced by ODP. Finally, there is considerable alignment between ODP and the latest OASIS SOA Reference Architecture Foundations and the TOGAF 9 meta-model. All of these developments and correspondences underscore the validity of the ArB's choice to use ODP as the basis for the SAIF Canonical Definition.

However, the ArB does believe that many of these efforts cited above are insufficiently focused on the important issue of the explicit representation of computationally-capable static and behavioral semantics, that is, they do not *a priori* start from the position of "interoperability as a 1st-class citizen."

The efforts tend to be focused on a single enterprise rather than taking a cross-enterprise view and, as a result, do not bring sufficient rigor to the importance of cross-enterprise standards at both the human and technology level in the larger context of understanding component capabilities from a cross-enterprise interoperability perspective; and the

efforts do not explicitly define their various "viewpoints" from multiple role-based perspectives, a feature that is essential in surfacing critical component characteristics from an interoperability perspective.

1.1.5 Conventions Used in this Document

1.1.5.1 *Index*

Readers will find a comprehensive Index at the end of this document. Every attempt has been made to make the Index useful for targeted reference to selected topics within the SAIF Canonical Definition document.

1.1.5.2 *Glossary*

The SAIF Canonical Definition document does not include a Glossary. Rather, the HL7 Architecture Board (ArB) maintains an online SAIF Glossary—http://www.SAIFGlossary.xxx—that includes definitions of relevant terms, specialized concepts, constructs, and artifacts as used in either or both the SAIF Canonical Definition and HL7 SAIF Implementation Guide. The online Glossary is updated between publications of the SAIF-CD.

1.1.5.3 *Reference Material*

Reference Material containing additional information that is not part of the SAIF Canonical Definition including material such as auxiliary diagrams, examples, and additional explanations of material formally presented in the SAIF Canonical Definition document but deemed to not be an essential part of the balloted, normative content can be found in the various Appendices to the SAIF-CD.

1.1.5.4 *Footnotes*

When absolutely necessary for clarification of critical concepts, the SAIF Canonical Definition document includes footnotes. In the SAIF Canonical Definition document, footnotes are not, in general, used to provide definitions as these are collected in the SAIF Online Glossary. (HL7 ArB, 2011)

1.1.5.5 *Reader Feedback*

Readers wishing to suggest improvements to materials in this SAIF Canonical Definition are encouraged to subscribe to the HL7 Architecture Board list server and send their suggestions to arb@hl7lists.org.

1.2 Governance Framework

2 Purpose

The purpose of the Governance Framework (GF) is to provide a language and set of constructs for individual organizations to define explicit sets of terms and processes that make the often-implicit "rules of the game" explicit, and thereby ensure a common – i.e. shared – understanding between the various organizations that are focused on achieving a given *jointly negotiated shared purpose*. Specifically, this is meant in the context of realizing such shared purpose in a technical solution that requires a specified type of interoperability (see Figure 5: Interoperability Types versus Deployment Context). In addition, the language of the GF enables organization-specific governance activities to be focused on known development-cycle risks, thereby maximizing the effectiveness and efficiency of resources expended in the name of governance.

2.1.1 Governance, Management, and Methodology

Governance is *not* equivalent to either management or methodology. Rather, it is both influenced by and related to both concepts. Following is a brief list of some of the differences between these three interrelated concepts [(reference)]:

- *Governance* establishes rules that control decision-making.
- *Methodology* establishes processes that comply with governance rules and may introduce additional rules.
- *Management* makes decisions according to governance rules.

- *Governance* does not dictate when or how to make a decision. It determines who should make the decision and establishes limits for that person or group.
- *Methodology* establishes processes that carry out specific types of decision that adhere to governance rules.
- *Management* is responsible for day-to-day operations and for ensuring that decisions made adhere to governance and methodology rules.

- *Governance* cannot replace management or methodology, nor can it compensate for poor management or poor (or inappropriate) methodology.
- Poorly defined and executed *methodology* can jeopardize the business goals associated with governance.
- Poor *management* can undermine a governance system and a methodology and will jeopardize associated business goals.
- Neither management nor methodology can replace governance, nor compensate for poor governance.

Governance is therefore best seen as a "meta" process which describes and oversees "how decisions about decision making" are made. At a high level, a well-defined governance system is characterized as having [(reference)]:

- *identified* constraints and control guidelines on management decisions
- *defined* the responsibility for and authority to make various decisions
- *enumerated* the consequences of non-compliance to governance metrics

Thomas Erl's recent book summarizes governance as follows:

"A good system of governance helps the members of an organization carry out responsibilities in a manner supportive of the organization's business goals and vision. It mitigates conflict by clearly defining responsibilities and assignments of authority, and further reduces ambiguity by articulating constraints and parameters in practical forms (such as rules and decision guidelines). It also helps balance tactical and strategic goals by expressing the intents and purposes of its rules. (Thomas Erl, 2011)"

2.1.2 Shared Purpose

As stated above, the GF provides a language that can be used to explicitly define a set of "governed items and associated processes" including the relevant artifacts, metrics, roles, etc. It is important to note that the language of the GF is ***not*** specific to either the governance of people, organizations, enterprises, etc., or the governance of technology components, i.e. it applies equally well in both settings. This feature is of essential importance since, in fact, the governance that occurs at a computational interface via constructs such as pre-conditions, post conditions, contracts, roles, accountabilities, etc. is, in fact, a technical realization of an agreement between two or more participating parties to achieve a *shared purpose*. In order to be successful, such an agreement must clearly define responsibilities, expectations, and response to non-performance, the basic content of a contract.

Although governance is an important construct within a single department/organization/enterprise, it becomes a critical success factor when more than one independent entity – i.e. when the entities seeking to achieve a given shared purpose come from different governance spheres. The SAIF-CD assumes that execution context to achieve the shared purpose will be realized through a collection of technology-based components, the *explicit details* of which can be expressed in artifacts defined by SAIF Implementation Guides using the languages of the Behavioral, Information, and Enterprise Consistency and Conformity Frameworks defined in the SAIF-CD. The details of the shared purpose are not critical to the use of the language of the GF, i.e. governance is needed because the shared purpose of the community is to achieve objectives that cannot be achieved by participants acting autonomously. Thus, the shared purpose could be setting or refining international standards, collaborating to deliver healthcare services, developing technical components to enable system interoperability in order to share information or coordinate component behaviors in the context of healthcare delivery, health program evaluation, research, quality assurance, research or clinical trial needs, regulatory reporting obligations, etc. In the context of technical interoperability and shared purpose, well-defined governance is a Critical Success Factor.

Finally, it should be noted that governance is not a "one size fits all" construct. In fact, there are numerous dimensions that govern the decisions that will ultimately answer the questions "What needs to be governed?" and "How should it be governed?" In response to the first question, the GF provides language that can productively be applied to mitigate risk. With respect to the second question, two of the most important dimensions that determine "how much governance" a particular negotiated instance of shared purposed interoperability requires in order to succeed are Interoperability Type and Deployment Context. (See Appendix for a detailed discussion of the relationship between these two constructs.)

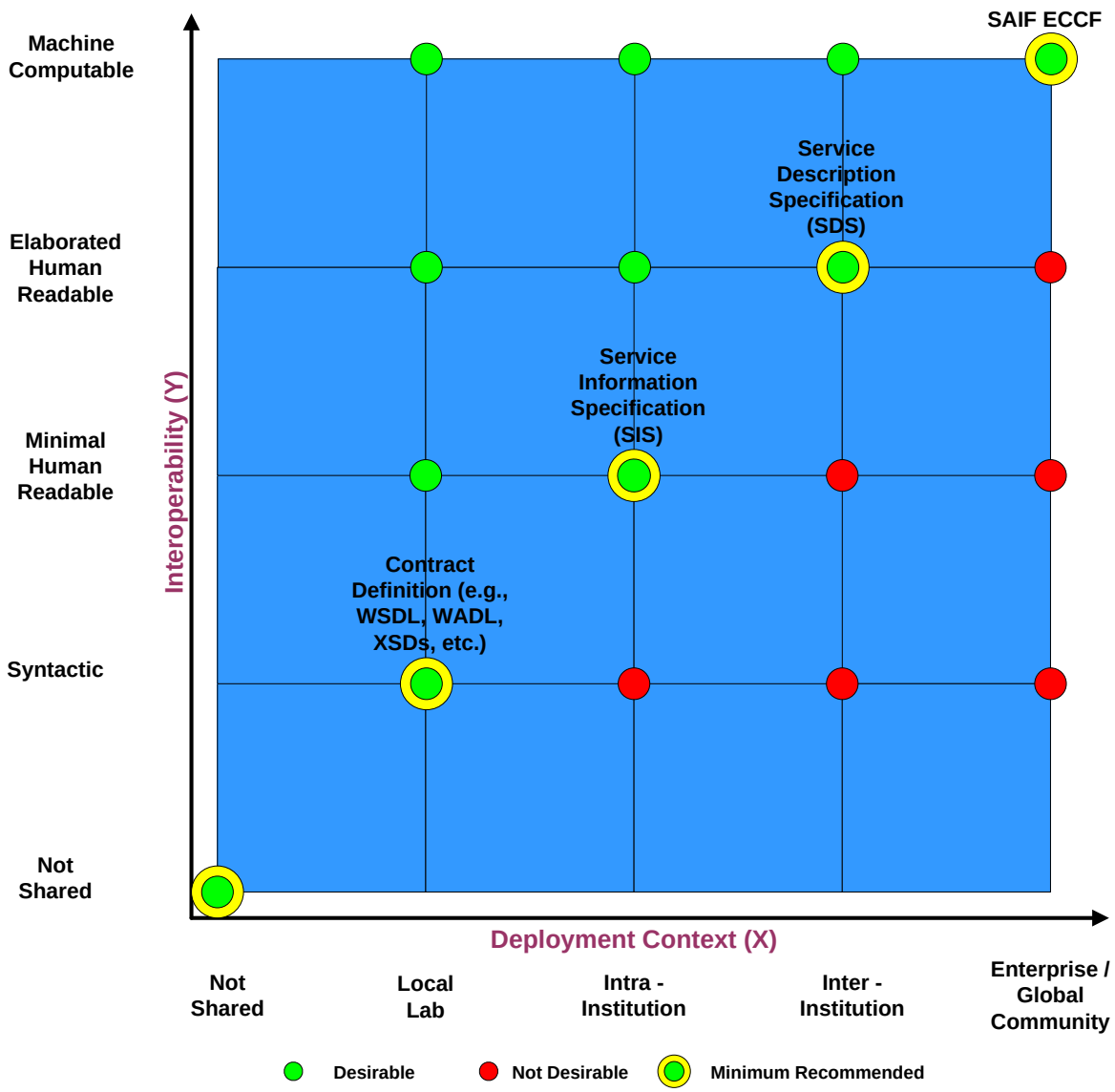


Figure 5 The amount and type of governance

Figure 5 above depicts the amount and type of governance required for a given shared purpose interoperability scenario depends on multiple factors, two of the most important being the Deployment Context and the Interoperability Type that contextualizes a particular shared purpose scenarios.

In summary, the parties participating in a shared purpose scenario realized through technical component interoperability do not need to agree to be governed by the same set of rules for all aspects of their respective operation. Those rules affecting their participation in shared activities need to be explicitly defined and negotiated through a GF-based mechanism. The establishment of shared rules is intended to reduce risks when working across boundaries. Evaluation of the types and impact of potential risks will prioritize those areas where clear "rules of engagement" are essential to success.

2.2 GF Concept Map

The core concepts and relationships of the GF languge are pictured in the Concept Map and defined in the following section "GF Terms of Art." Note in particular that the concept of "governance" itself – as expressed via the use of GF language – is colored yellow to indicate that it is an *organization-specific* constuct that – as explained earlier in

this document – is expressed through an organization-specific instance of the GF language, i.e. it is expressed in an organization-specific *GF grammar*.

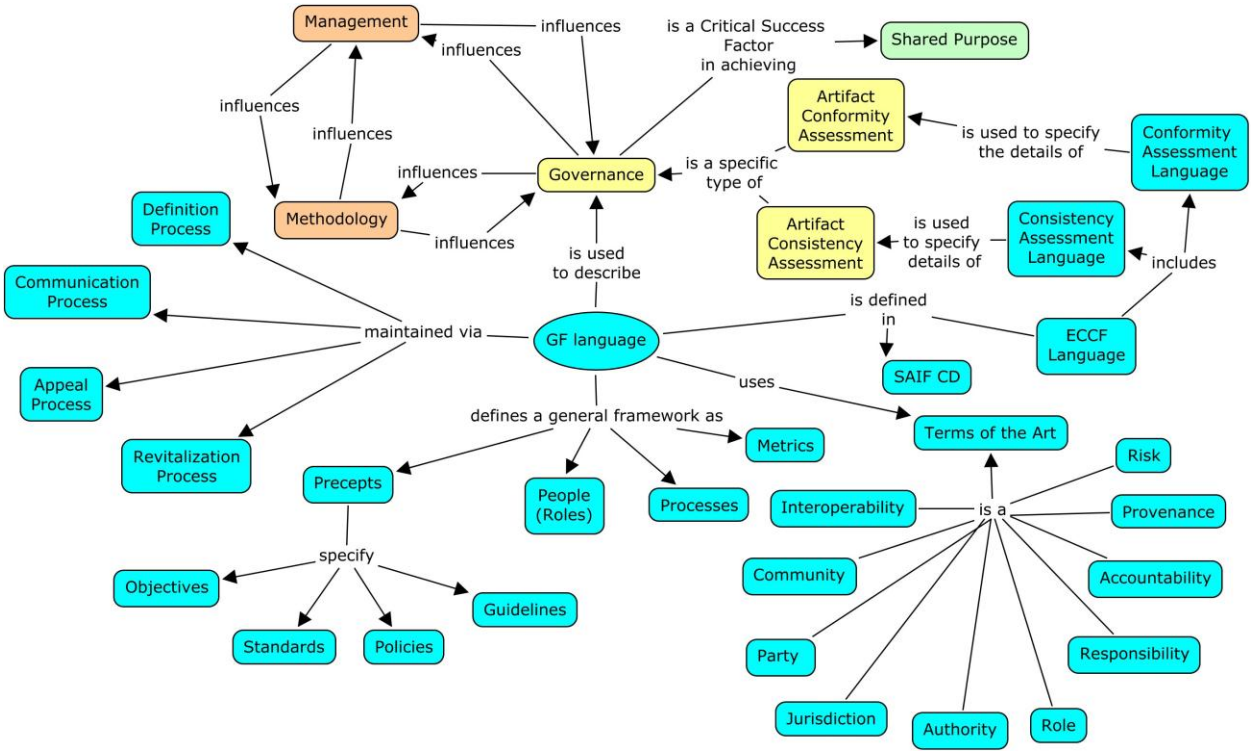


Figure 6 Governance Framework Concept Map

2.2.1 GF Terms of Art

The following terms are used in defining precepts and their relationships to each other. The source of these concepts is generally the *INTERNATIONAL STANDARD ISO/IEC 15414 ITU-T RECOMMENDATION .911 - Information technology – Open distributed processing –Reference model – Enterprise language* (ISO RM-ODP). The concepts are paraphrased here to be more business-reader friendly and to permit this chapter to be read alone. In some cases, concepts are from other named sources. In addition, some concepts are paraphrased to add clarity for this framework.

Note: Several of the concepts in the GF language are similar in meaning to concepts used by the Behavior Framework (BF). If essentially identical semantics for a given BF term are found under another name in the BF, the BF synonym is noted in the GF term's definition. A concept map showing the relationships between GF and BF terms can be found at the end of this section.

2.2.1.1 *Interoperability*

Interoperability is the capability of a set of parties to work in concert to achieve a shared purpose. In the context of the SAIF-CD, it is assumed that at least part of the "work" will involve technology components, standards, etc. Interoperability among parties with different jurisdictions requires a clarification of all boundaries and the means to communicate across them, such that information that originates in one party is able to be understood consistently by another. The IEEE definition states that interoperability is the ability of systems to exchange information and use the information exchanged. How information is used in the receiving system depends on the intent of the exchange. Syntactic interoperability refers to the capability to reliably send and receive information. Semantic interoperability refers to the ability to process the information received with the same understanding of the meaning of the information as the originating system and to use the information received appropriately. Being able to have effective computable interpretation of received information requires a significantly greater codification of meaning than to just reliably display information for a human to interpret.

If information is not commonly understood by the human parties in a collaborating community, the capability of systems being used to support such collaboration will be unable to computationally use the information safely and effectively. Since health information is exchanged and subsequently used to directly or indirectly influence the care of people, misuse of information poses a significant risk that must be mitigated.

2.2.1.2 *Risk*

Risks are adverse outcomes of deliberate acts or external events that are considered of sufficient impact to be actively managed. Types of risks may range from not achieving the shared purpose and objectives, to more profound outcomes such as risking patient safety or violating privacy conventions. Managing risk become conscious mitigation strategies to minimize the probability of the risk event occurring or to reduce the impact if the risk does occur. In any shared purpose scenario, working collaboratively across boundaries increases the potential of risks as well as opportunities for mutual benefits. A Risk Profile is the set of organization-specific or community specific risks which have been identified, categorized, and assessed with respect to their Likelihood and Impact to the organization and/or specific development projects – as that profile is viewed from the perspective of shared purpose.

2.2.1.3 *Community*

[ISO ODP 10746-3] defines community as a configuration of objects formed to meet an objective. The objective is expressed in a contract, which states how objectives can be met by defining roles and interactions required, assignments of objects to the roles, and policies governing their collective behavior.

A community is a set of parties collaborating to achieve a shared purpose. The scope of the community could be across disciplines or departments within a single organization; across organizations within a single geographical area; across geographies that are regulated by different legislation within a single country; or across the world.

A *federation* is a community of collaborating parties with different jurisdictions that cooperate by agreement to meet shared objectives. The key definitional characteristic of a federated community is that some decisions must be made explicitly in concert, rather than being made autonomously by participating parties. Communal decisions may be made by a central authority made up of members with delegated authority from their respective parties. Clearly, not all decisions need to be made communally, but a clear distinction of which decisions must be made centrally and which may be made locally needs to be explicit, especially those affecting the shared purpose.

2.2.1.4 *Party*

Party: "A party is an enterprise object modeling a natural person or any other entity considered to have some of the rights, powers and duties of a natural person. (Tyndale-Biscoe, Nov 2002)"

A party is a particular identifiable individual or organization that is expected to participate in one or more communities. A party may be described by its identity or by its general type. Defining participating parties by type requires a mechanism for identifiable parties wishing to participate to be able to express interest and be accepted by the interoperability community, either by consensus, or by meeting preset criteria.

Parties play more than one role and a single role can be played by more than one party. Participation in a community occurs via roles that specify the expected collaborating behavior. A party can participate in multiple communities at the same time, taking on different roles in each community.

2.2.1.5 *Jurisdiction*

Jurisdiction is the delineation of the boundary conditions of the scope of authority of a party. The boundary is determined by a geographical area and a subject matter or policy scope. Parties have jurisdiction within a particular scope of authority which may be delegated from another party with a higher authority. The relationships between jurisdictions may be implicit or may be codified in regulations or policy. An interoperating community has a jurisdiction of its own that is specified by contract of the agreeing participants.

2.2.1.6 *Contract*

A contract is a formal agreement among parties to behave in accordance with the policies and processes accepted by the community in which they participate. The contract clarifies the roles, responsibilities and policies required to act in concert to meet the shared objectives. A specialized community of parties may be formed to control the establishment and evolution of the contract. Participants of a federated community represented by the controlling community agree to the contract by actively participating. The very nature of interoperability is collaboration among parties who give up some autonomy of decision making within the scope of activities needed to achieve the shared purpose, but retain autonomy in other aspects of their endeavor.

2.2.1.7 *Authority*

Authority is the ability of a party to act autonomously. In many circumstances authority to act has been delegated according to particular policies. The party with the higher authority is a principal and the delegated party is an agent. Delegated authority from the principal party to the agent usually involves an expectation to be held accountable for the decisions and actions taken. Automated systems typically act as agents of responsible parties and carry out predetermined behaviors under specified conditions.

2.2.1.8 *Accountability*

Accountability is the obligation to take responsibility for actions and to demonstrate that actions are completed satisfactorily. The responsible party agrees to perform certain actions or to produce certain deliverables. Accountability means that some mechanism must exist for showing that accepted responsibilities are carried out and to what extent they are successful. Metrics or reporting mechanisms may become elements of interoperable systems demonstrating the shared objectives have been satisfied.

2.2.1.9 *Role*

A role is a collector for the behavior of a party needed to carry out its responsibilities according to a community contract. A specific name is given to the explicit set of responsibilities that identifies the competence of an organization, a person or an automated component acting as an agent, to perform specified actions. The set of responsibilities may include actions that have been delegated from a higher authority. Behavior is further refined into specific actions that may become operations in an automated system.

2.2.1.10 *Responsibility*

Responsibilities are explicit behaviors or actions associated with a community role. Responsibility for acting is stated as a permission (you may act), an obligation (you must act), or sometimes as a prohibition (you must not act), including the conditions under which each action is valid.

While a party in a particular role is expected to be competent to perform all specified actions or behaviors, some actions may have resource availability or other pre-requisite conditions to be met before they can be performed. The measure of a role's ability to act is considered to be the capability of a role. The amount of action due to resource availability is capacity. Resources can include space, equipment, supplies, specific information or simply time availability of a party in a particular role.

2.2.1.11 *Provenance*

Provenance is a term borrowed from the antiques industry. It referred to the documentation of what ownership a particular antique item has had over time. In the SAIF context, provenance refers to the documentation that identifies the jurisdiction of the source of each conformance statement (or the artifact containing a group of them) in a specification, from that statement's origination as documented requirements to implementable specifications for technical components. The history may be included within a specification or by reference to an external artifact.

Provenance may also refer to the auditable history of the context of information that originates in one system and is used in another, including any transformations that occur along the way. The term Provenance may also be used for other metrics to identify expected recording of actions taken for accountability purposes.

2.2.2 Governance Language

The Governance Framework language is made up of four interdependent concepts, which taken together define what the rules are, who makes the rules, what processes are needed to implement the rules and how the rules are measured or enforced. The following structure is based on that recommended by the book "*SOA Governance: Governing Shared Services On-Premise and In the Cloud* by Thomas Erl, Robert Laird and Robert Schneider.

Governance system design must consider all four together. A tabular structure is a convenient template, although actual documentation styles can vary considerably, as long as the specific concepts are linked.

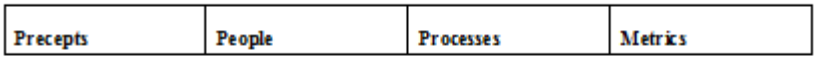


Figure 7 Governance design documentation template *(from Erl et al, 2011)*

2.2.2.1 *Precepts*

A precept is an authoritative rule of action. Precepts are the essence of governance because they determine who has authority to make decisions, establish constraints for those decisions, and prescribe consequences for non-compliance.

Precepts codify decision making rules using four "sub-dimensions" or "characteristics describing a given precept":

- **Objectives**, which broadly define a precept and establish its overarching responsibility, authority, and goals
- **Policies**, which define specific aspects of a precept and establish decision-making constraints and consequences in terms of permissions, prohibitions, obligations or authorizations
- **Standards**, which specify the mandatory formats, technologies, processes, actions, and metrics that people are required to use and carry out in order to implement one or more policies
- **Guidelines**, which are non-mandatory recommendations and best practices

2.2.2.2 *Processes*

A process is a collection of steps taking place in a prescribed manner and leading to an objective. A step may be associated with multiple roles. Every step shall have one or more actors.

It is important to make a distinction between governance processes and other types of processes. Governance processes provide a means to control decisions, enforce policies, and take corrective action in support of the governance system. Governance processes are further elaborated in the section below.

Other processes, such as those employed to carry out the intended purpose, can be heavily influenced by governance precepts, but are not specifically processes that are directly related to carrying out the governance system. The BF may be used to specify these additional processes. Technically, any process is considered a management activity, but a governance system is dependent on governance processes to ensure compliance with its precepts.

A community is likely to use a variety of processes to support its precepts. Some may be automated, while others require human effort. Automated processes can help coordinate tasks (such as steps required to collect data for approvals), but can still rely on people to make important decisions (such as making the actual approvals based on the presented data).

2.2.2.3 *People (Roles)*

People (and groups of people) make decisions in accordance with and within the constraints stipulated by governance precepts. For a governance system to be successful, people must understand the intents and purposes of the precepts and they must understand and accept the responsibilities and authorities established by the precepts. Governance systems are therefore often closely associated with an incentive system. This allows the community to foster a culture that supports and rewards good behavior, while also deterring and punishing poor behavior.

When exploring the involvement of people in relation to governance systems, it is further necessary to identify the role or roles they assume. Community roles position people (and groups) in relation to governance models and further affect the relevance of precept compliance and enforcement.

There are two ways that people can relate to precepts and processes: they can help author the precepts and processes and they can be dictated to by their application. Opportunity for those affected by the precepts to provide feedback to the authors is recommended.

Other entities can take on roles in specifications involving non-governance processes, but only people can participate in governing processes.

2.2.2.4 *Metrics*

Metrics provide information that can be used to measure and verify compliance with precepts.

The use of metrics increases visibility into the progress and effectiveness of the governance system. By analyzing metrics, we can gain insight into the efficacy of governance rules, and we can further discover whether particular policies or processes are too onerous or unreasonable. Metrics also measure trends, such as the number of violations and requests for waivers. A large number of waiver requests may indicate that a policy might not be appropriate or effective.

The ECCF describes specific types of metrics as conformance statements that are used to determine whether technology components can be certified to fulfill the behaviors specified.

2.2.3 Governance Processes

The processes to establish and maintain precepts and their related components are different from the processes defined within the context of each precept. The governance processes are all about what it takes to make the rules, communicate what the rules are to all interested parties, make exceptions to the rules and evaluate and change the rules when circumstances change or more effective rules are identified.

2.2.3.1 *Definition Processes*

The definition processes are those by which a precept is established, agreed to and then maintained as feedback on its use is provided. The workflow may include approval for establishing a new precept, authoring a definition and related components, approval for use, deployment into the environment of use, evaluation for relevance and efficacy as circumstances change, and subsequent ratification, revision, replacement, or retirement.

2.2.3.2 *Communication Processes*

Communication processes about precepts and their related processes and metrics are needed to inform the people expected to follow the processes. Various forms of communication channels may be necessary to raise awareness, clarify specifics, gain agreement and then hold people accountable. Awareness of risks and their consequences, rationale for selecting the specific precepts and their processes and metrics, and support for executing them may also be needed. Tools and other resources that minimize the effort required to comply will increase buy-in. Training for active participants in the processes is also likely to be necessary.

2.2.3.3 *Appeal Processes*

Appeal processes and transition strategies permit precepts to be overturned or modified by exception. Time-limited dispensations to do something other than what the precepts expect can ease transitions and avoid unnecessary disruption. However, the precepts are intended to reduce risk, and accepting appeals means a conscious decision to accept the increased risks.

2.2.3.4 *Revitalization*

Every precept and its related components should be evaluated periodically to determine if the related risks are being mitigated effectively, whether the precept is still relevant to the current circumstances, or whether there are possible alignments necessary among interdependent precepts to avoid gaps and confusion. Feedback from related metrics and appeals may be used, as well as evaluation of any rationale or assumptions identified when the precept was defined. New roles, technology opportunities or resource constraints may suggest a review of related precepts. In many ways, changes in circumstances require revisiting governance. Also, changes in governance may cause ripple effects in any automated application that is involved in precept execution.

2.2.4 Relationship between the Governance Framework and the Behavioral Framework

The Governance Framework provides the language for defining the specifics of the various organizational and technical development activities that must be defined, executed, and managed via overarching governance processes to reach agreement on a shared purpose and how to collectively achieve that purpose in the context of one or more defined cross-boundary scenarios. In contrast, the Behavioral Framework provides the language to describe the various contracts, transactions, and processes – at a technical level – which are necessary to produce a technical realization of previously specified shared purpose scenario. The languages defined by the GF and BF are similar in overarching motivation. However, each has a somewhat different focus and emphasis. Following are two lists the first which identifies terms defined in both the GF and BF but used in different contexts within the two languages, and the second listing terms mentioned in the GF but defined in the BF.

Terms defined in both GF and BF

- objectives
- policies
- contracts
- communities
- roles
- processes

Terms mentioned in GF but defined in BF

- operations
- obligations
- objects
- permissions
- prohibitions

3 Behavioral Framework

3.1 Purpose

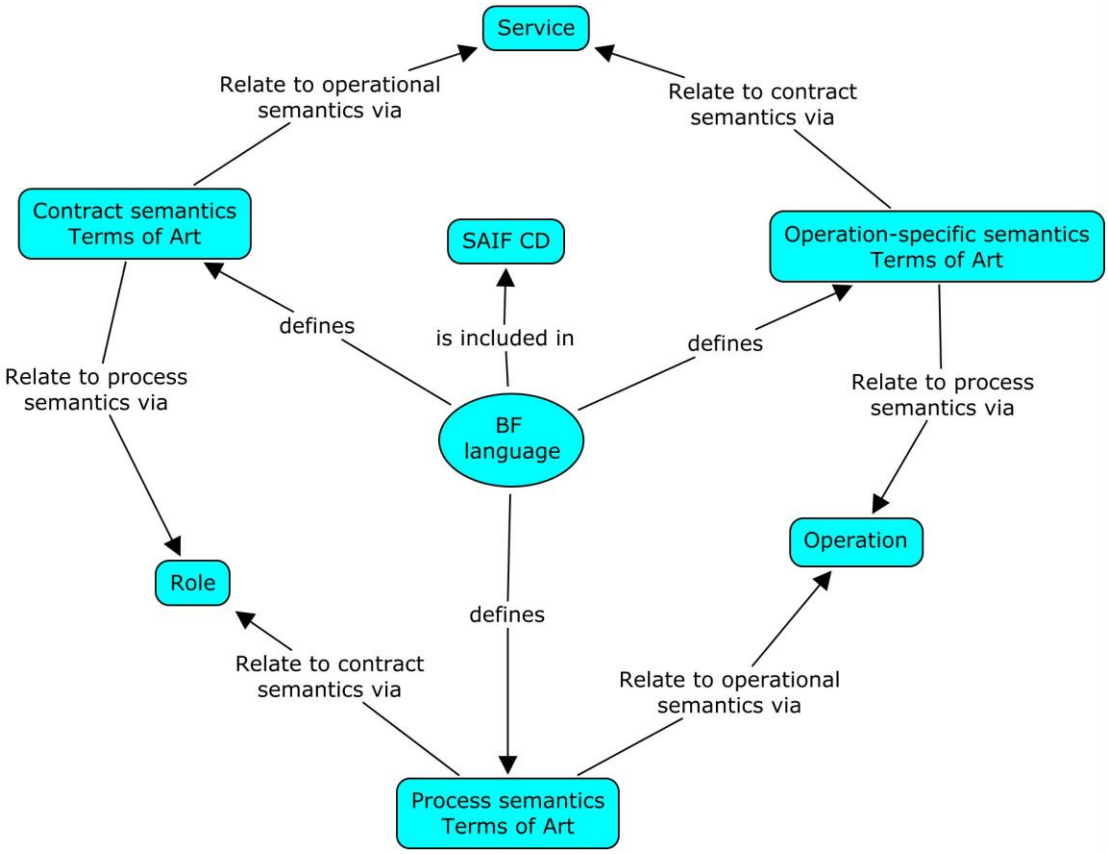


Figure 8 BF language concepts and relationships for describing contract semantics.

The purpose of the Behavioral Framework is to provide the language necessary to explicitly and unambiguously define *dynamic* semantics used to specify the *behavior* of enterprise objects involved in shared purpose scenarios. The BF language is meant to be used in combination with the IF language (which focuses on explicit expression of static/informational semantics) – to fully specify the details of the various roles, responsibilities, capabilities, expectations, accountabilities, etc. of a given object as it is involved in these scenarios. The BF semantics can be grouped together into three categories (see BF Overview Concept Map):

1. C*ontracts.* These semantics help to define enterprises as composed of objects (people, organizations, technical components, etc.) organized as communities with certain business objectives, leading them to create agreements called contracts in order to specify their behaviors. The fundamental unit used within the contracts to specify desired behavior is the service, organized following Martin Fowler's accountability analysis pattern, such that each service explicitly identifies the responsible and commissioning roles. [In particular, the Conceptual Perspective of the SAIF-CD, the BF language surrounding contracts serves – via the use of similar (and often identical) language – as a link between an organization's negotiated shared purpose and the technical realization of that shared purpose in technical architectures and their associated components.]

2. O*perations.* These semantics break down the details of the information exchange between the roles within a service, organized around the concept of a basic unit of exchange called operation. [The semantics of contracts are most often used at the Logical and Implementable Perspectives of the SAIF ISM to describe and define the architectural and technically implementable details of interactions – at the contract level – between individual components. However, operations – like contracts – have much of their original

semantics defined – or at least sketched – at the organization level in the larger context of business process (aka "workflow") and the semantics that organizations participating in shared purpose scenarios agree are required to achieve a given shared purpose.]

3. P*rocesses*. These semantics allow organizations to define complex interactions composed of multiple operations involving potentially many different services and roles.

The three categories of BF semantics do not exist in separate, mutually exclusive realms. Rather, the above categorization is primarily created as a cognitive aid in assimilating the BF language, and secondarily based on the source of the language (contracts and operations coming primarily from RM-ODP, and processes coming primarily from BPMN2). Overall, direct relationships between the concepts are more likely to exist within each category, with a small number of bridging relationships across the categories. In particular, the service concept acts as a bridge between contract and operation semantics, since service is the mechanism used to describe behavior in a contract, and operations are used to specify the details of the interactions within a service. Roles bridge contract and process semantics, since roles are what binds particular enterprise objects to their behavior within a contract, and roles also are used to specify the participants in a process. Finally, operations themselves act as the link between operation and process semantics, since the individual steps in a process which require interactivity between two roles are specified as particular operations of a service.

Shared purpose scenarios are often initially defined at an organizational level and then subsequently manifest at a technical level. The SAIF-CD recognizes this "problem space" vs. "solution space" topology through its use of Perspectives of the Interoperability Specification Matrix (ISM). In particular, the ISM's Conceptual Perspective represents the problem space view of a given component and is outward facing toward the larger issues of a given organization and its various shared purposes. As such, the BF language applied to the Conceptual Perspective usually focuses on the Enterprise Dimension. In contrast, the ISM's Implementable Perspective represents the solution space view of a technical component as a realization of the organization's shared purpose requirements. Finally, the IMS's Logical Perspective serves as the traceable bridge that links the problem space with the solution space. The concepts defined in the BF language in many cases will have distinct manifestations across the different perspectives, but the BF does not try to create separate concepts for each of the perspectives as this exercise will result in unnecessary redundancy at the canonical level. For example, an enterprise might need to specify a particular enterprise level contract defining business services between real world parties, and its corresponding technical contract to be realized in a particular implementable technical service. The SAIF-CD leaves it to the SAIF IG grammars to explicitly define the distinctions between services, contracts, roles, etc. across multiple perspectives and their correspondences.

The BF language is architecturally neutral in the sense that it allows component designers and developers to unambiguously discuss contracts, isolated operations, and amalgamated processes independent of their particular choices of implementation architectures, modeling constructs, etc. Thus, the BF language can productively be used to define the behavioral semantics of shared purpose scenarios involving any one of a number of interoperability paradigms including messages (e.g. as implemented using various flavors of HL7 messages), services (e.g. as modeled using SoaML or the OASIS SOA Reference Model and implemented using SOAP or REST technologies), or documents (e.g. modeled in HL7 CDA, openEHR archetypes, or 13606 containers). Modeling, design, and implementation paradigms such as these are specified in organization-specific SAIF-CD-compliant SAIF Implementation Guides (SAIF IGs)[1].

[1]The BF adopts and adapts RM-ODP (ISO RM-ODP) as a reference model. On one hand, the BF uses a small set of ODP modeling concepts which were found central for defining distributed components from the perspective of achieving shared purpose through interoperability. On another hand, the BF adds further level of detail such as a set of concepts from the BPMN2 metamodel to model processes. It also adds a small set of concepts to facilitate the distinction between conceptual and logical perspectives. The languages defined in the ODP and SAIF-CD are abstract and therefore require elaboration and instantiating in specific SAIF IGs, e.g. through the use of representational grammars such SoaML, UML 2.3, UML profile for ODP, etc.

NOTE: Even though the service concept is explicitly a fundamental one in the BF language (thus fulfilling the "service-aware" requirement of the SAIF), compliant SAIF IGs are not required to use a grammar that explicitly uses the "service" construct. What would be required is to organize behaviors around the fundamental accountability pattern that in the SAIF-CD is called a service. Furthermore, additional premises and best practices of service oriented architecture, such that services are created without limiting which particular objects are bound to commissioning roles, are not implicitly or explicitly required by SAIF-CD

.

3.2 Contract Semantics

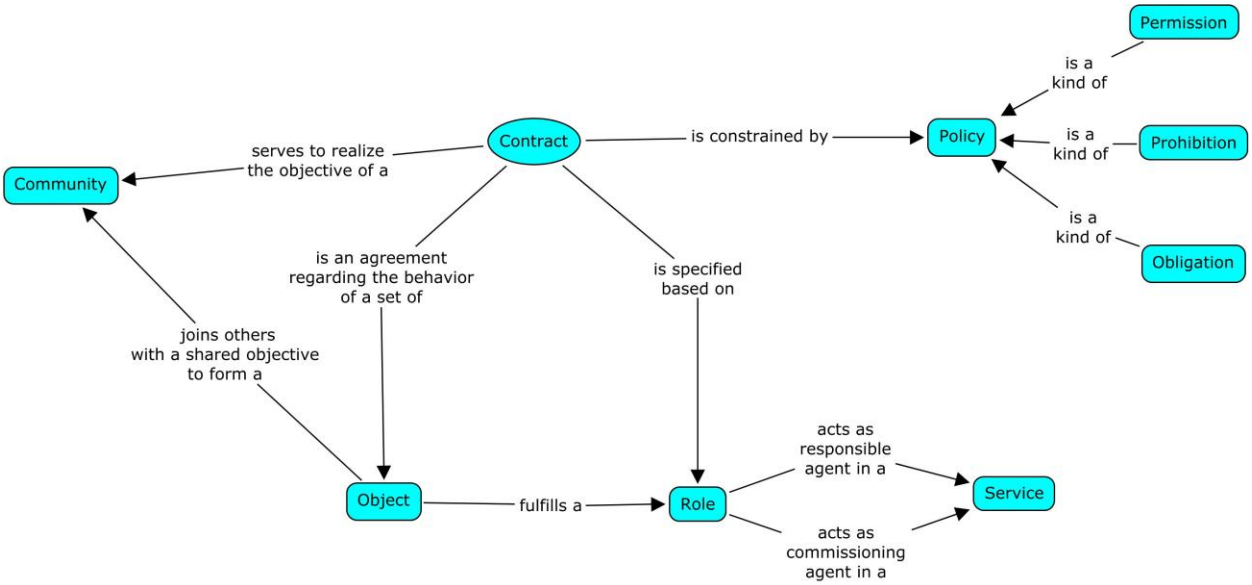


Figure 9 BF language concepts and relationships for describing contract semantics.

The BF contract semantics define the idea that enterprises are composed of **objects**, which could include either real world entities as well as IT systems. Objects are organized into **communities**, with objectives that include shared purposes requiring some degree of interoperability. In order to achieve these objectives, communities establish **contracts** between their objects specifying their behaviors. The ability to properly specify these behaviors in order to achieve interoperability is the main topic of the BF language. Agreed upon behaviors in a contract are organized along the abstract analysis pattern known as accountability [cite Fowler], which states that there is an agent responsible for the behavior and an agent that commissions the behavior. In BF contract semantics this accountability is known as a **service** and the contract allows each object to fulfill the **role** of commissioning or responsible agent for specific services. Contracts can be further constrained by **policies**, which can be in the form of **prohibitions**, **permissions**, and **obligations**.

The terms of art (in bold in the previous paragraph) defined by the BF language are taken primarily from the RM-ODP foundations (ISO, 2010) and enterprise language (Tyndale-Biscoe, Nov 2002). The concepts included from ODP were chosen because of their collective expressiveness in describing key organizational and policy concepts, in a way close to their natural language expressions.

The emphasis is not on supporting the description of social concepts such as acts, roles and entities for the purpose of recording information in a system–as such, these terms should not be viewed as synonymous with HL7 RIM terms (for example) – but more broadly to describe enterprise objects that will be involved in instances of shared purpose scenarios. Many of these concepts have analogues in the GF, a reflection of the fact that the shared purpose semantics that are ultimately expressed at the technical component level via component-to-component interoperability are initially determined at an organizational level. In general, readers of the SAIF-CD can view the

GF as outward facing, i.e. directed toward the problem space, whereas the BF is more inward facing, i.e. directed toward the solution space. These are not absolute constraints. What follows is a detailed set of definitions for these terms.

Contract: An agreement governing part of the collective behavior of a set of objects. A contract specifies, for each object involved, the different roles they may or must assume. Contracts may also specify policies for the objects, quality of service requirements, indications of duration or periods of validity, behavior which invalidate the contract, liveness (OWlCKI, 1982) and safety conditions.

Object: A model of an entity (entity is defined as any concrete or abstract thing of interest). An object is characterized by its behavior and its state. Objects are the subjects of a contract and fulfill particular roles in services and processes. Note that the concept of object is broader than the traditional notion of software objects or business objects used in building object-oriented and enterprise system. It is a model of any entity.

Community: A configuration of objects formed to meet an objective. This objective is expressed in a contract.

Role: Identifier for a behavior, which is to be fulfilled by an object as part of a contract. Specifically, the BF requires each role to be associated with a service either as a commissioning or a responsible agent. Roles are also the identified participants in a process.

Service: A related set of behaviors that add value by creating, modifying, and/or consuming information, involving collaborations between a responsible agent (the service provider), who expresses some guarantees, and commissioning agent (the service user or consumer), who receives the guarantees. The collaborations may involve a complex series of interactions, organized along operations. In a contract, roles fulfilled by particular objects identify who act as the responsible and commissioning agents.

Policy: A set of rules applied to a particular purpose. Policies are included in contracts, but may also be applied to many other objects or concepts in any of the dimensions.

Obligation: A prescription that a particular behavior is required. An obligation is fulfilled by the occurrence of the prescribed behavior.

Permission: A prescription that a particular behavior is allowed to occur. A permission is equivalent to there being no obligation for the behavior not to occur.

Prohibition: A prescription that a particular behavior must not occur. A prohibition is equivalent to there being an obligation for the behavior not to occur.

Note: A specific grammar instantiation of the BF language can provide a specific way of defining structuring, behavior and policy aspects of the community (for example, the use of the OMG SBVR notation), add further level of detail to the concept of objective (for example, the use of OMG Business Motivation Model) and so on.

3.3 Operation Semantics

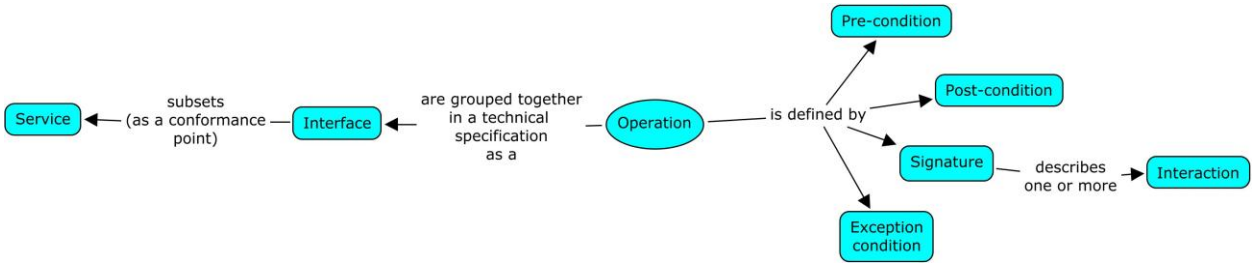


Figure 10 BF language concepts and relationships for describing operation semantics.

The BF operation semantics provide a way to specify and organize the information exchanges required for interoperability, specifically the exchanges between the responsible and commissioning roles of a service. The basic meaningful unit of information exchange is the **operation**, which may necessitate one of more **interactions** .As an illustrative example, a laboratory results service might include an operation to retrieve a result given a patient and accession number. This particular operation might involve two interactions, the query from the commissioning role including the patient and accession number parameters, and the result answer back from the responsible role. Operations, in some HL7 contexts have also been called "transactions," but SAIF-CD prefers the RM-ODP term because "transactions" in a different context (i.e., database systems) imply specific ACID conditions, including ability to rollback, that are not meant to be part of this concept. An operation is fully described by its **signature** (which specifies the interactions involved), pre-conditions, **post-conditions**, and **exception conditions**. Each service provides one or more operations, grouped together into **interfaces**, which define a specified subset of the total set of operations in a service. This subset serves as a conformance point in specifications.

The terms of art (in bold in the previous paragraph) defined by the BF language are taken primarily from the RM-ODP computational language (ISO RM-ODP). In RM-ODP operation is a special kind of interaction, the others being signals and streams. SAIF-CD maintains the simplicity of a single construct (operation) as the basic unit of defined behavior, allowing the SAIF IG grammars to specify more varieties based on the needs of the particular enterprise. The following are the definitions of the concepts introduced by BF operation semantics:

Operation: The smallest unit of behavior, involving information exchange between commissioning and responsible roles in a service, which provides business value. Operations are specified by their signature, pre- and post-conditions, and exception conditions.

Signature: The precise definition of the interactions involved in an operation, including attributes such as direction, optionality, and content.

Interaction: An atomic piece of information that is transmitted in one direction from an object to another. One or more interactions must exist together in the context of an operation for there to be business value as part of the information exchange. A single interaction that is part of a larger operation provides no business value in isolation, for example, a query without a response.

Pre-Condition: a predicate that a specification requires to be true for an operation to occur.

Post-Condition: a predicate that a specification requires to be true immediately after the occurrence of an operation.

Exception Condition: exists when an operation fails to fulfill its service guarantees

Interface: A grouping of operations of a service required to be implemented together in a specification.

3.4 Process Semantics

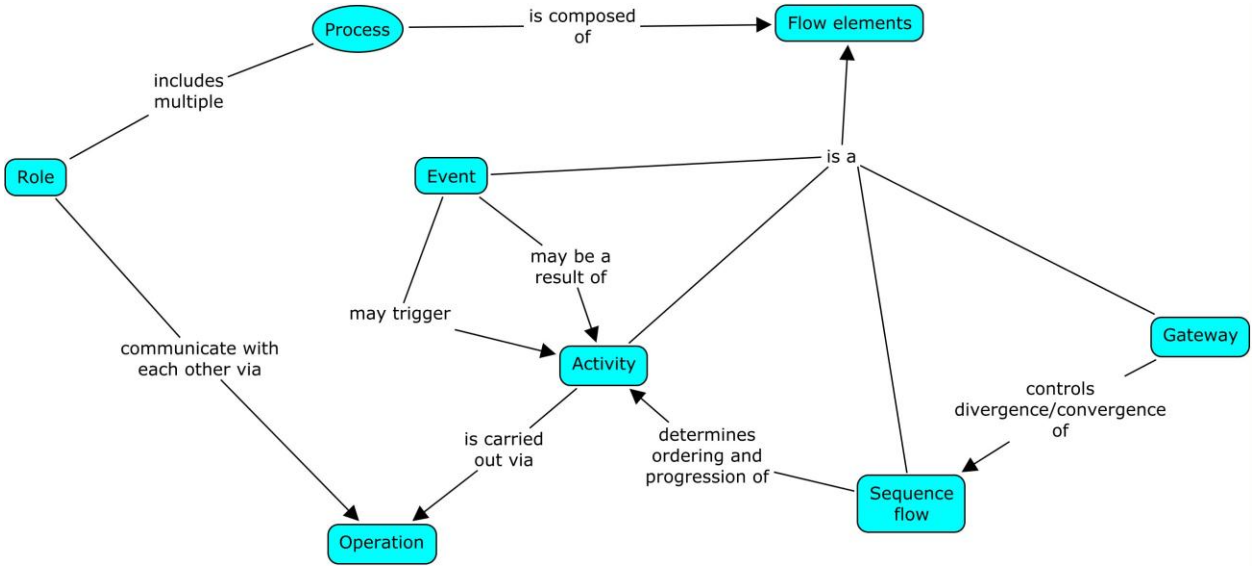


Figure 11 BF language concepts and relationships for describing process semantics.

The BF process semantics allow for complex behaviors known as **processes**, which potentially include many different service operations in a sequence, involving multiple participants defined as roles. The sequencing and relationships between the multiple behaviors of a process are described using a set of **flow elements**, which usually correspond to elements of a particular notation. Although the key concepts in BF process semantics come from the BPMN2 metamodel, the full BPMN notation would be considered a grammar, and its use, if desired, would be specified by the SAIF IGs. The concepts used in the BF language are abstract enough such that a particular SAIF IG may choose grammars other than BPMN and still be SAIF-CD compliant. The main flow elements of the process, specifying the action steps, are **activities**, which are carried out via service operations when they require information exchange between process roles. **Sequence flows** are flow elements that determine the sequencing of activities in a process. Events are flow elements that represent triggers or results of activities. Another flow element is the **gateway**, which serves to organize options and parallelism in sequence.

Process: A collection of steps (defined as activities) taking place in a prescribed manner and leading to an objective Contracts may specify the participants involved as roles in the process, corresponding to the roles in all the services for which operations may be invoked over the course of the process.

Flow elements: The units used to describe the process and its sequence of steps. In a SAIF IG grammar, the flow elements usually correspond to elements in a particular process description notation.

Activity: A process flow element that represents a step of work to be performed. An activity can be composed of further smaller activities, and described as a sub-process (SAIF IG grammars will determine precisely how this decomposition is to be expressed). Any information exchange that is necessary for an activity must be explicitly carried out as a service operation.

Event: A process flow element that represents some kind of occurrence ("something" that happens), which in turn causes an activity to occur (a trigger) and/or occurs as a consequence of an activity (a result).

Sequence flow: A process flow element that determines the ordering and progression of activities in a process. Typically, a process notation specified in a SAIF IG might denote sequence flows as lines and arrows connecting the activities.

Gateway: A process flow element that controls the divergence and/or convergence of sequence flows. It allows branching, forking, merging, and joining of process flow.

4 Information Framework (IF)

4.1 Purpose

The Information Framework chapter defines the language describing the various artifact types and inter-relationships of the Informational Viewpoint from the three SAIF Perspectives. The concept map below provides an overview of the IF language.

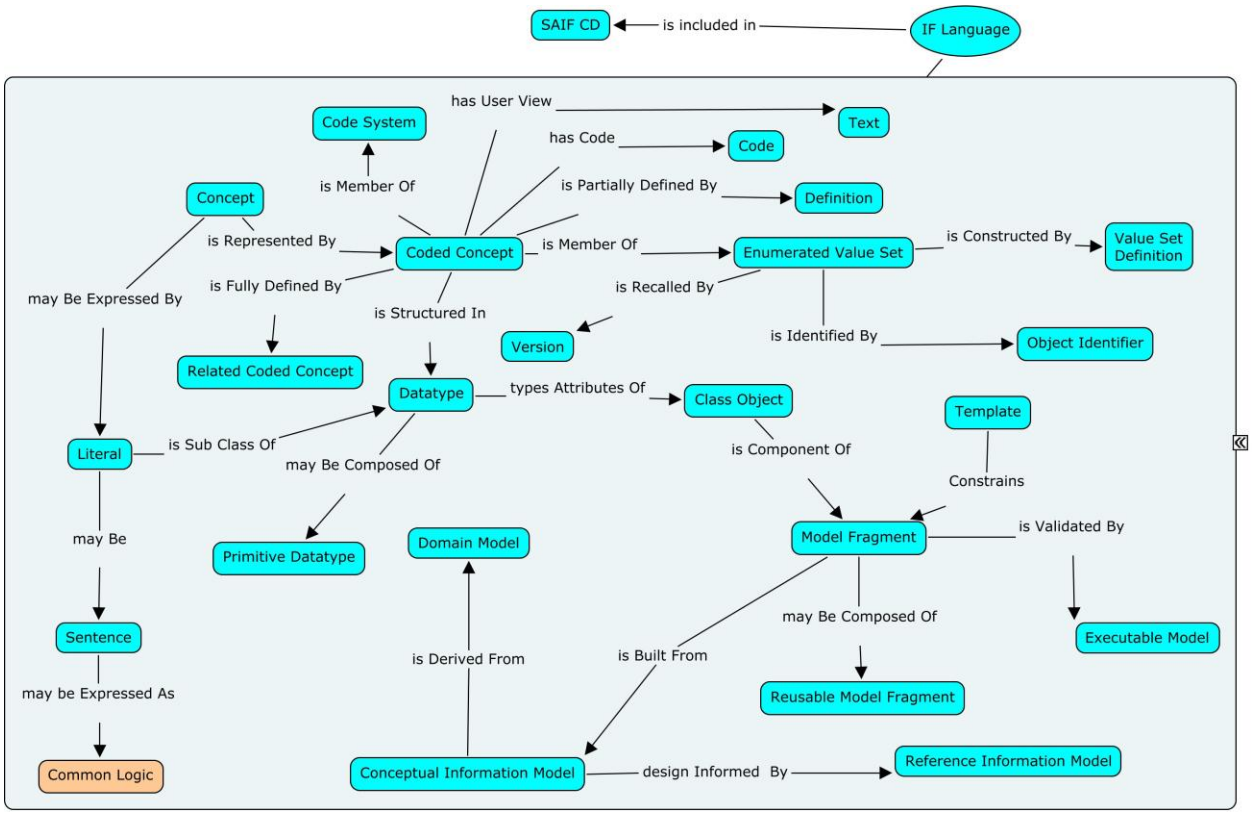


Figure 12 Information Framework Concept map

4.2 Goals

The goal of the information framework is to describe how the static information of importance to a given domain and the experts within that domain is captured and refined through a traceable process to yield an implemented or implementable information artifact. This implementable information artifact, when developed using the methods defined in this framework, delivers the static semantics that contribute to the definition of computable semantic interoperability between systems. The information definitions contained in these artifacts are reusable, and given the appropriate level of enterprise governance in the process of model development, yield consistency across the range of information modeling tasks encountered within an organization.

4.3 Data and Information

Data is the raw material from which information is derived. In order to allow information systems to use data to address most healthcare use cases, we must first convert it to information.

A simple natural example gives us a basic understanding of this conversion process. For example, let's take images. Light is transmitted through the lens of the eye and focused onto the fovea of the retina where rods and cones transmit the photons of light energy to the visual cortex of the brain, interpreting and preserving color and contrast.

The light is processed, its intensity determined, the directionality from the source is noted and the light with context is integrated with the visual context and referenced against other historical information stored in the brain. All of this data is put into context and thus can be used as information to interpret the raw photons and to assess the light as an image, either of beauty, threat, unclassified wonder, etc.

The parallel information technology process is the capture of a digital image through the lens of a camera. In this case, the photons are focused by the camera lens onto a sensor. The sensor stabilizes the image, activates specific chromatic sensors to determine color, and passes the information to a processor to generate the image in one of a number of possible mime types. Thanks to the standardization of the processing and use of standard mime types, these images can then be used by a variety of applications for a variety of purposes with no loss of information (this is dependent on the mime type used since some are lossy).

Streaming data across an enterprise is no more useful than streaming photons without the processing enabled by the rods and cones of the retina or the processor in a digital camera. There must be context provided so that the data can be used as information for a useful purpose, or rather, a meaningful use in today's healthcare parlance.

We therefore can say that information is "data in context". Hence the SAIF Information Framework Book is about putting data into a context that information systems can properly manage and apply data for useful purposes. It is the context of data and its unambiguous organization into a hierarchy of information models that provides the properties of semantic interoperability when shared with other information systems. The more a system adheres to the SAIF principles, the more interoperable that system will be with a wide range of other systems that also apply the SAIF principles.

This document is meant to lay out those principles in their canonical form so that these principles may be used across a wide range of implementations and hence is agnostic to the eventual implementation language or model persistence.

Information Framework Components

i. Concepts and concept organization
    - Un-encoded concepts
ii. Datatypes
iii. Class objects
    - Terminology binding
iv. Information Models
    - Templates
    - Executable Models
    - Conceptual Information models
    - Domain models
    - Logical Information Models
v. Summary

4.4 Concept Component

A concept is the basic unit of data used in communication and each concept represents an atomic unit of thought that references a concrete or abstract thing. Concepts are organized into terminologies and these terminologies have specific models that define how the concept metadata is described and what, if any, rules can be applied to the concepts to create more complex concepts out of simpler concepts. The simpler concept is called a primitive concept and the more complex concepts formed by the combination of two or more concepts are called pre-coordinated concepts. This allows a more precise definition of a concept that improves the chances of semantic interoperability between partners.

Primitive Concept	Pre-coordinated Concept
	
Pneumonia	Right lower lobe Streptococcal pneumonia
233604007	233604007 |pneumonia| : 246075003 | causative agent | = 9861002 | Streptococcus pneumoniae |, 363698007 | finding site | = 266005 | structure of right lower lobe of lung |

Figure 13 Example of concepts

4.5 Controlled Terminology

The purpose of a terminology is to provide a clear and unambiguous way to describe concepts so that two or more individuals can gain a shared meaning of those concepts. A concept is the basic unit of communication and each concept represents an atomic unit of thought that references a concrete or abstract thing. A controlled terminology provides the organizational framework for concept ordering, inheritance and rules that govern the use of the concepts. For example, Jim Cimino described several rules that a sound controlled terminology should adhere to. These include vocabulary content, concept orientation, concept permanence, non-semantic concept identifiers, poly-hierarchy, formal definitions, rejection of "not elsewhere classified" terms, multiple granularities, multiple consistent views, context representation, graceful evolution, and recognized redundancy {Cimino, 1998 #94}. (NOTE: The degree to which a given SAIF IG may require these particular attributes in terms of bindings to terminologies is, in fact, an IG-specific decision. The concept of Controlled Terminology is part of the SAIF-CD descriptive language for specifying informational/static semantics.)

The concepts can be expressed in a number of ways. Common expressions of a concept may be verbal, symbolic, textual or coded. Once a concept expression is agreed upon it can be used for the purpose of interacting with trading partners that need to share information.

In verbal communication of these terminological concepts, the spoken language must be known by the communicating parties as well as the dialect and inflection in some cases. Often times those terminological concepts may have multiple meanings depending on the context in which they are used, even when the spelling in a given language is identical. Therefore, the textual representation of a concept is inadequate to completely provide the meaning of a term when it is separated from its context of use.

Information systems depend on an explicit and unique meaning of a concept and hence cannot rely on verbal or textual representations of concepts. Textual representations may be misspelled, abbreviated, or expressed in a different language with different spellings as the example below shows.

	
Concept	Streptococcal Pneumonia
Alternate spelling	neumonia
Abbreviation	S Pneumonia
Misspelling	Streptococcal pneumonia

Figure 14 Example of alternative text for a concept

Concepts must be encoded with unique identifiers in order to disambiguate identical textual or verbal representations of different concepts. These encodings must be unique within a given code system or namespace. There is no guarantee that the code value is unique across other terminology namespaces and in fact there are many instances where the coded representation of a concept is reused across different terminology namespaces. The table below shows a small part of the 921 LOINC and CDC Race and Ethnicity codes that overlap. Without knowing (and sending) the code system with the code, there is risk that ambiguity will exist once the data is subject to query.

LOINC NAME	LOINC Code	Race Code	Race Name
HCG Ur Ql	2106-3	2106-3	White
HCO3 BldA-sCnc	1960-4	1960-4	Tununak
HDLc SerPl-mCnc	2085-9	2085-9	Micronesian
Insulin 2H p 75 g Glc PO SerPl-mCnc	1564-4	1564-4	Scott Valley
Insulin 3H p 75 g Glc PO SerPl-mCnc	1567-7	1567-7	Big Cypress

Figure 15 Concept overlap

Coded concepts are used as a) structural vocabulary or b) descriptive vocabulary. Structural vocabulary is used to describe the model elements that carry the descriptive vocabulary which is used at the instance data of a model.

Finally, vocabulary can be divided into those terms used in the “model of meaning” and those used in the “model of use" as described by Rector(Rector, Rogers et al. 2004). The model of meaning is that model supplied by the definitional structure of the controlled terminology that defines the concepts through either formal definition (description logic for instance) or informal definitions in text including the fully specified names. The model of use describes how a terminology is actually deployed in an electronic health record or other application that includes the grouping into pick lists or value sets, the ordering of the concept presentation, and the display names of those concepts.

4.6 Un-encoded concepts

Not all concepts received in messages or received as service payloads will be encoded in a specific terminology. In many cases the concepts will be included as literals, i.e. not bound to any specific terminology or code system. These are often referred to as “free text” entries. There are several ways to process these entries including natural language parsing, storage as native text entries or conversion to lingual interpretations that can be machine processed.

One of the methods of taking free text entries and converting them to machine process-able data entries is via the ISO 24707 Common Logic specification. While literals can be converted to machine process-able data entries, the process requires an understanding of first order logic.

Common Logic Controlled English Entry: **John goes to Boston by bus.** This entry is called a "sentence" in common logic.

This sentence may be expressed in a machine interpretable format via common logic in the following graphic.

Conceptual graph display form:

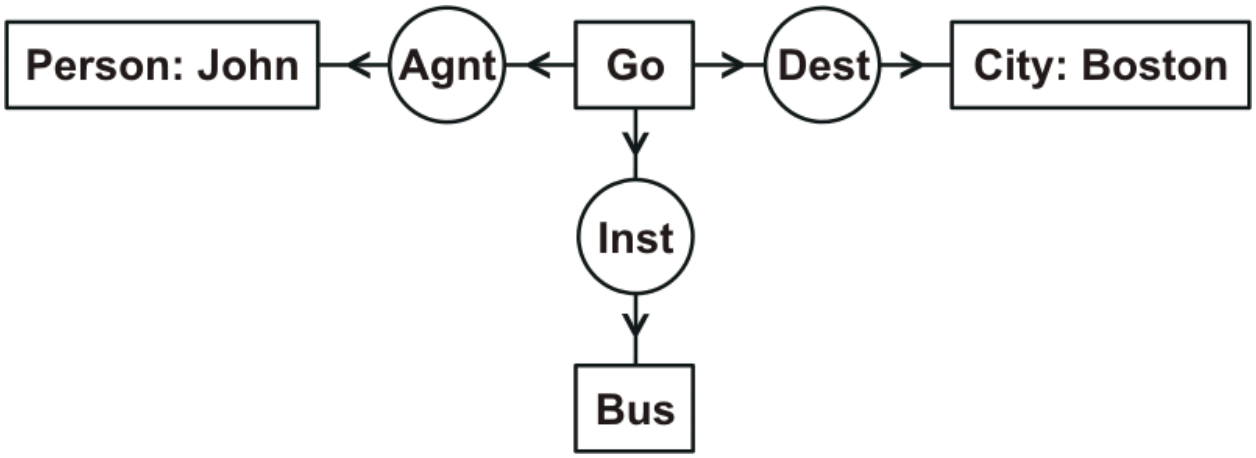


Figure 16 Conceptual Graph display Form

Conceptual Graph Interchange Format (CGIF):

[Go *x] [Person John] [City Boston] [Bus *y]

(Agnt ?x John) (Dest ?x Boston) (Inst ?x ?y)

Common Logic Interchange Format (CLIF):

(exists ((x Go) (y Bus))

(and (Person John) (City Boston)

(Agnt x John) (Dest x Boston) (Inst x y)))

This syntax is not familiar to most developers and hence is included here as a mechanism for further study of ways to construct logic statements to handle free text or literal entries.

4.7 Concept Grouping

4.7.1 Code Systems

There are several ways to organize concepts for models of use. The collection of all concepts in a particular terminology is called a coding system or more simply, a code system. Some code systems contain only the concepts that describe like or similar concepts. This set of "similar concepts" is referred to as a "semantic type". Examples of

code systems that contain concepts of a single semantic type include the CDC Vaccines Administered code system (CVX) and the Standard Occupational Codes (SOC) code system that defines occupational categories. Other code systems have many semantic types defined in non-overlapping subdivisions, the prime example being SNOMED CT where top level categories include products and geographical locations as well as clinical findings or procedures.

4.7.2 Semantic Types

The semantic type is a category for an item or group of items (concepts in our case) that all share a similar meaning (semantics) as defined for that group. The semantic type can then be used to distinguish the use and purpose of different items in the group. Examples of semantic types taken from the National Library of Medicine's Unified Medical Language System (UMLS) include virus, fungus, laboratory test and professional society, all placed into a hierarchical structure. It is common to refer to a reference set of semantic types as fillers for an attribute of the abstract information models such as Conceptual Information Models. In this case it is inappropriate to define specific codes or code systems from which these semantic types might originate so that the Conceptual Information Model maintains maximal reuse capability and subject matter expert familiarity. Being able to refer to a semantic type as the appropriate concept group for an attribute allows a domain expert to provide requirements in their language and allows a terminologist downstream in the development process to assign appropriate code System content to that abstract semantic type.

4.7.3 Value Sets

Typically a set of concepts are organized into a group that can be used as fillers for a field in a data entry form. The set of concepts used for this purpose is referred to as a value set. A value set need not draw all of its member concepts from a single code system. The life of a coded concept does not end when the submit button is depressed and the data element is stored in the database. The data will almost always have a secondary use and in order to use that data appropriately, it must be stored with the appropriate metadata to understand the coded concept in context. This will include enough metadata to resolve the exact value set membership at a given point in time, namely at the time the user submitted the data. This means that a value set member must be stored with the date of the value set creation and some unique identifier for the value set. When this value set is ordered in a particular way for optimal use in an interface, it is often called a pick list. There is psychometric evidence that the ordering of a concept in a pick list is important in evaluation of data input and this metadata may be optionally stored as well {Sudman S, 1996 #257}. This attention to value set membership is necessary to enable valid longitudinal analysis of data. Without this metadata it would be impossible to know what coded concepts a user could have chosen from as a response in a form field, hence data would not be comparable over time as the choices could have been changed by addition or deletion.

4.8 Data Type

A data type is a data storage model or template that defines the attributes for a specific type of value or range of values. It acts to formalize the requirements for data of specific types so that all of the attributes needed to process the data are known by a receiver.

Data types may be simple where the attributes of the data type each hold only a single data value (primitive types) or they may be complex where the attributes may hold a pointer to other data types that hold the actual data values. The more complex data types may also have a mechanism to define constraints on the data type so that an abbreviated set of attributes may be sent and a processor can still validate the contents of the constrained type without requiring all attributes to be populated. In this way a single data type definition can satisfy multiple use cases. This constrained data type is called a data type flavor.

Data types can be grouped into a set of canonical types. The canonical data types are classified as nominal, ordinal, quantitative, narrative text or image mime types. Nominal types express a categorical response that does not have a natural ordering. This includes names of entities or simple observations of natural phenomenon such as color or consistency for example. Ordinal values express concepts that have a natural order. Examples of ordinal values include grades such as A-F and sizes such as small, medium and large. Quantitative types include numerical values

expressed as ratios, integers, real numbers or ranges that have a mathematical interpretation. Narrative text data types are used to express descriptions in natural language. Finally, there are types of information that are typically symbolic to human interpretation but may be processed by machines as digital data. Examples are radiology images, digital wave forms and gel electrophoresis patterns.

4.9 Classes

A class is a collection of attributes that pertain to a specific encapsulated concept. Note that this definition includes UML classes, OWL classes, and other more loosely defined things such as SNOMED-CT concepts. For example a person can be described by a set of attributes that are always reflective of fixed properties of a human being. The properties include a date of birth, a genetically determined gender, a race to which the person belongs and an ethnicity that reflects an ancestral population group. Attributes have properties that control their use and possible values including their type and are collected into an information structure called a class that can be used as a component of larger information models. Classes have relationships to other classes and relationships have properties of their own such as whether they are monotonic (1:1) or open ended such as 1: many or 0: many. The data elements of a class - attributes and relationships - may be formally defined in the context of a framework such as ISO 11179.

Classes are defined within the context of an information model (see below) that provides the context in which they are understood and used.

4.10 Terminology binding

Attributes of a class can be coupled with the set of concepts used to describe the possible values of that attribute. This identification of the concept fillers for a given attribute in a given class is called terminology binding. The binding at the class level is broad and can usually best be done with a semantic type rather than a value set until such time that the class is used incorporated as a component of a specific information model that is to be used for a specific data purpose in a specific domain. For example, I could have a laboratory class with a result value attribute. When the class is unbound to a specific information model, we can only say that the terminology for that attribute will come from some data set that can express a lab value. That data set might be an ordinal type, a narrative type or a nominal value for example. If I now include my class in a specific information model where I know the only result values that I will get are blood types, I can bind the attribute to a specific value set that contains all of the human blood types and no other values are possible.

4.11 Information Models

Information models represent a collection of classes and the relationships between those classes. The relationships may be classes themselves in more complex modeling methods and are reflective of a specific domain of discussion. In other words, the relationships between classes are not static from information model to information model and change depending on what behavior (or larger concept) the model is expressing. Information models for a given domain may be subdivided into small, reusable sub-models. This is a useful way to provide consistency of class relationships that are common across information models. An example would be the physical address class relation to an entity class which is always a static relationship since a physical entity always occupies some physical location. There are many examples of the small, reusable models in healthcare modeling.

Information models may be UML class or instance diagrams, constraint statements on some other model, ontologies, or terminology models. Information models may be expressed against many underlying definitional frameworks, or none at all (e.g. concept map); which is appropriate depending on the use to which the model will be put.

Information models may be concrete where they define a specific set of classes with specific relations and specific terminology bindings or they may be abstract where the classes have optionality to the classes they are related to and the terminology is not set by bindings of specific values. These abstract models can be used to define information requirements from which more specific constrained information models are derived.

Useful information models are internally consistent in several senses, including their semantics and their engineering methodology; building these models is challenging. Several different methods may be used to build such models. The classic method is specialization of a class where the parent class has only the necessary and sufficient attributes to define that parent and the children classes add attributes to define specialization of the parent class. This approach favors implementation consistency over semantic consistency. An alternative is to constrain an abstract parent class that contains a superset of all attributes of a class type. This approach favors semantic consistency over implementation consistency.

Below are two examples of demographic information models. The first example is the Person archetype of the Demographic Information Model from openEHR.

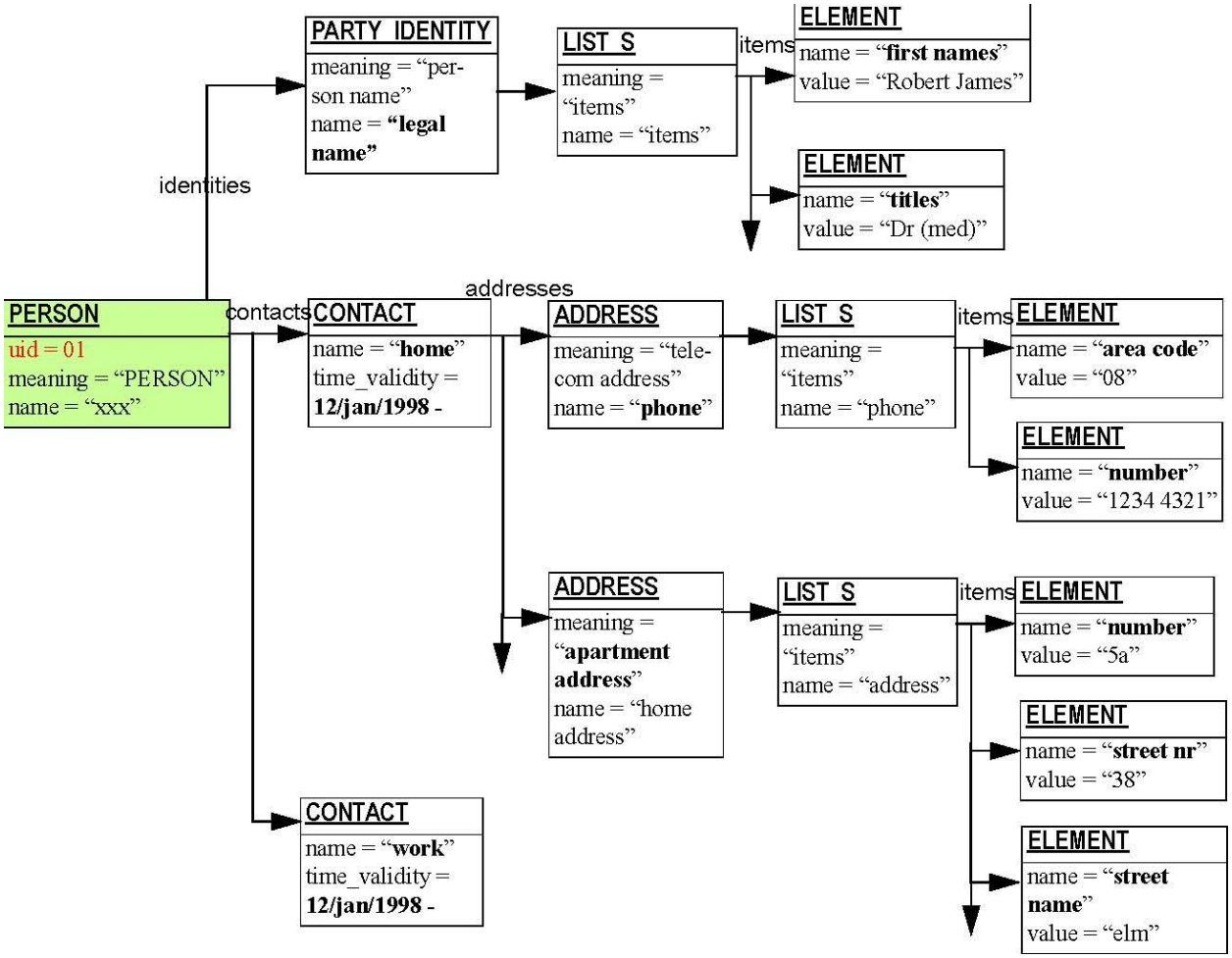


Figure 17 openEHR Person Demographic Information Example© (openEHR Foundation, 2001-2007) -

Below is the second example, which is the E_Person universal (COCT_RM030200UV08) CMET from (Health Level Seven International, Inc., 2011).

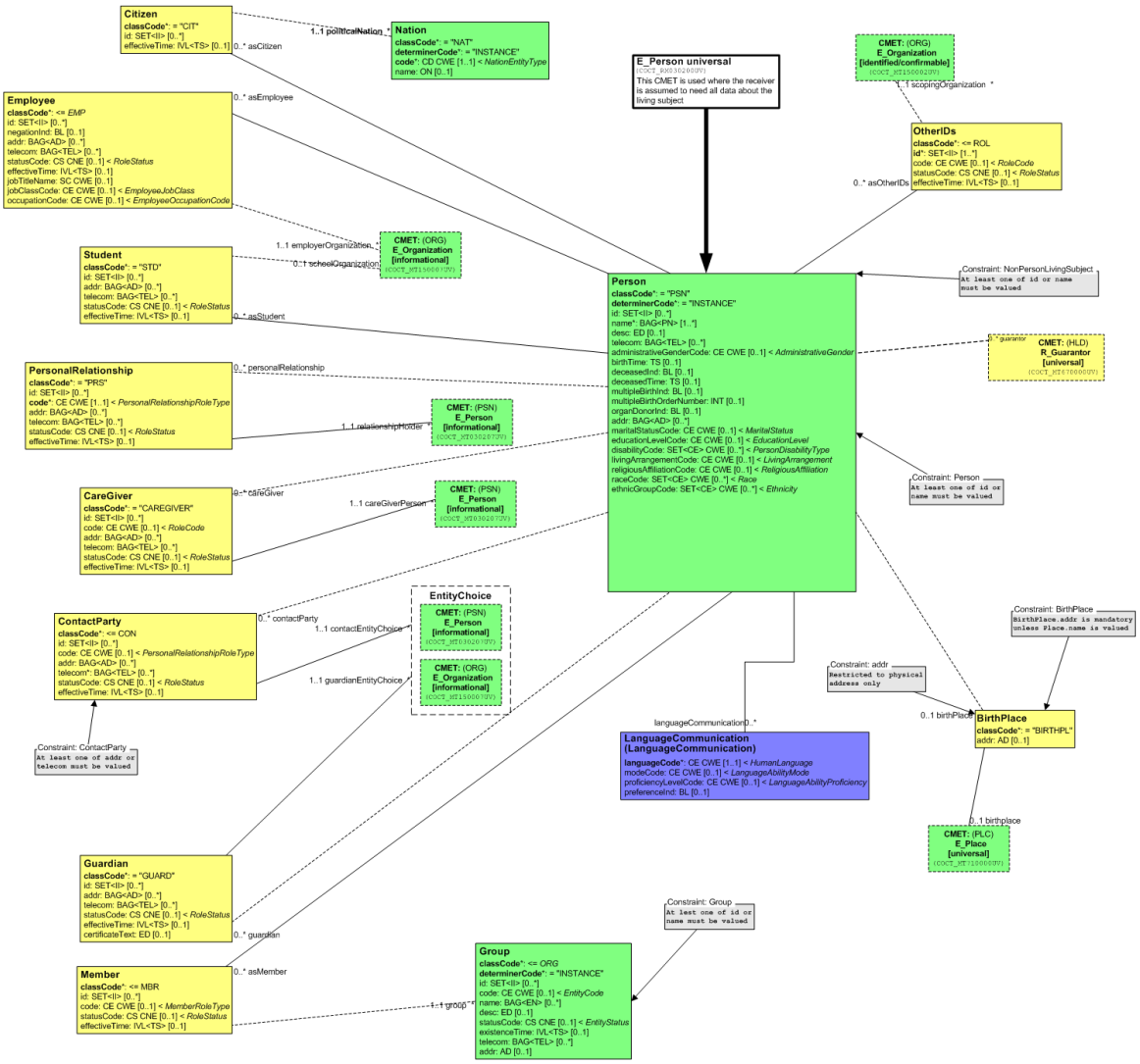


Figure 18 E_Person universal (COCT_RM030200UV08) CMET

Building such models consistently is a challenge. Adding attributes to classes based on an ad-hoc empiric analysis of a particular domain of discourse is fraught with inconsistency, incompleteness and intense effort and is unlikely to lead to semantically interoperable models (e.g. modeling domains based on ISO 11179 alone with no additional methodology). This is because there is no overarching information model to guide the developers of these "common data elements" in a consistent way and hence each model may be developed via the understanding of a different observer rather than via a guiding information model of the domain. The forms of models described below (Conceptual Information Model and Reference Information model) introduce consistency across the information models and lets one construct a Logical Information Model that is faithful to the business requirements and to the reference information model.

4.11.1 Reference Information Model

A reference information model is a formal model of an entire domain of discourse. It serves as a guide or pattern for all derived concrete classes of a domain or sub-domain of interest. A reference information model is essential to the development of a consistent representation of specific information models in a domain of discourse. It allows for the interpretation of relationships of sub-domains to each other, helps us understand the relationships between artifacts in an information model derived from the reference information model, and allows for the consistent definition of information artifacts and therefore consistent use. It helps to avoid the "re-invention of the wheel", such as multiple different interpretations of the same concepts in different contexts, by providing a framework that leads a modeler

down a well-worn path. Applications may be able to leverage the underlying reference information model to help can share data in a well encapsulated framework.

4.11.2 Domain Information Model

Domain models express the full information model and relationships that exist in a specific realm of knowledge in the business language of the domain itself. This might be a realm such as cancer care or infectious disease surveillance. It is domain specific and does not try to express every contact or peripheral information modeling for related but distinct domains of knowledge.

4.11.3 Bridging between the Domain and the reference model

These two models – the domain model and the reference model – are related in that the expression of the domain model in terms of the reference model provides a stable, robust construct that is suitable for use in interoperability. A bridge must be built to traverse between these two models. Building this bridge is an iterative manual process. The bridging process leads to a model that is called the "Conceptual Information Model" – this is the model from which the actual interoperability specifications are derived.

4.11.4 Logical Information Model

A Logical Information Model is an information artifact that provides a level of granularity such that the model may be directly consumed by a developer to build one or more implementation specific artifacts. The logical model is informed by both the conceptual model and the reference model. All classes and attributes are defined and the terminology to be used in implementations has been identified at a level of value domains, but not yet constrained to a point that all values would be used in any specific implementation.

4.12 Templates

A template describes a pattern of use of a model fragment. It is a statement of restrictions on the attribute value domains, cardinality and optionality of the information model when it is applied to a particular use case or context. Templates often provide additional definition and documentary material that describe how the information models are applied to very specific use cases or contexts. This material needs to be consistent with the underlying model fragments to which it applies. Templates may be broken down into reusable modules.

4.13 Executable Models

In order to assist implementation, it is useful to provide executable forms of the models. In these models, the information model is represented in a form that can be interpreted by other software that can perform useful functions such as validate instances or generate code. Examples are W3C XML schema, schematron, etc.; many forms exist. These executable forms are frequently incomplete representations, limited to what the software and/or specifications are capable of doing.

4.14 Summary

Through this canonical information framework, the static information artifacts that serve to provide semantic interoperability between trading partners has been described.

It is crucial to realize how each artifact provides additional context to enhance the semantics of its more primitive related artifact. It is this additional semantic layering that allows the progressive levels of interoperability that allows greater understanding of the information at each level.

The diagram below shows how each artifact wraps context around its related artifact.

At each level, a declaration of interoperability capability can be made.

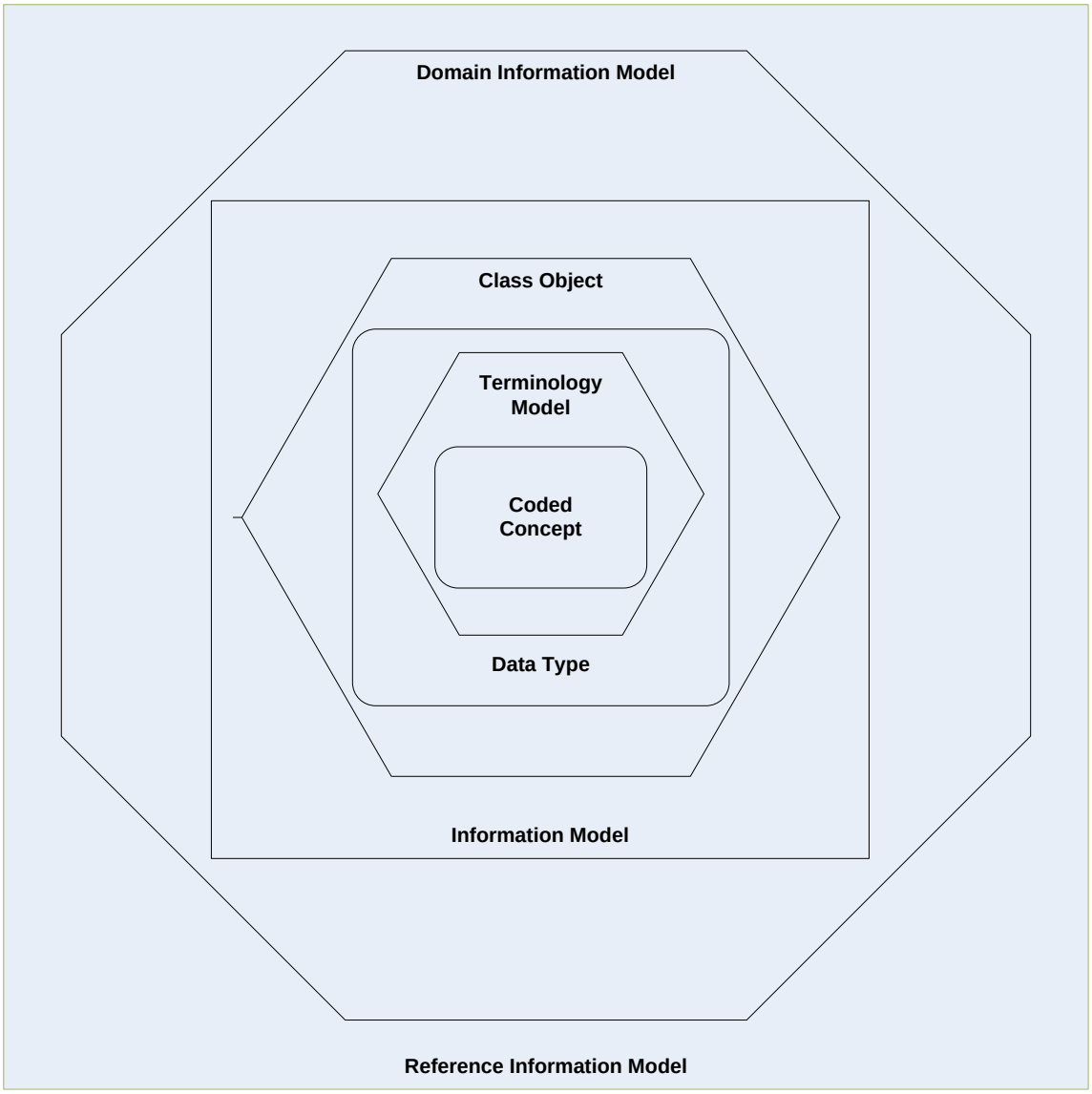


Figure 19 Artifact context wrapping

5 Enterprise Consistency and Conformity Framework (ECCF)

5.1 Purpose

The Enterprise Consistency and Conformity Framework defines the language that describes the semantics of the relationships between the cells formed by the intersection of the dimensions (columns) and the perspectives (rows) of the Interoperability Specification Matrix (ISM). The concepts defined in the SAIF Canonical Definition document to ensure coherent discussions in the context of one or more SAIF Implementation Guides (SAIF IGs). Recall that the ISM is a *Type*. Each SAIF Implementation Guide (SAIF IG) uses the ISM to define an IG-specific *Profile*, the Interoperability Specification Template (IST) as a realization of the ISM. A specific collection of artifacts in a particular instance of an IST is referred to as an Interoperability Specification Instance (ISI). A more detailed discussion of the ISM, IST, and ISI and their relationships is provided in Section 6.

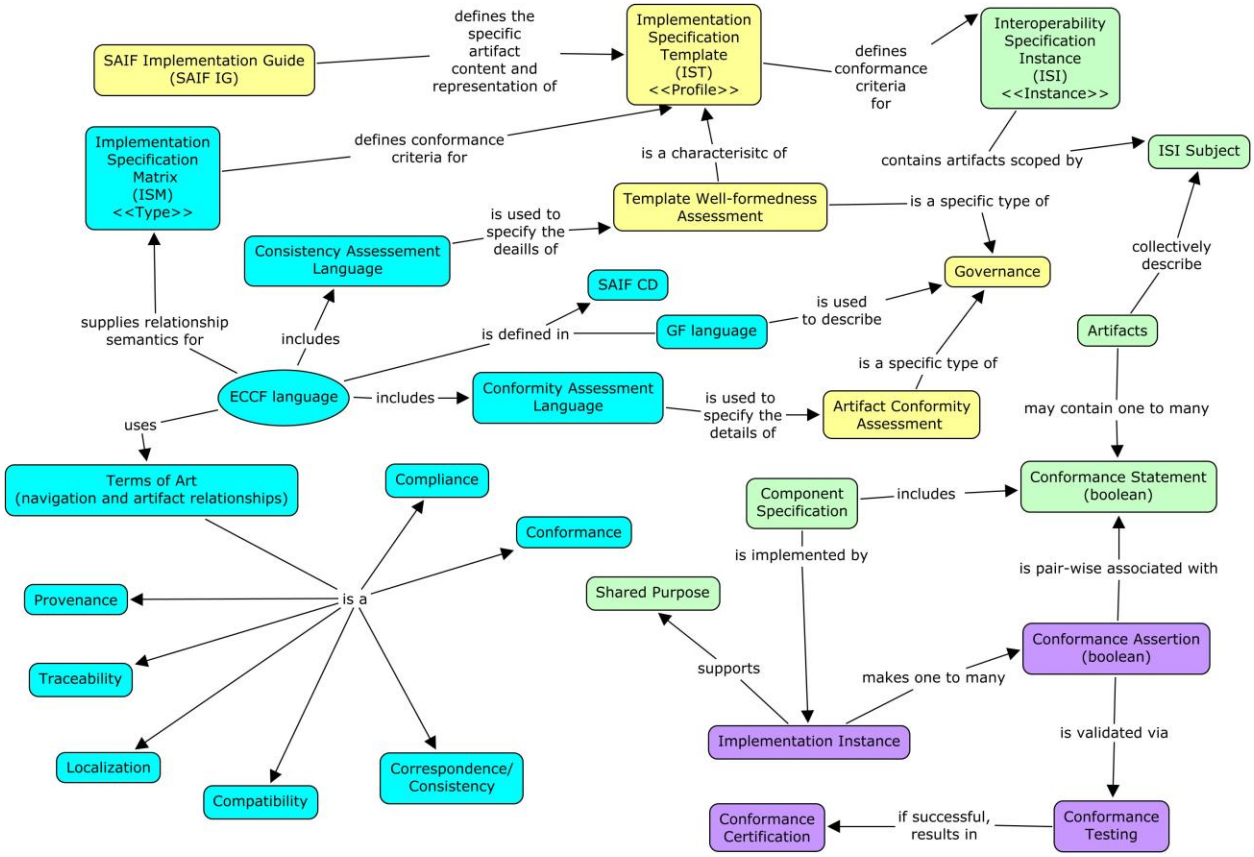


Figure 20 ECCF Terms of Art Concept Map. (See Figure 1 for color convention semantics)

5.2 ECCF Terms of Art

The terms *consistency* and *conformity* are both composite terms whose meaning is derived from the collective meanings of the ECCF terms of art. In addition, both terms have formal roots in both the ISO standards and ODP arenas. As shown in the concept map (above), ECCF language as defined in the SAIF-CD is instantiated in individual SAIF IGs with a focus on both *Conformity Assessment* and *Well-formed-ness (Consistency) Assessment.* Within the context of the SAIF-CD, the two core concepts are defined as follows:

Consistency: "Well-formed-ness" of artifacts both within the artifact itself, i.e. its content and representation conventions, and between artifacts, i.e. identical semantics are correctly and accurately represented across artifact boundaries, and explicit and implicit dependencies are accurately and consistently represented. *"Steadfast*

adherence to the same principles, course, form, etc. Agreement, harmony, compatibility, and especially correspondence or uniformity among parts of a complex thing." (Definitions.net, 2011)..

Conformity: A measure of the *conformance* of a given implementation instance to a given specification AND/OR a measure of the *compliance/correctness* of a given specification to another specification, usually in the context of the compliant specification being deemed a valid transformation from the original specification. "*Conformity assessment is the name given to processes that are used to demonstrate that a product (tangible) or a service or a management system or body meets specified requirements. (ISO)"*

Interoperability Specification Instance (ISI) Subject: Each instance of an Interoperability Template, referred to as an Interoperability Specification Instance (ISI), contains artifacts whose scope collectively defines a particular component, for example, system, sub-system, service, document, or message. This scope is referred to as the Interoperability Specification Instance Subject.

Conformance: "Conformance relates an implementation to a standard. Any proposition that is true of the specification must be true in its implementation. (ISO, 2010)"

The ECCF provides a language that enables specification developers and consumers to explicitly understand and communicate about various aspects of a given component that impact its use in one or more interoperability scenarios. A key aspect is the ability to speak quantitatively about the degree to which a given implementation satisfies the static or informational and dynamic or behavioral semantics, or both, as defined in the various artifacts contained in an ISI. A given implementation instance is said to be conformant to a given specification if the implementation instance satisfies the various requirements defined in the specification.

The ECCF does *not* define conformance at the "global" implementation level—an implementation is either *conformant* or *non-conformant* to a given specification. Rather, conformance is defined at the more granular level of the *Conformance Statement*, a testable, Boolean-valued statement of a specific requirement (static or dynamic) of the component as explicitly specified in the component's ISI.

A given implementation then makes *pair-wise Conformance Assertions*, claiming that it satisfies particular Conformance Statements. These claims can be validated on a one-by-one basis through either automated or human-based testing. Thus, within the context of the ECCF, the concept of Conformance has two defining characteristics:

- Conformance is only used to discuss the relationship between an implementation and a specification.
- Conformance is tested and certified at a granularity determined by Conformance Statements contained in component-specific artifacts in an ISI. Conformance Statements in a given ISI are associated pair-wise with Conformance Assertions made by the implementation claiming conformance to the ISI. This relationship is shown in the illustration that follows. Note that Conformance Statements are testable Boolean requirements collected at Conformance Points as defined in RM-ODP.

Conformance Statements: Paraphrasing from [ISO/IEC 10746-2 (ISO, 2010)]: "A conformance Statement is a statement that identifies testable requirements at a specified Conformance Point within a specification, explicitly defining the behavior which must be satisfied at these points. Conformance Statements will only occur in standards which are intended to constrain some feature of a real implementation, so that there exists, in principle, the possibility of testing."

The conformance of a given implementation instance to a particular specification is verified based on the truth value of a pair-wise Conformance Assertion made by an implementation instance against a given artifact-resident Conformance Statement within a given specification.

Note that the requirement that each Conformance Statement be testable and verifiable, that is, that each Conformance Statement be a Boolean statement, does not require that the statement be testable by automated means. Often Conformance Statements made from the Conceptual Perspective, and particularly those made in the Enterprise dimension, may only be verifiable as True through human examination of a given implementation instance. Thus, the critical defining feature of a valid ECCF Conformance Statement is its Boolean testability and not its particular mode of verification.

Conformance Assertions: Conformance Assertions are made by a given implementation instance and are linked pair-wise to Conformance Statements made within a given artifact as part of a component specification. The pair-wise association of specification-resident Conformance Statements with implementation-instance-resident Conformance Assertions enables creation of testing harness and user verification frameworks. This enables a given implementation instance to be "verified" or "tested" as "conformant to a given specification." Note that the words "tested," "verified," and "certified" are subject to confusion and conflated definitions and usage. The ECCF therefore uses very specific definitions of terms to proactively prevent this confusion.

Conformance Testing: - Quoting from [ISO/IEC 10746-2 (ISO, 2010)*]:* "A Reference Point (RP) is a point in the specification which a specifier nominates to be a candidate Conformance Point, that is, a place where behavior may need to be observed to determine conformance. A specifier may define many RPs in the specification but it may be that only a subset of these can be used for testing in specific scenario. These are referred to as conformance points. Note that in the context of SAIF, the notion of an RP can be stated as "the statement(s) in a given artifact that that are referred to as an ECCF Conformance Statement").

1. **Perceptual:** an RP where there is some interaction between the system and the physical world, for example, a human-computer interface.
2. **Programmatic**: an RP where a programmatic interface can be established to allow access to a function.
3. **Interworking :** an RP where there is a physical communication channel through which information exchange can be monitored.
4. **Interchange** - an RP where an external physical storage medium can be introduced into the system, for example, in cases where information can be recorded on one system and then physically transferred, directly or indirectly, to be used on another system.

NOTE: ODP defines four broad categories of Reference Points, the first two of which are relevant to the SAIF-CD (points 3 and 4 are only relevant in the context of a specific implementation and are therefore outside the scope of the SAIF-CD and are included simply for completeness with respect the ODP reference).

From the preceding discussion of Conformance Statements and Conformance Assertions, it should be clear that Conformance Testing, that is, the process whereby a given implementation instance is evaluated to determine which of its various Conformance Assertions are valid implementations of a given specification's Conformance Statements:

- Is a granular construct, i.e. it is determined at the level of individual Conformance Assertions made by the implementation instance and not a global characteristic of a given implementation instance (unless, of course, the specification contains only a single global Conformance Statement against which the implementation instance can claim conformance); and
- Exists in a one-to-many relationship between specifications and implementations, i.e. there is a one-to-many relationship between a given specification instance and the collection of implementation instances that can claim conformance to the specification.

Compliance: Quoting from [ISO/IEC 10746-2 (ISO, 2010)]: "Requirements for the necessary consistency of one member of the family of specifications or standards with another are established during the standardization process. Adherence to these requirements is called compliance."

In the context of SAIF, Compliance refers to logical consistency and correspondence between a source artifact and a target artifact, with the target having undergone a transformation (usually a restriction). That is, given an existing source artifact such as a specification or standard, and a target artifact that resulted from applying a known transformation to the source, the target is in Compliance with the source if the transformation is considered "legal" by the source artifact's originator.

Compliance can be established between artifacts in a single ISI cell or, alternatively, across multiple ISI cells. When a Compliance relationship crosses cell boundaries, it can do so either horizontally or vertically. Diagonal

Compliance is also possible although less common then vertical or horizontal Compliance relationships. Thus, localization is considered a form of Compliance.

Unlike Conformance, Compliance is seldom overtly tested since non-compliant transformations producing non-compliant artifacts usually cause other issues which can be discovered in either Correspondence monitoring or Conformance testing.

Certification (Conformance Certification): the outcome of *successful* conformance testing, i.e. the results of that testing. Certification should not be confusion with the testing that results (potentially) from the test/evaluation. Certification of Conformance (or lack thereof) is based on the ability of a given implementation instance to satisfy one or more of the Conformance Assertions made by the implementation instance against the pair-wise Conformance Statement in the specification.

Correspondence and Consistency: Quoting from [ISO/IEC 10746-2]: "Viewpoint correspondence is a statement that some terms or other linguistic constructs in a specification from one ODP viewpoint are associated with (e.g. describe the same entities as), terms or constructs in a specification from a second ODP viewpoint. The forms of association that can be expressed will depend on the specification technique used."

In the SAIF ECCF, *Correspondence* is used synonymously with the term *consistency*, the latter term having been chosen over the former as the *nom de plume* of the ECCF because of the more commonly shared understanding of the term as opposed to the term "correspondence." Both terms are focused on the notion of logical coherence of a given ISI that is "unified" in its expression of a given component's various Dimensions and Perspectives. Thus, a *consistent, well-formed* specification – demonstrates a high degree of correspondence between its various components. This is a somewhat hard-to-define but relatively easy (to the trained eye) to perceive "expressive traceability."

In summary, the notion of Correspondence underscores the fact that the Dimensions of an IST are not orthogonal, but rather express different aspects of a single component, system, sub-system, and specification.

Traceability: In everyday parlance, traceability refers to the ability to link an instance with a concept, for example, a requirement, with an implementation-resident functionality. In the context of SAIF, traceability has a somewhat more formal meaning. Traceability defines the relationship that links an attribute or other feature of a particular artifact defined in a particular dimension and at a particular perspective. This includes but is not limited to semantics or Conformance Statements. Note that traceability is a vertical relationship spanning all Perspectives and including any implementation instances associated with a given specification. Traceability includes both Conformance and Compliance relationships.

Provenance: The documented "reverse traceability" of an existing artifact from its current state to its origination, including whatever attribution, context or both, is associated with the various lifecycle changes of the artifact. Provenance is, among other things, the source for documenting the various constraints and localizations that a given item undergoes as it moves from, for example, a Conceptual to a Logical to an Implementable specified artifact.

Localization: A specialization of compliance whereby some aspect of an artifact's semantics, informational (static) or behavioral (dynamic), or other defining attribute is restricted compared to its original occurrence. Localization commonly occurs as a concept passes from one or more of the following: the Conceptual Perspective to the Logical Perspective, the Logical Perspective to the Implementable Perspective, and the Implementable Perspective to an implementation instance.

Compatibility: Given a specification, two implementation instances are said to be Compatible if-and-only-if they can successfully engage – without further modification of their implementation specifics – in any shared purpose scenario that can be expected to be supported based on the reference specification that is implemented by the involved instances. In other words, two implementation instances are said to be Compatible if they do not "localize" by specifying context-specific, non-interoperable constraints.

6 Interoperability Specification Matrix (ISM)

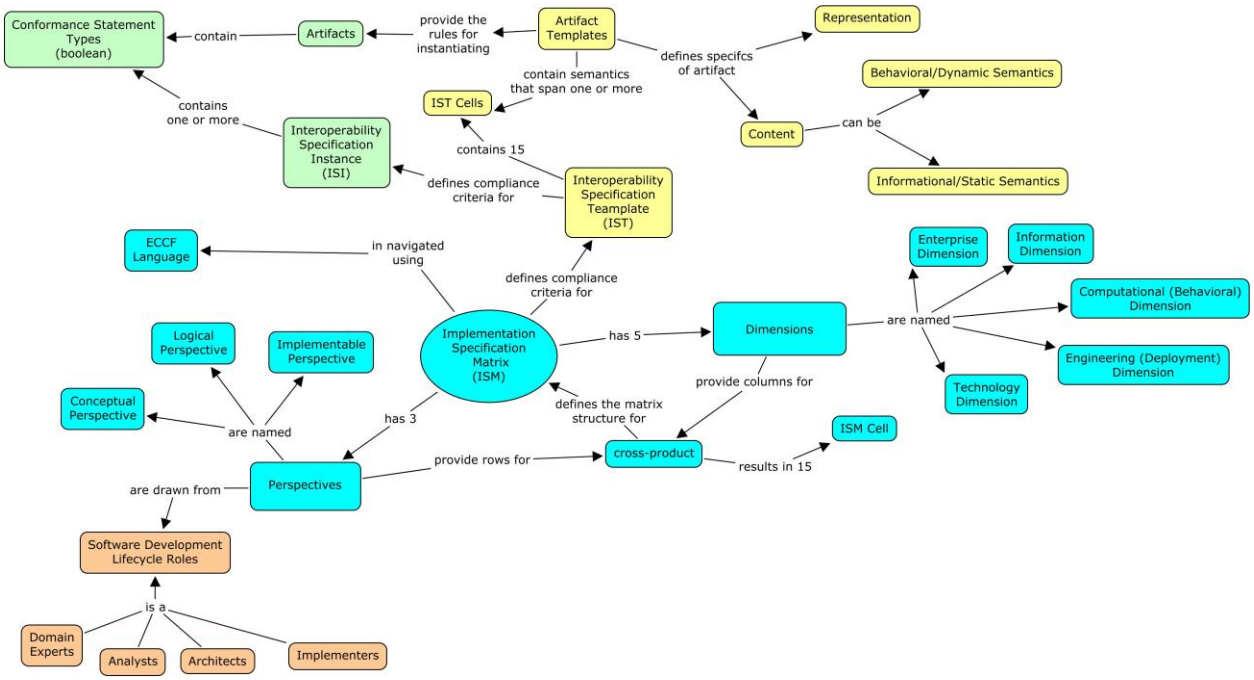
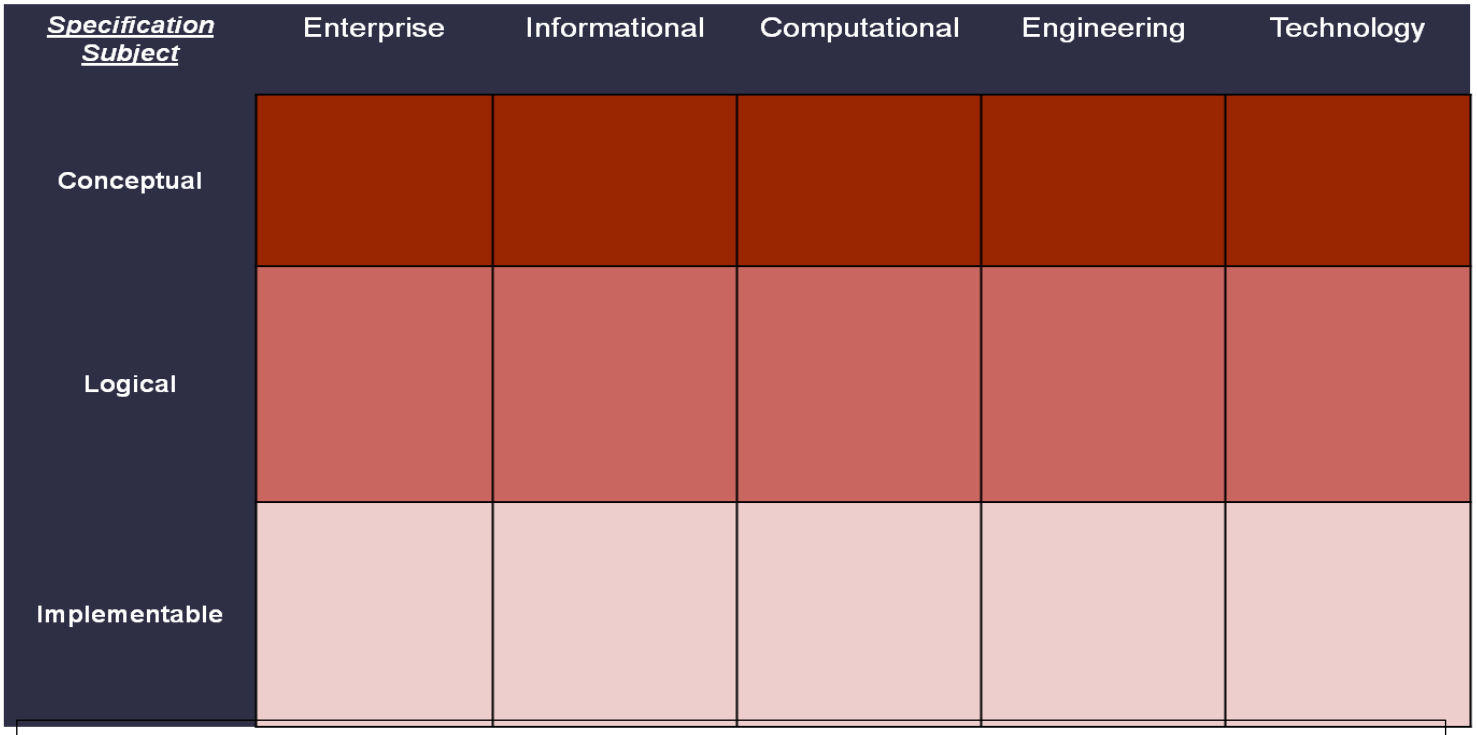


Figure 21 Interoperability Specification Matrix Concept map. (See Figure 1 for color convention semantics).

The Interoperability Specification Matrix (ISM) defines a 5-column-by-3-row matrix ("table") which distributes the multiple aspects of a given component's specification across the various cells of the of matrix. The structure of the ISM is based on proven cognitive models for describing complex systems which revolve around the notion of partitioning complexity based on a number of Dimensions while simultaneously viewing each of these dimensions from multiple Perspectives.



NOTE: At the SAIF-CD level, no specific artifacts (i.e. individual cell content) is specified as this is in the domain of an organization-specific SAIF IG. The SAIF-CD is responsible for defining the semantics of the ISM's construction (i.e. meaning of columns and rows) and its relationship to its derived <<profile>>, the Interoperability Specification Template (IST)

Figure 22 Interoperability Specification matrix.

NOTE: In the context of a specific SAIF IG, the ISM defines a <<type>> construct which is then explicitly made manifest in a SAIF IG-specific <<profile>> that specifies the content and representation of all artifacts that collectively comprise a given component's specification. The process of defining an ISM-conformant matrix for a given IG – a construct referred to as an Interoperability Specification Template (IST) – involves the use of restrictions and specializations of the concepts and constructs used to define the ISM. A collection of specification artifacts for a given component is then an <<instance>> of the profile and is referred to as an Interoperability Specification Instance (ISI). Finally, given a particular specification instance, one or more implementations of that specification can be developed and deployed and, in the process, subject to conformance certification testing to determine the degree of fidelity that the implementation has relative to the specification. (See Figure 2 and Section 7.3 for details and a more complete discussion.)

6.1 ISM Artifacts Types and Conformance Statement Types

As shown in the preceding concept map, the ISM defines prototypic artifacts ***types,*** the specific content and representation of which are defined in a particular SAIF-CD-compliant SAIF IG. In addition, although the SAIF-CD does not define specific artifacts, it does require that specific artifact *instances* contain testable – i.e. Boolean – Conformant Statements. Thus, in parallel to the SAIF-CD definition of artifact types, the SAIF-CD defines Conformance Statement ***types***. These types are, in turn, defined in SAIF IG ***profiles.*** Finally, a given artifact in an ISI can contain multiple Conformance Statement ***instances*** against which a given implementation of a component specification can make pair-wise Conformance Assertions. (See Appendix for examples of artifacts and associated Conformance Statements.)

6.2 Dimensions

The names of the Dimensions in the SAIF ISM are identical to the Viewpoint names in RM-ODP. However, the semantics are not identical. In particular, the SAIF-CD Dimensions are restrictions and/or specializations of the various RM-ODP Viewpoint languages. The SAIF-CD-specific definitions are as follows:

6.2.1 Enterprise Dimension

The Enterprise Dimension focuses on defining salient aspects of the "organizational context." In the context of interoperability, this means "the intra- or inter-organizational deployment and interoperability context" for which the specification-specific artifacts are being defined.

For each of the three perspectives, the Enterprise Dimension should aspects of the interoperability context that emerge from an understanding of business objectives and business rules. This includes relevant pre- and post-conditions for interoperability scenarios.

Due to the basic nature of the Enterprise dimension, most information at the Logical and Implementable Perspectives originates in the Conceptual Perspective. Very little "new" information is added at the Logical and Implementable Perspectives in the Enterprise Dimension.

6.2.2 Information Dimension

The Information Dimension focuses on defining the informational or static semantics that are relevant with respect to interoperability interactions.

These semantics are expressed using Information Framework (IF) grammar and include constructs such as information and data models, data types, and value sets, discussed in the IF chapter of this document. However, as discussed in the IF chapter, the scope of the Information *Framework* is *not* limited to use in Information *Dimension* specifications.

6.2.3 Behavioral (Computational) Dimension

The Behavioral (Computational) Dimension focuses on defining the behavioral or dynamic semantics that are relevant with respect to interoperability interactions. These semantics are expressed using Behavioral Framework grammar and include constructs such as contract and interface specifications and accountability profiles, discussed in the BF chapter of this document. The BF makes extensive use of the RM-ODP Enterprise Language, a set of well-defined concepts and constructs that are defined as part of the RM-ODP Enterprise Viewpoint. Therefore the scope of the Behavioral *Framework* is not limited to use in Behavioral Dimension specifications.

6.2.4 Engineering Dimension

The Engineering Dimension focuses on defining the deployment topologies that are relevant with respect to interoperability interactions. The RM-ODP (ISO RM-ODP) contains considerable detail about the construct "transparencies." Discussion of transparencies is beyond the scope of the SAIF-CD. However, certain SAIF-IGs could benefit substantially from including certain transparency constructs in their organization-specific IGs. Specifically, salient details of different implementable meta-models (for example, specifications supporting interoperability scenarios based on messages, documents, or services) can be explicitly captured across the three perspectives of the Engineering Dimension.

6.2.5 Technology Dimension

The Technology Dimension focuses on defining various implementable standards for hardware or software as relevant, which will ultimately support the specification. This definition is referred to as the "technology semantics" of a component as used in interoperability scenarios.

Artifacts defined under the Technology Dimension often make reference to artifacts in other ISM cells in order to appropriately contextualize the referenced artifacts. Further discussion of the Technology Dimension is appropriate

for SAIF-IGs and includes topics such as technology-specific deployment or configuration guides, technology selection criteria, and maintenance and migration plans. Conformance Statements are not defined under the Technology Dimension as often as they are under the other dimensions. Refer to the discussion of conformance in the ECCF chapter.

6.3 Perspectives

The perspectives correspond to standard role-based terminology of contemporary software engineering processes.

The names of the perspectives or rows of the ISM reflect views of specification artifacts associated with software engineering roles, that is, Domain Expert, Analyst, Architect, Developer, and others as discussed below. The HL7 ArB chose to use three perspectives rather than more finely granulated alternatives, for example, the six Perspectives of Zachman2.

It is possible to associate each specified artifact with a row in a RACI (Responsibility, Accountability, Consulted, and Informed) matrix. This can explicitly link the artifact to the appropriate organizational roles for a SAIF IG.

NOTE: The SAIF-CD definitions of the three SAIF Perspectives and their associated software-engineering role are given in the following discussion. It is important to note that the SAIF-CD Perspectives are not formally linked with the Object Management Group's levels-of-abstraction in Model-Driven Architecture (MDA). That is, the SAIF Conceptual Perspective is not semantically equivalent to the MDA concept of Computationally Independent Model (CIM), the Logical Perspective is not equivalent to the MDA Platform Independent Model (PIM), nor is the Implementable Perspective equivalent to the MDA Platform Specific Model although this Perspective is the SAIF Perspective that most closely aligns with an MDA analogue.

6.3.1 Conceptual Perspective

The artifacts of the Conceptual Perspective are of interest to and readable by Domain Experts(DEs) or Subject Matter Experts (SMEs). These artifacts are most commonly focused on the "Problem-Space" rather than the "Solution Space," and contain, distributed across the five columns of an ISM, explicit, unambiguous descriptions of the various dimensions of the component or system that being specified.

Artifacts of the Conceptual Perspective are normally developed by "outward-facing analysts" who have reasonable domain knowledge and can facilitate dialogues with DEs and SMEs. These analysts also take the results of such dialogues and represent the content in structured artifacts which remain understandable to DEs or SMEs. These sometimes formally structured artifacts may include clearly-stated business rules, concept maps, and simple UML class or activity diagrams.

A fully-specified Conceptual Perspective thus should be both readable and vettable by DEs and SMEs and rigorous enough to serve as input into the development in the Logical Perspective.

6.3.2 Logical Perspective

Artifacts in the Logical Perspective represent traceable translations of Conceptual-level artifacts into a form and format, usable by and useful to architects and "inward-facing analysts." Also included are additional specification materials required by architects preparing artifacts for consumption by developers.

Note that there is no firm or fixed line that definitively and unambiguously determines where the Conceptual Perspective ends and the Logical Perspective begins. The same is true of the lack of definitive boundaries between the Logical and Implementable Perspectives.

For a given SAIF-IG, the most important aspects of defining artifacts in a given perspective are the combination of role-based awareness based on artifact creation and consumption, and consistent placement of artifacts across multiple specifications.

6.3.3 Implementable Perspective

Artifacts in the Implementable Perspective are typically defined by developers, often through dialogues with designers, architects, or both. Note that the artifacts in the Implementable Perspective are not actual implementations, but rather *implementable in a number of implementation instances*. Thus all the necessary technical bindings, including data types, value sets, class libraries, and interface specifications, can be found distributed across the ISM dimensions at the Implementable Perspective. These artifacts will enable one or more instances of the specification to be realized by one or more development teams.

7 Appendix

7.1 ISM Specification Matrix, Template and Instance

The SAIF Interoperability Specification Matrix (ISM) defines a structure for categorizing artifacts that collectively describe a complex component or system. As such, the ISM can be viewed as a formal *Type.* The ISM defined by the SAIF Canonical Definition is ultimately realized as an ISM *Profile,* referred to as an *Interoperability Specification Template* (IST) in a particular SAIF IG. An IST defined by a particular SAIF IG specifies the *content* and *representation* of specific artifacts in the various dimensions and perspectives of the ISM.

Figure 24 depicts an exemplar Interoperability Specification Template (IST) containing named artifacts, the specific content and representation of which would be formally defined in the SAIF IG in which the IST was defined.

Specification Subject	Enterprise	Informational	Computational	Engineering	Technology
Conceptual	**Business Context, Reference Context**	**Domain Analysis (Information) Model**	**Collaboration Analysis, Functional Profile(s), Service Roles and Relationships**	**Existing Platform capabilities**	
Logical	**Business Governance**	**Project-oriented Domain Information Model, Constrained Information Model, Localized Information Model, Hierarchical Message Definition**	**Collaboration Types, Interface Specification and Functional Groups, Interaction Types and Collaboration Participations, Contracts Parts**	**Existing Platform models, libraries, etc.**	**Security Standards**
Implementable	**Rules, Procedures**	**Localized Information Model, Transforms, Schema**	**Collaboration scripts, Orchestrations, Realized Interfaces**	**Execution Context, Platform Bindings, Deployment Model**	**Security Services Routing Services**

Figure 23 Examplar Interoperability Specification Template

Once the requirements for specifying artifacts have been defined, multiple *instances* are produced using the appropriate tools and technologies. Each instance contains actual artifacts whose content and representation are conformant to the criteria specified in the IST. A specific collection of artifacts describing a particular component –

e.g. service, message, document, etc. – is referred to as an Interoperability Specification Instance (ISI), i.e. an ISI is an instance of an IST.

Finally, a given ISI may then be implemented via one or more specific implementations, each of which may be evaluated for its conformance to the specification instance through the evaluation of implementation-specific Conformance Assertions which are made and linked *pair-wise* to associated Conformance Statements in the specification instance as illustrated in the following graphic:

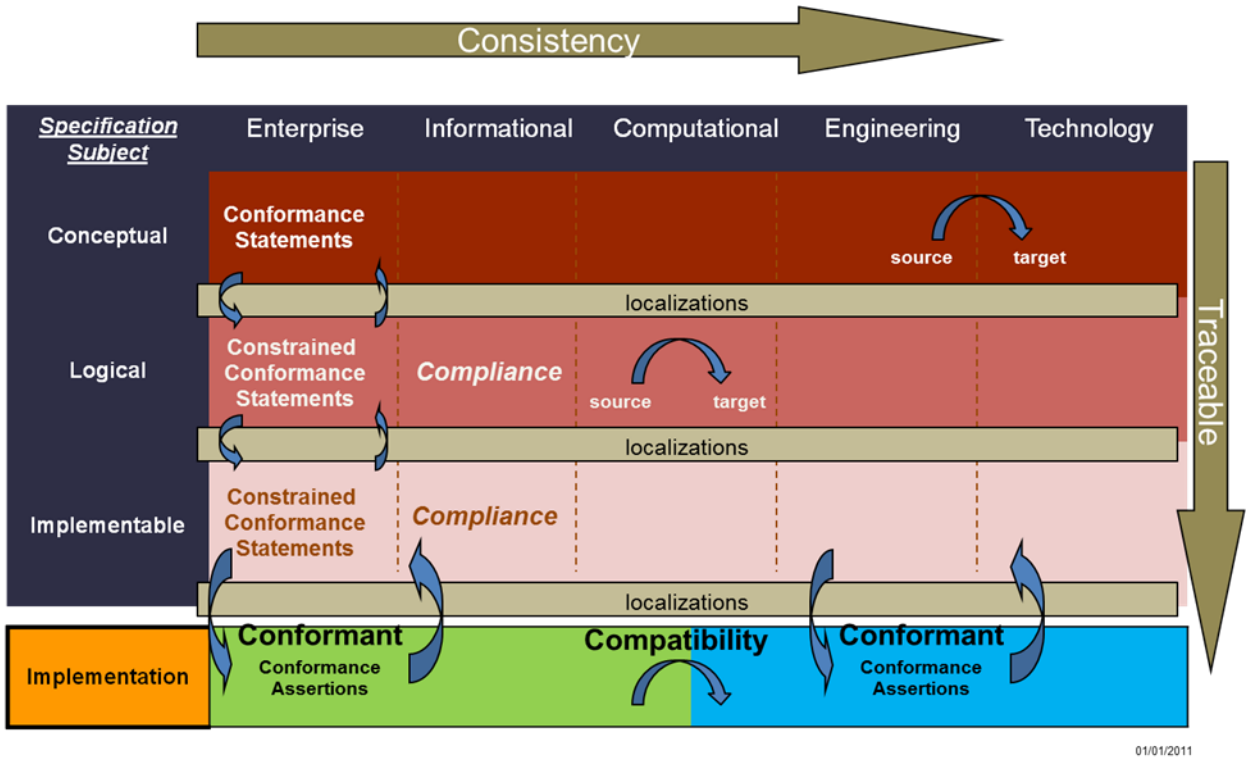


Figure 24 Another view of an IST

Figure 24 depicts another view of an IST notated to indicate some of the specific relationships defined by the language of the ECCF. Note the present of *Localizations* between each Perspective as well as between the Implementable Perspective and candidate implementations. Specific Localization semantics are an example of one type of contextualization that a SAIF IG may make in its application of the SAIF-CD languages.

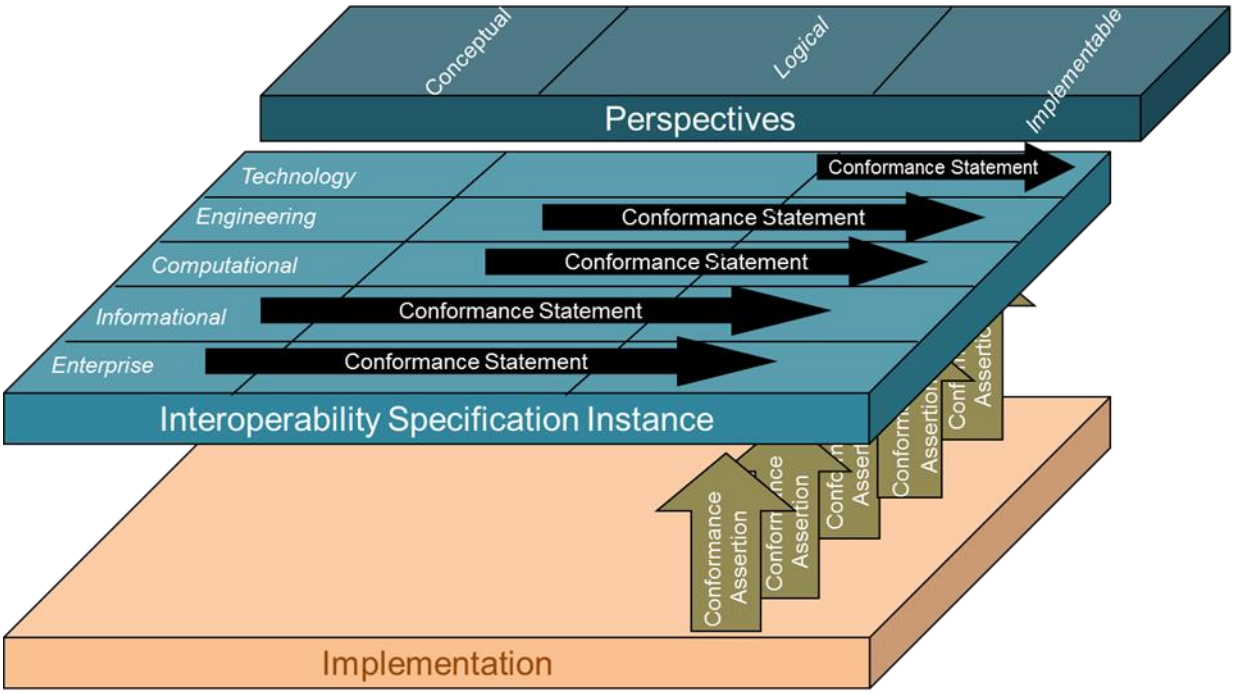


Figure 25 Binding II to SI through Conformance Assertions

Figure 25 depicts the graphical representation of the binding of an implementation instance to a specification instance through the use of testable Conformance Assertions made by the implementation against pair-wise Conformance Statements defined in the Interoperability Specification Instance.

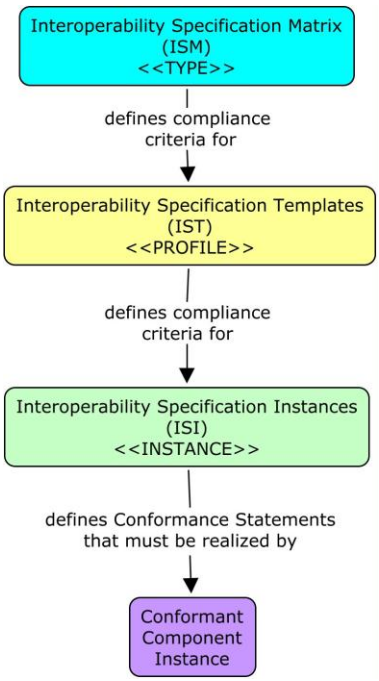


Figure 26 Relationships between the ISM, IST, and ISIs.

Figure 26 shows the relationship between the ISM, the IST, and ISIs. The Interoperability Specification Matrix (ISM) is a *type* as defined in the SAIF-CD. The Interoperability Specification Template (IST) is a *profile* which is defined in *each* SAIF IG through the application of restrictions and specializations of the ISM language. The multiple component specification, referred to as Interoperability Specification Instances (ISIs), are *instances* of the artifact content and representations specifics defined in the IST. Note that the terms "type," "profile," and "instance" are represented in the illustration as UML-like *stereotypes.*

Note that neither the definition of the ISM nor its realization in a given SAIF-IG as an IST specifies a process whereby a given matrix instance is to be populated. That is, there are no rules such as "all of the required artifacts in the Conceptual row of the ISM should be fully specified before artifacts in the Logical row are specified."

Each ISI has a particular scope, for example, system, sub-system, or service, i.e. a scope that is defined by the collection of artifacts in the ISI. The scope of the ISI is referred to as the *Specification Subject (SS).* Each cell in an ISI can contain multiple artifacts which may or may not contain artifact-to-artifact links or relationships, and which may be hierarchical in terms of level of detail or abstraction.

The normative content of the Enterprise Conformance and Compliance Framework of the SAIF Canonical Definition is the definitions and details of the various inter-cell and inter-artifact relationships. Refer to the discussion in the ECCF chapter.

Given a particular ISI that, by definition, contains artifacts that collectively specify a given component from the perspective of one or more interoperability scenarios, one or more development teams can develop an implementation of the specification, thereby "binding" a specific implementation instance to the specification.

The ECCF chapter of the SAIF Canonical Definition establishes the concept of *conformance* of a given implementation instance to a given ISI in terms of evaluation of specific Conformance Statements made within specification artifacts, and the Boolean veracity of those statements to Conformance Assertions made by a given implementation instance. These concepts are discussed more fully in the ECCF chapter of this document.

In summary:

- The artifacts collected in a given ISI contain descriptions of a given component's informational or static and behavioral or dynamic semantics, features and functions.
- Specifications regarding a component's informational or static semantics and other informational aspects are expressed using the Information Framework grammar.
- Specifications regarding a component's behavioral or dynamic semantics and other behavioral aspects are expressed using the Behavioral Framework grammar.
- The relationships between artifacts within and between cells, row-by-row, column-by-column, or column-by-row basis, are defined using the Enterprise Conformance and Compliance Framework grammar.
- The content and representation of each artifact must be defined in the context of the organization's SAIF IG.
- The overall management of the life cycle of each artifact, including the correctness and completeness of the artifact as well as RACI relationships for the artifact, are defined by the Governance Framework grammar.

7.2 Foundational Principles

The material in this section is not part of the Canonical Definition of HL7 SAIF. It is included to provide context for the definitions of the four SAIF-CD Frameworks. Four Foundational Principles are discussed:

1. Shared Purpose
2. Fowler's Accountability Pattern
3. "Service-Awareness"
4. Relationship of SAIF-CD to the Reference Model for Open Distributed Processing (RM-ODP)

7.2.1 Shared Purpose

Shared Purpose between participating parties is manifested in cross-enterprise or cross-organizational interoperability, i.e. communication across organizational boundaries. Both parties must decide on the multiple details that collectively define an interaction or set of interactions. There must be an agreed upon value received for cost and effort expended. At minimum, the basic dimensions of a Shared Purpose agreement answer the questions "who," "what," and "when."

A Shared Purpose is at the heart of any successful instance of technical interoperability. Successful execution of a Shared Purpose agreement as it is realized in technology depends on explicit definition and representation of contracts, roles, interactions, behaviors, accountabilities, policies, and enforcement (governance). The SAIF-CD has leveraged considerable work by multiple sources in the area of Shared Purpose, in particular by adopting and adapting material from:

- Martin Fowler—Accountability pattern
- SOA literature—conceptual notion of "service-awareness"
- Reference Model for Open Distributed Processing—selected terminology (ISO RM-ODP)

Discussion follows of the contribution and context of each of these resources as used in the SAIF-CD.

7.2.2 Fowler's Accountability Pattern

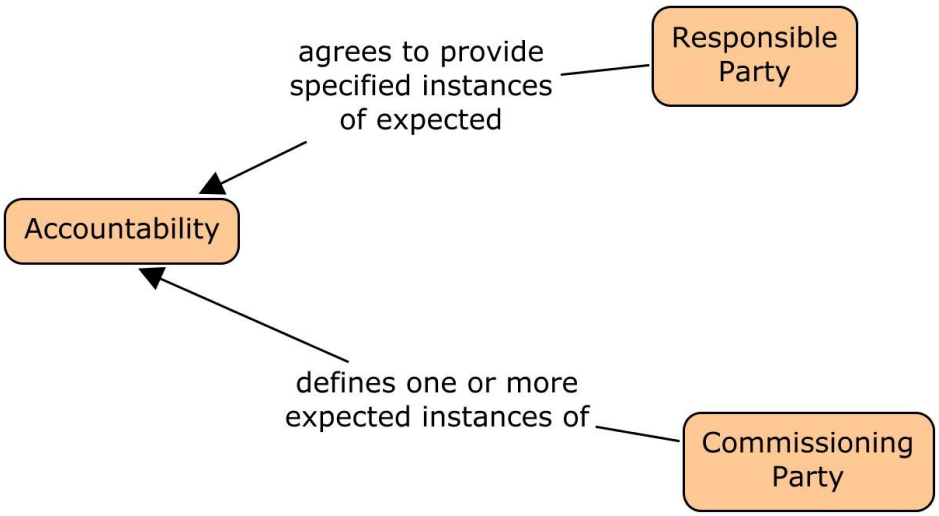


Figure 27 Concept Map representation of the Accountability Pattern of Martin Fowler

The Accountability Pattern of Martin Fowler (Fowler & Feathers, 1997) defines the notion of a Contract through the explicit representation of Accountability, that is, a Commissioning Party establishes a contract with a Responsible Party to accomplish one or more tasks. The success of the Responsible Party's actions can be assessed by the Commissioning Party via one or more agreed-upon Accountabilities which can take a form such as deliverables or tasks executed (Fowler & Feathers, 1997).

Although not shown in the diagram, Fowler's Accountability pattern formalizes the notion of a ***contract*** as a "collection of accountabilities" which have been agreed to by the Commissioning and Responsible Parties between whom the contract is established. Accountabilities are assumed to be the result of behaviors on the part of either or both parties (more likely the Responsible Party), and a variety of interactions between the two Parties can also be described in the context of Accountabilities. For example, in order to accomplish a particular task, the Responsible Party may need the Commission Party to do something first. Also implicit in the diagram is the notion that the contract exists for a specified period of time.

Although some of the terminology used by this pattern— Commissioning Party, Responsible Party, is not used in the SAIF-CD, it is replaced and elaborated upon by specific language from the Reference Model for Open Distributed Processing.

7.2.3 "Service-Awareness"

The Service Aware Interoperability Framework Canonical Definition (SAIF-CD) has matured and evolved over the three years since the HL7 Chief Technology Officer asked the HL7 Architecture Board (ArB) to provide a roadmap and specific deliverables that would result in development and specification of an enterprise architecture for HL7. In that time, there has been considerable confusion over the term "service-aware." In contrast, the term "interoperability framework", although broad with respect to the exact type of interoperability, is much less subject to confusion.

The "Service-Aware" in the SAIF-CD indicates that the **behavior** of a given component is the primary classifier of that component from the perspective of the component's involvement in an interoperability scenario focused on achieving a shared purpose. Other terms are often associated with design, implementation, or run-time specifics that are important but secondary to characteristics that define the expected interaction-based behavior of a given component. As a consequence, the term "service-aware" replaces other concepts often used to describe a component, including those based on specific implementation technologies and information-exchange types.

The term "service-aware" is used as the primary identifier of the frameworks of the SAIF-CD because each of the concepts is *overtly considered* when working an environment based on contemporary service-based architecture

paradigms. Examples are SOA and service-based technologies such as SOAP or REST paradigms. The concepts can also be realized in a non-service environment assuming there is a commitment to formalizing the semantics of interactions. The ArB chose the term "service-aware" to underscore the importance of these core concepts where the requirement for interoperable interactions is of central importance. The SAIF-CD and any conformant SAIF-IG can be operationalized without the use of service-based technologies. Interoperability scenarios to achieve Shared Purposes can productively be executed using approaches based on messages, documents, or other hybrid strategies and technologies. However, definition and specification of every scenario, without regard to implementation technology, relies on certain core concepts and constructs that are collectively defined as bringing "service-awareness" to the discussion. These concepts, most of which are at least implicit in Fowler's Accountability pattern and which are elaborated in RM-OPD, include:

- Role (a scenario-specific application of Fowler's *Party*)
- Behavior
- Contract
- Interaction
- Accountability
- Policy (not covered in Fowler although it is implicit in *Contract*)
- Exchanged Information (not covered in Fowler although it is implicit in *Accountability*)

The following diagram shows the core concepts and relationships that result from contextualizing and making explicit the semantics of Martin Fowler's Accountability pattern in a Service-Aware framework such as the SAIF-CD.

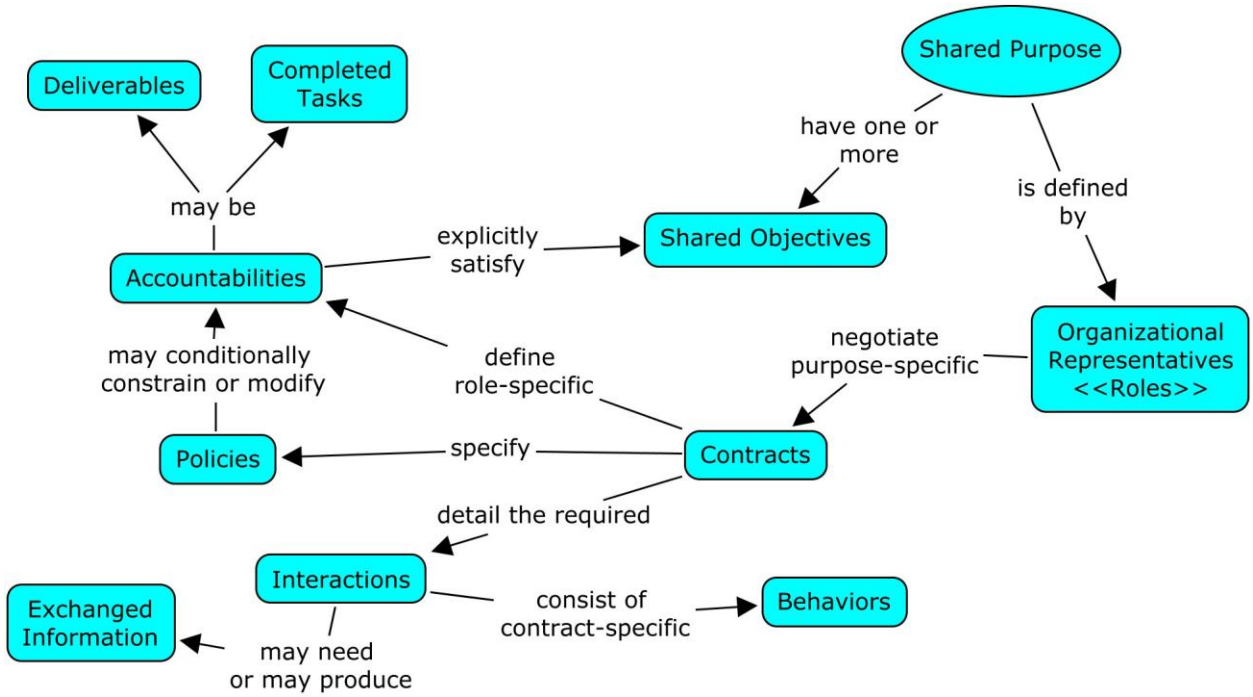


Figure 28 Shared purpose concept map

A Shared Purpose is defined by two or more parties and is explicitly described in a contract. The SOA literature refers to implementation-based parties in terms of Roles rather than the more general notion of Party, recognizing the fact that a given instance of a Party can assume more than one Role. Roles (that is, time-limited capabilities and competencies) are capable of executing specific behavior, a subset of which is relative to the contract-of-interest and referred to as Interactions. Contract-specific Interactions may require the exchange of Information as specified in the Contract. Contracts also specify Accountabilities (i.e. Deliverables and/or Tasks to be completed) and Policies (which may constrain or modify Accountabilities)

7.3 Defining a SAIF Implementation Guide

7.3.1 "SAIF enough – the Linear Value Proposition"

A common misunderstanding regarding the application of the SAIF Canonical Definition to a given enterprise revolves around the two-part question:

- What artifacts should be included in the enterprise's SAIF-IG?
- Given the artifacts specified in the SAIF-IG, does each component need to be *fully specified* in order to be considered SAIF-IG-compliant?

During the development of the SAIF-IG, at the Center for Biomedical Informatics and Information Technology (CBIIT) of the National Cancer Institute (NCI), the concept of "just enough specification" was introduced in response to the second question. It became clear that the answer to the question was a definitive NO, that is, all components did not have to be equally well-specified. Further, the best method for determining how much effort to devote to a given component's specification is a value-proposition-based decision based on understanding both the Deployment Context in which the component would be involved in interoperability scenarios, and the Interoperability Type required by those scenarios. Well-localized Deployment Contexts requiring "only" syntactic interoperability require minimal semantic specification using the various ISM artifacts defined in the CBIIT SAIF IG. As the Deployment Context becomes larger and the Interoperability Type moves from Syntactic to Computable Semantic (or both), the requirements for increased levels of explicit specification increases.

The important concept that emerged was what CBIIT terms the "linear value proposition," that is, easy things such as deploying PERL code in a single lab, should be easy; harder things should be harder, and really hard things such as deploying a service into the global community with the requirement that it support machine-to-machine computable semantic interoperability, should be the hardest.

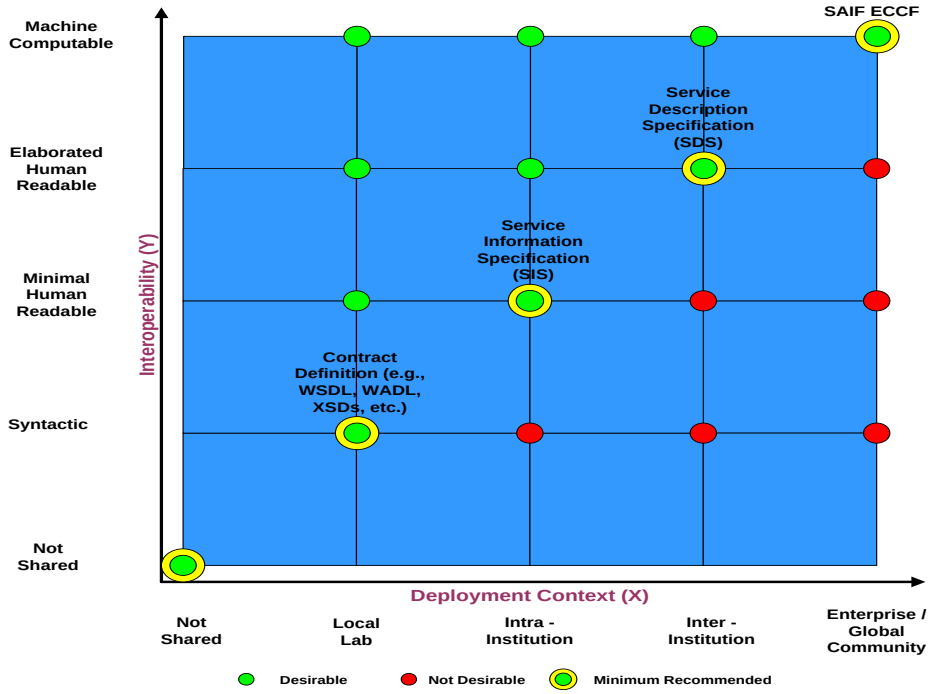


Figure 29 Deployment Context versus Interoperability Type matrix (courtesy of NCI Center for Biomedical Informatics and Information Technology (NCI CBIIT)

7.3.2 Deployment Context versus Interoperability Type

A Deployment Context is "the size and/or diversity of the community that is negotiating one or more shared purpose scenarios." For a given Deployment Context, the Interoperability Type (that is, the specific requirements for the level of interoperability needed between a given component and other components in the same Deployment Context (such as Syntactic, Human Semantic, or Computable Semantic) may vary. As the size or diversity of the Deployment Context increases and/or the Interoperability Type becomes more computation-centric, the requirements for explicit representation of technical details of the involved components increases. The SAIF-CD supports the notion of a "linear value proposition" by enabling an environment where "just enough specification" to tractably satisfy the requirements of a given shared purpose scenario can be defined and managed. (Graphic courtesy of the Center for Biomedical Informatics and Information Technology (CBIIT) of the National Cancer Institute (NCI)).

7.3.3 Defining Specification Artifacts: Content, Representation, Location

As indicated above, the canonical representation of SAIF does not specify the content, representation, or location of individual artifacts. Artifact specification is, instead, done in the context of a given enterprise's SAIF-IG. (Note that several SAIF-IGs have been and are being developed by HL7, the US Department of Defense, Canada Health Infoway, Australia NeHTA (National eHealth Transition Authority), and the Center for Biomedical Informatics and Information Technology (CBIIT) of the NCI and are generally available for review and study.)

In general, the most important aspect of artifact specification is its content, followed by its representation. Its location in a given ISI is really only of major importance with respect to the consistency of the location of a given artifact (or, more correctly, artifact type) across multiple specification instances within the context of an IG.

In addition, a given artifact may occur in more than one ISI cell, a reflection of the fact that the Dimensions and Perspectives of the ISI matrix are not normalized (as would be the case, for example, if the ISI were instantiated using the Zachman2 matrix of Dimensions x Perspectives). From the perspective of interoperability scenarios, normalization and cell-specific location are not as important as explicitness and consistency.

7.3.4 Building SAIF Specifications

From a standards development point of view, the SAIF is about providing sets of artifacts that can be compiled in specifications to discuss the terms of interoperability for a particular subject or topic. The Interoperability Specification Matrix is therefore concerned mainly with providing the means by which implementation groups, realms, or enterprises will describe these terms.

By itself, the Canonical SAIF does not provide sufficient foundation to achieve a shared purpose interoperability scenario. A given Implementation Guide must also provide

- Sets of principles used to craft specifications
- Discussion of the concepts being used from the SAIF, additional concepts, and refinements if necessary
- Templates for specifications that will include artifact types, cardinality of concepts, optionality, choices of interaction and communication patterns, and other characteristics as needed.
- Potential sample choices for artifact selection
- The implications for conformance when using a given artifact

Thus, while the Canonical SAIF provides a framework for what concepts need to be expressed and why they need to be expressed, it cannot denote how to express them, when an artifact surfaces methodologically, or where an artifact will be realized.

An implementing enterprise can also specify terms of compliance for HL7 specifications. For example, it may be useful for HL7, as a SAIF-implementing enterprise, to say that in certain Logical specifications, all information models need to be compliant with the RIM. All Implementation Guides will not be created equal, and may use different artifacts to demonstrate the same SAIF concept. Implementation Considerations

Governance is a means to reduce risk. What is governed is dependent on the shared purpose. A common understanding and agreement on a shared purpose is the first order of business in establishing a community. Aspects of interoperability that need to be governed include, but may not be limited to:

- Community participation refers to what parties in what roles are eligible to participate and what are the prerequisites for their participation.

- Policies refers to those policies within each party's jurisdiction that influence the interoperability behavior of participating systems. Systems may encode business rules that are not explicitly specified but cause incompatibilities in exchanged information or unanticipated behavior of participating systems. Aligning policies across jurisdictional boundaries is one of the most difficult tasks of a federated community.

- Identity management refers to how instances of people, people in roles, systems, technical components, information artifacts and other factors are to be uniquely identified and tracked through processes included within the scope of interoperability.

- Artifact definition and approval refer to the change management process for each type of artifact, which may be for that artifact only and may be independent from other types. Artifacts may be dependent on one another and the relationships among them must be explicit and also tracked. In the SAIF context, the full slate of ECCF artifacts are interdependent and must be managed as a coherent whole in order to support technology components that are fit for purpose and whose interoperability capabilities are consistent with each other.

- Technology component configuration refers to system interoperability for potentially multiple dependent components each having their own change management processes while being interdependent. The usual system lifecycle of development, testing and deployment is increasingly complex in an interoperability environment. Multiple technical architectures can interoperate effectively if their interfaces are conformant to specifications that constrain the behavior across system boundaries to enable consistent operations.

- Accountability refers to accountability for the completeness, quality, integrity and security of information that originates in one system and is transmitted to and used by another.

- Change management refers to an essential element in collaborations, as interdependent parts often undergo change on different schedules. The ability to assess the impact of change prior to implementation can minimize anticipated disruption as changes occur. Continual change is the expected state in a volatile environment and flexible designs and evolutionary implementation are reasonable responses.

Index

8 Works Cited

Definitions.net. (2011). *Definitions*. Retrieved 09 18, 2011, from Definitions.net: http://www.definitions.net/definition/Consistency

DOD Deputy Chief Information Officer. (n.d.). DoDAF Architecture Framework. *http://cio-nii.defense.gov/sites/dodaf20/*.

Fowler, M., & Feathers, M. (1997). UML Diagrams for Chapter 2 of Analysis Patterns. In *http://martinfowler.com/apsupp/apchap2.pdf*.

Health Level Seven International, Inc. (2011). *CMET - E_Person_universal(COCT_RM030200UV08.* Ann Arbor, MI 48104: Health Level Seven Internationa, Inc.

HL7 ArB. (2011, 04). *Saif online glossary*. Retrieved 09 18, 2011, from HL7 WIKI: http://wiki.hl7.org/index.php?title=Category:SAIF_Glossary

ISO. (2010). *ISO/IEC 10746-2 Information technology -- Open distributed processing -- Reference model: Foundations.* ISO.

ISO RM-ODP. (n.d.). RM-ODP, ISO Standard (RM – ODP, ISO/IEC IS 10746 | ITU-T X.900. *iso.org*.

Lopez, D. M. (2009, February). *A Development Framework for Semantically Interoperable Health Information Systems*. Retrieved from http://www.sciencedirect.com/science/article/pii/S1386505608000877

openEHR Foundation. (2001-2007). *OpenEHR Person Demographic Information Example.* _: openEHR.

OWlCKI, S. L. (1982, July). Proving Liveness Properties of Concurrent Programs. *ACM Transactions on Programming Languages and Systems, 4*(3), pp. 455-495.

Rector, A. L. (2004). *Models and inference methods for clinical systems: a principled approach.* Stud Health Technol Inform 107(Pt 1).

The Open Group. (n.d.). TOGAF. *http://www.opengroup.org/togaf/*.

Thomas Erl, S. G. (2011). *SOA Governance: Governing Shared Services On-Premise and In the Cloud(Prentice Hall Service-Oriented Computing Series from Thomas Erl).* Prentice Hall.

Tyndale-Biscoe, S. (Nov 2002). *RM-ODP Enterprise Language* . ITU-T Rec. X.911: ISO/IEC 15414 .

Wikipedia. (n.d.). *Language*. Retrieved 09 17, 2011, from http://en.wikipedia.org: http://en.wikipedia.org/wiki/Language#cite_note-16

World Wide Web Consortium. (2001). *Web Services Description Language (WSDL) 1.1.* World Wide Web Consortium.

Zachman, J. (n.d.). Zachman Institute for Framework Architecture. *http://www.zifa.com/*.