

COST Technical Committee "Transport and Urban Development"

COST Action C21

"Urban Ontologies for an improved communication in urban civil engineering projects" - TOWNTOLOGY Project

<http://www.towntology.net/>

Technical report n°2

Industrial standards to structure the construction information

Catarina Ferreira Da Silva (CSTB), Anne-Françoise Cutting-Decelle (Ecole Centrale de Paris)

Version: 2

Preparation date: November 2005 (revised in April 2006)

**Index**

- 1. Introduction ..... 3
- 2. The ISO 12006 Family ..... 3
- 3. The Industry Foundation Classes (IFC) ..... 5
- 4. Industry Foundation Classes for GIS (IFG) ..... 8
- 5. The ISO 18629 – Process Specification Language ..... 9
- 6. ISO 13584-PLIB ..... 18
- References ..... 25

## 1. Introduction

This document aims at succinctly present an overview of some of the standards used in the industry sector to structure domain information. One should take into account that it is not neither exhaustive nor complete. Thus it should be considered as work in progress. Anyway, this document focuses on industry-oriented standards. As so, this document version does not include general “de facto” standards for information representation, such as the Ontology Web Language framework.

## 2. The ISO 12006 Family

Whilst the ISO 12006-2 standardises a model for classification systems by suggesting various terminological Construction specific items, the ISO/PAS 12006-3 suggests a generic standard framework for object-oriented information. Thus, the ISO 12006-3 complements the ISO 12006-2 by providing a generic organisation model that is Construction-domain independent, for the ISO 12006-2 specific Construction terminology.

Indeed, ISO 12006-3<sup>1</sup> is a standard that defines a schema for a taxonomy model, which provides the ability to define concepts by means of properties, to group concepts, and to define relationships between concepts (Figure 1). Objects, collections and relationships are the basic entities of the model. The set of properties associated with an object provide the formal definition of the object as well as its typical behaviour. Properties have values, optionally expressed in units (ISO 12006-2, 2001), (ISO/DIS 12006-3, 2004).

The role that an object is intended to play can be designated through the model and this provides the capability to define the context within which the object is used. Each object may have multiple names and this allows for its expression in terms of synonyms or in multiple languages. The language name of each object must always be given in English (the default language). An object may also be named in terms of the language of the location in which it is determined or used. Objects may be related to formal classification systems through the provision of references.

---

<sup>1</sup> Please, find more information about this standard at <http://www.iso.org/iso/en/stdsdevelopment/tc/tclist/TechnicalCommitteeDetailPage.TechnicalCommitteeDetail?COMMID=1963&scopelist=>

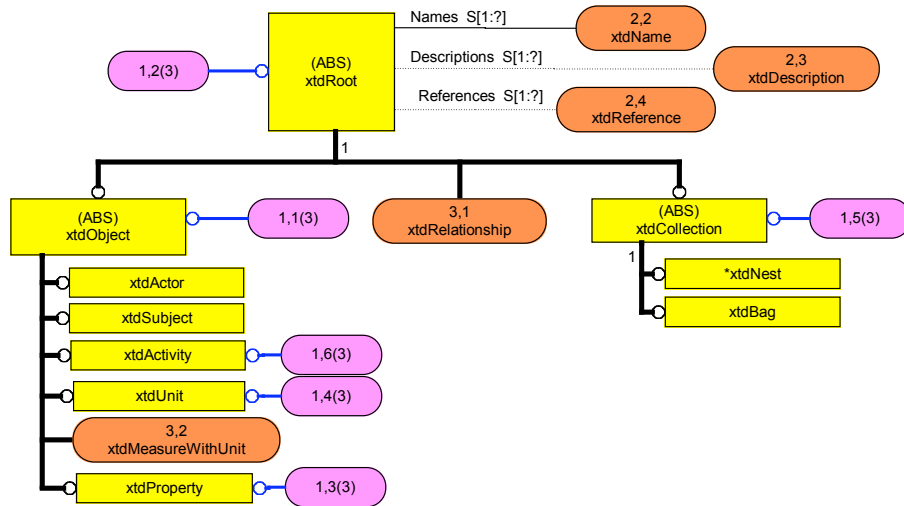


Figure 1 - Top level structure of ISO 12006-3 represented through the EXPRESS-G graphical language

The representation language of this standard is EXPRESS from ISO 10303 part 11. Even if the EXPRESS language supports the modeling of multiple inheritance, within the ISO 12006-3 model the intent is that *xtdRoot* entity provides the minimal attributes that any object needs to be considered a valid member of the object model. *xtdRoot* entity is the root of the entire ISO 12006-3 model inheritance graph (Figure 1). The *xtdRoot* entity is an abstract (ABS) one and it is specialized into one of *xtdObject*, *xtdRelationship* or *xtdCollection* entity. The *xtdObject* is the abstract concept holding all object classes in the model. Objects are divided into Subjects, Activities, Actors, Units, Measures with Units and Properties. Subjects and Activities are the things and processes that are described. The other classes are description classes related to other Objects and themselves through Relationships. From a practical viewpoint, this standard has been used in different projects across Europe, namely the LexiCon<sup>2</sup> (Netherlands), BARBi<sup>3</sup> (Norway), and the Edibatec<sup>4</sup> (France).

<sup>2</sup> The LexiCon is available at <http://www.stabu-lexicon.com/>

<sup>3</sup> The BARBi browser is available at <http://www.barbi.no/barbibrowser/search.do?action=invoke>

<sup>4</sup> The interested reader can find more information at <http://www.edibatec.org/Accueil/Default4.htm>

### 3. The Industry Foundation Classes (IFC)

The Industry Foundation Classes (IFC) model has been progressively developed by the International Alliance for Interoperability<sup>5</sup> (IAI) since 1995. There have been several releases of the model that have been implemented in software for data exchange and sharing across applications. From the date of IFC 2x release (October 2000), a part of the model has been protected against change. This part of the model is referred to as the ‘Platform’ and is that part of the model that was formally accepted as ISO PAS 16739 in November 2002 under the external harvesting procedures of ISO TC184/SC4 (Figure 2).

The IFC model is a response to interoperability requirements within building construction by a significantly large group of industry practitioners including government and other statutory bodies, clients, consultants and contractors together with a substantial number of software vendors. The primary target of the IFC Model is the interoperability among software applications within the building and construction market sector. IFC classes are therefore defined according to the scope and the abstraction level of software systems dealing with building and construction specific content. Such a model has been primarily developed to enable the exchange and sharing of Building Information Models (BIM) to increase the productiveness of design, construction, and maintenance operations within the life cycle of buildings. The IFC model therefore describes an object model with concepts (classes or “terms”), relations (as direct associations or objectified relationships), and attributes (or properties).

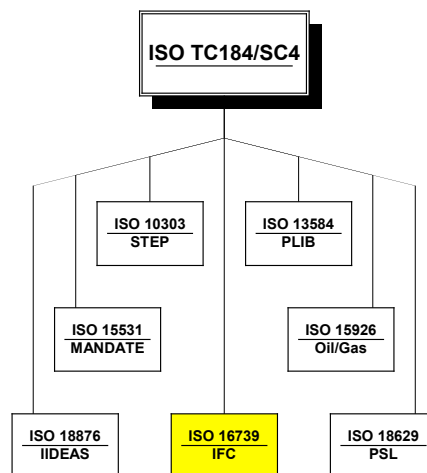


Figure 2 - IFC in the ISO family

IFC adopted and/or adapted certain parts of the STEP standards including:

1. Formal specification of IFC is in the EXPRESS language from ISO 10303 part 11;
2. Encoding of files for data exchange is undertaken using ISO 10303 part 21; and
3. The IFC model uses schema that have been adopted from the resource standards within ISO 10303, particularly parts 41, 42, 43 and 46.

<sup>5</sup> More information is available at <http://www.iai-international.org/>

IFC entities are modelled:

1. As predefined concepts (or IFC classes and relations), if such concepts are commonly used across many general and construction domain specific software products. These “statically” defined concepts allow to speed-up the exchange and sharing of information and reduces the risk of ambiguous interpretation;
2. As open concepts (of IFC proxies for classes, or IFC property sets for attributes), if such concepts are subjected to deep domain knowledge or are specific to certain localities. These “dynamically” defined concepts allow to enhance the scope of terms used within IFC by enabling the IFC model to be extended by the use of an external classification or taxonomy concept.

The predefined concepts form a well defined hierarchy of terms in the sense of a taxonomical hierarchy. At the leaf nodes of that hierarchy more dynamic means to extend the definitions (like enumeration for special occurrence types or type objects for common types) towards more granularities.

The IFC model comprises several schemas that are organised according to the layer they belong to. The schema *IfcKernel* defines the most abstract part within the IFC architecture which contains the most abstract IFC entity – the *IfcRoot*. Each entity defined in the core, interoperability or domain layer of the IFC model inherits (over some intermediate steps) from the *IfcRoot* entity. It provides for the fundamental properties of identification, ownership and change information, and optional label attribution. There are three fundamental entity types in the IFC model derived from the *IfcRoot*, that form the first level of specialization within the IFC class hierarchy, namely (Figure 3):

1. **Objects** are the generalisation of any semantically treated item within the IFC model. An object is the abstract supertype – *IfcObject* – and stands for all physically tangible items (e.g., wall, beam or covering), physically existing items (e.g., spaces), or conceptual items, (e.g., grids or virtual boundaries). It also stands for processes (such as work tasks), for controls (such as cost items), for resources (such as labour resource), or for actors (such as persons involved in the design process), and so on. An object gets its context information from the relationships it is involved in.
2. **Relations** are the generalization of all relationships among items that are treated as objectified relationships in the IFC model. A concept of relationships is the objectified relationship – *IfcRelationship* – which is the preferred way to handle relationships among objects. This allows to keep the relationship specific properties directly in the relationship object and to uncouple the relationship semantics from the object attributes. The introduction of the objectified relationships also allows the development of a separate subtype tree for the relationship semantics.
3. **Properties** are the generalization of all characteristics (either types or partial type, i.e., property sets) that may be assigned to objects. The property definition – *IfcPropertyDefinition* – is the generalisation of all characteristics of objects. It reflects the specific information of an object type, versus the occurrence information of the actual object in the project context. The property definition gets applied to the objects using the concept of relationships.

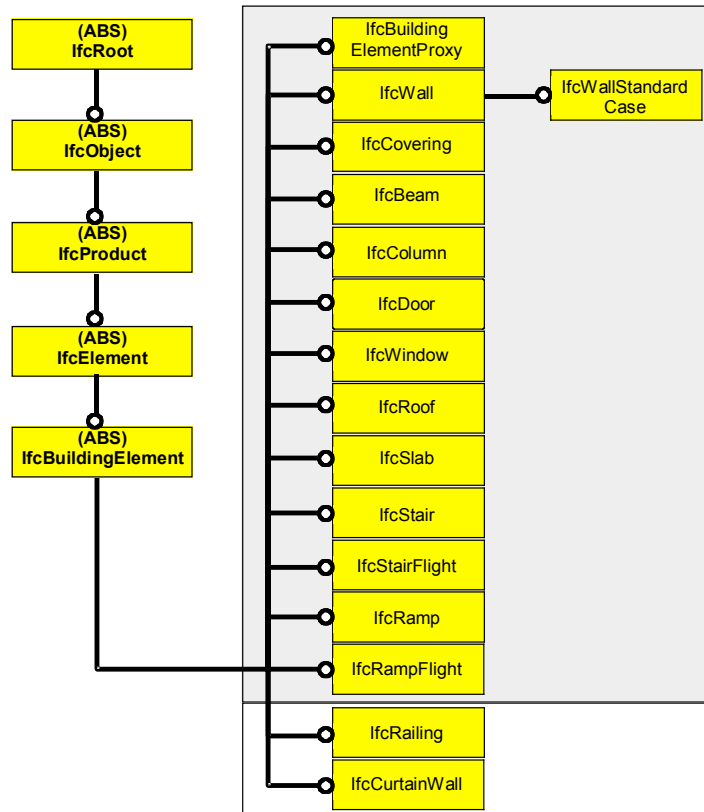


Figure 3 - A small example of the IFC model

The entities of the *IfcKernel* schemata are organised in a taxonomy, which is not the case for the entities belonging to other schemas. This is a consequence of the object oriented approach of the IFC model. In that case, there is no real need for an explicit organisation of the concepts; rather the *IfcEntities* are related among themselves through explicitly defined relations. For instance the concept of *IfcPerson* is not defined in the kernel (i.e., taxonomically speaking, there is no relation between *IfcPerson* and *IfcRoot*), but *IfcPerson* is related to *IfcActor* through its attribute *TheActor*, which is typed *IfcPerson*. In other words, *IfcPerson* inherits from *IfcActor* and this establishes the relation between the two entities.

From a “semantic perspective”, the IFC model per se cannot be considered an ontology or a taxonomy. But it can be considered has a model for an ontology or a taxonomy. Yet, part of it has been used to support reasoning and to exchange meaningful pieces of information among different software tools. It can also be used to represent domain information in a semi-formal way. This means the information representation is constrained by the EXPRESS language constructors while allowing to associate natural language descriptions.

## 4. Industry Foundation Classes for GIS (IFG)

The Industry Foundation Classes for GIS<sup>6</sup> (IFG) is a project being developed by the IAI (for the AEC community) and ISO TC211 (for the GIS community). The scope of the project is to define the interface between the AEC world and the GIS world and to develop strategies to improve interoperability and collaboration. In other words, the focus of the project is to make it possible to communicate relevant intelligent information from various GIS standards to CAD systems using IFC.

An objective of the project is to use the Industry Foundation Classes (IFC) model as the specification for the exchange of limited but meaningful information between GIS and AEC/FM CAD systems and vice versa. The aim is to use entities that are already established within the Coordination and Code Checking views of IFC 2x so as to be able to reuse insofar as possible the tools, techniques and capabilities already developed by vendors at the AEC side of the demonstration. New and enhanced capabilities are provided in the IFC model (version 2x3 release candidate 1) to recognise:

- specifically geographic elements that are not within the domain of AEC/FM activities
- coordinate system mapping
- qualified geometry (including contour lines, sight lines, survey points etc.)
- element proximity

It is expressly not an objective of the IFG project to create an information model that can completely represent geographic information. The GML (Geographic Markup Language) model is recognised as the standard for this purpose and the provision of capability within the IFC model through the project is intended to be sufficient to enable relevant information to be transformed between the IFC and GML formats.

The IFG project is supported by Statens Bygningstekniske Etat in Norway. Project management is undertaken by Boligprodusentene and technical management by AEC3 Ltd. Information about the project is freely available including model specifications, standards comparisons, use cases and test examples.

---

<sup>6</sup> The interested reader can find more information at [http://www.iai.no/ifg/IFG\\_Outline.htm](http://www.iai.no/ifg/IFG_Outline.htm)

## 5. The ISO 18629 – Process Specification Language

The Process Specification Language (PSL) standardization initiative includes the semantics for describing the fundamental concepts of manufacturing processes (Kemmerer, 2005). PSL defines a neutral representation for manufacturing processes. Process data is used throughout the life cycle of a product, from early indications of manufacturing process flagged during design, through process planning, validation, production scheduling and control. In addition, the notion of process also underlies the entire manufacturing cycle, coordinating the workflow within engineering and shop floor manufacturing.

PSL is being standardized within Joint Working Group 8 of Subcommittee 4 (Industrial data) and Subcommittee 5 (Manufacturing integration) of Technical committee ISO TC 184 (Industrial automation systems and integration). PSL is a multi-part standard that currently includes:

1. Part 1: Overview and Basic Principles
2. Part 11: Process Specification Language: PSL-Core.
3. Part 12: PSL Outercore
4. Part 13: Time and ordering
5. Part 14: Resources
6. Part 21: External mapping to EXPRESS
7. Part 22: External mapping to XML
8. Part 23: External mapping to UML
9. Part 41: Activity
10. Part 42: Time and State
11. Part 43: Ordering
12. Part 44: Resources
13. Part 2xx series: Translator implementation guidelines

### 7.3.3 ISO 18629 : Process Specification Language : PSL

ISO 18629 is the newest in the family of standards aimed at facilitating interoperability for industrial data integration (of products and processes) in industrial applications in TC 184. Standardised within a joint committee, ISO TC 184 SC4/SC5, PSL provides a generic language for process specifications applicable to a broad range of specific process representations in manufacturing and other applications. PSL is an ontology for discrete processes written in the Knowledge Interchange Format (KIF) (Genesereth and Fikes, 1992) itself an ISO candidate in ISO/JTC1, (Common Logic, 2004). Each concept in the PSL ontology is specified with a set of definitions, relations, and axioms all formally expressed in KIF. Relations specify types of links between definitions or elements of definitions ; axioms constrain the use of these elements. In addition, the PSL ontology is based on set theory, first order logic, and situation calculus (Etchemendy 1992). Because of this reliance on theories, every element in the PSL language can be proven for consistency and completeness (Gruninger 2003). At the time of this writing, approximately half of the PSL definitions, relations and axioms have been proven to be consistent with the base theories.

PSL is an international standard for providing semantics to the computer-interpretable exchange of information related to manufacturing and other discrete processes. Taken together, all the parts contained in PSL provide a language for describing processes throughout the entire production within the same industrial company or across several industrial sectors or companies, independently from any particular representation model. The

nature of this language makes it suitable for sharing process information during all the stages of production. The process representations used by engineering and business software applications are influenced by the specific needs and objectives of the applications. The use of these representation models varies from one application to another, and are often implicit in the implementation of a particular application. One of the manufacturing models on which the PSL ontology is built is provided by the information models of the ISO 15531 MANDATE standard (standardisation of manufacturing management information) (Cutting-Decelle et al., 2000-1), particularly for resource management.

A major purpose of PSL is to enable the interoperability of processes between software applications that utilise different process models and process representations. As a result of implementing process interoperability, economies of scale are made in the integration of manufacturing applications.

All parts in ISO 18629 are independent of any specific process representation or model used in a given application. Collectively, they provide a structural framework for interoperability. PSL describes what elements should constitute interoperable systems, but not how a specific application implements these elements. The purpose is not to enforce uniformity in process representations. As objectives and design of software applications vary the implementation of interoperability in a application must necessarily be influenced by the particular objectives and processes of each specific application.

#### 7.3.3.1. Architecture and content of ISO 18629

PSL (ISO IS 18629-1, 2004) is organized in a series of parts using a numbering system consistent with that adopted for the other standards developed within ISO TC184/SC4. PSL contains Core theories (Parts 1x), External Mappings (Parts 2x), and definitional extensions (Parts 4x). This discussion focuses on Parts 1x and 4x ; these parts contain the bulk of ISO 18629, including formal theories and the extensions that model concepts found in applications. Parts 1x are the foundation of the ontology, Parts 4x contain the concepts useful for modeling applications and their implementation. Table 7.1 presents the organization of ISO 18629. Except noted otherwise, PSL version 2.2 is presented.

Series	Number	Name
<b>Core theories</b>	ISO IS 18629-1	<b>Overview and Basic Principles</b>
	ISO IS 18629-11	<b>PSL-Core</b>
	ISO IS 18629-12	<b>Outer Core</b>
	ISO IS 18629-13	<b>Duration and ordering Theories</b>
	ISO IS 18629-14	<b>Resource Theories</b>
<b>External Mappings</b>	ISO 18629-2x	<b>Mappings to EXPRESS, UML, XML</b>
<b>Definitional extensions</b>	ISO IS 18629-41	<b>Activity extensions</b>
	ISO IS 18629-42	<b>Temporal and state extensions</b>
	ISO IS 18629-43	<b>Activity ordering and duration extensions</b>
	ISO IS 18629-44	<b>Resource extensions</b>

Table 7.1 : organization of ISO 18629

- Core theories (Parts 1x) :

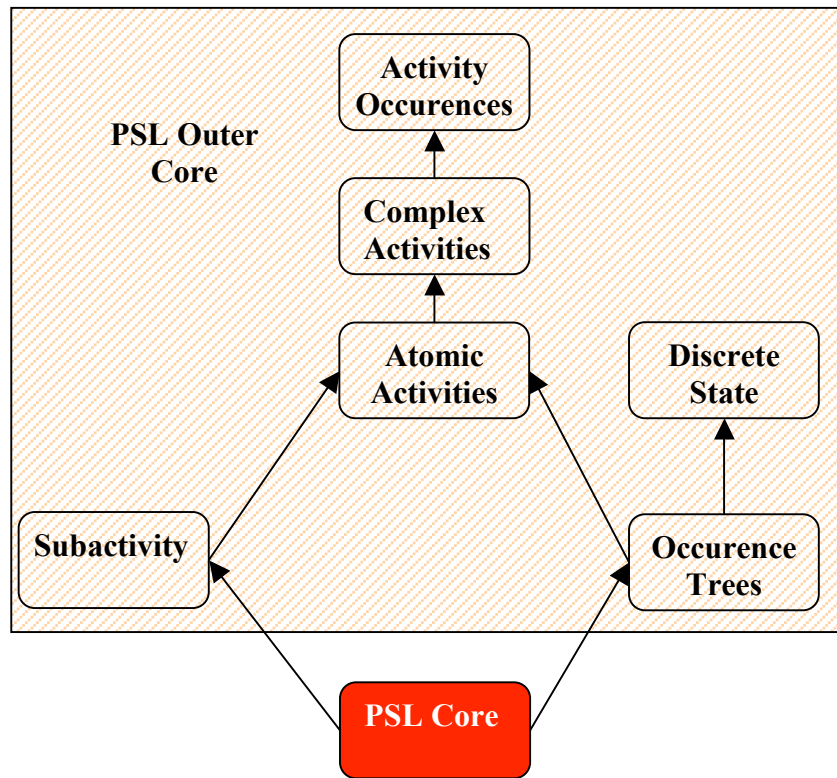
Core Theories include the PSL-Core, the Outer-Core, Duration and Ordering theories, Resource theories, and Actor and Agent theories. The core theories are contained in the parts

lx and based on first-order logic. They model basic entities necessary for building the PSL extensions. The PSL Core and Core Theories pose primitive concepts (those with no definition), function symbols, individual constants, and a set of axioms written in the language of PSL. Table 7.2 illustrates the primitives found in the PSL Core. These primitives and all the definitions in PSL are written in KIF for computer interoperability but the KIF writing is not shown here for the sake of readability. For KIF sentences expressing these relations and functions the reader is referred to the PSL Web site.

<b>PSL Core Primitives</b>	<b>Type</b>	<b>Informal definitions and axioms</b>
activity	relation	Everything is either an activity, an activity occurrence, a timepoint, or an object. Objects, activities, activity occurrences, and timepoints are all distinct kinds of things (disjoint classes).
activity_occurrence	relation	An activity occurrence is associated with a unique activity. But there are activities without occurrences.
timepoint	relation	Given any timepoint t other than inf-, there is a timepoint between inf- and t. Given any timepoint t other than inf+, there is a timepoint between t and inf+.
object	relation	An object participates in an activity at a given timepoint and only at those timepoints when both the object exists and the activity is occurring.
before	relation	The before relation only holds between timepoints. It is a total ordering, irreflexive, and transitive relation.
occurrence_of	relation	Every activity occurrence is the occurrence of some activity and associated with a unique activity.
participates_in	relation	The participates_in relation only holds between objects, activities, and timepoints, respectively.
beginof	function	The beginning of an activity occurrence or of an object are timepoints.
endof	function	The ending of an activity occurrence or of an object are timepoints.
inf+	constant	Every other timepoint is before inf+.
inf-	constant	The timepoint inf- is before all other timepoints.

**Table 7.1: Concepts in PSL Core (ISO IS 18629-11)**

Core theories are required to formally prove that extensions are consistent with each other, and with the core theories. The core theories are at the root of the PSL ontology against which every item that claims to be PSL compliant must be tested for consistency. They are a unique feature of PSL as no other standard in SC4 lends itself to formal, logic-based proof. Figure 7.12 illustrates concepts in the PSL Outer Core and their dependencies.



**Figure 7.12 : PSL Outer Core definitions and their dependencies (ISO IS 18629-12)**

Figure 7.13 extends Figure 7.12 to focuses on Duration, Ordering, and Resource Requirements theories.

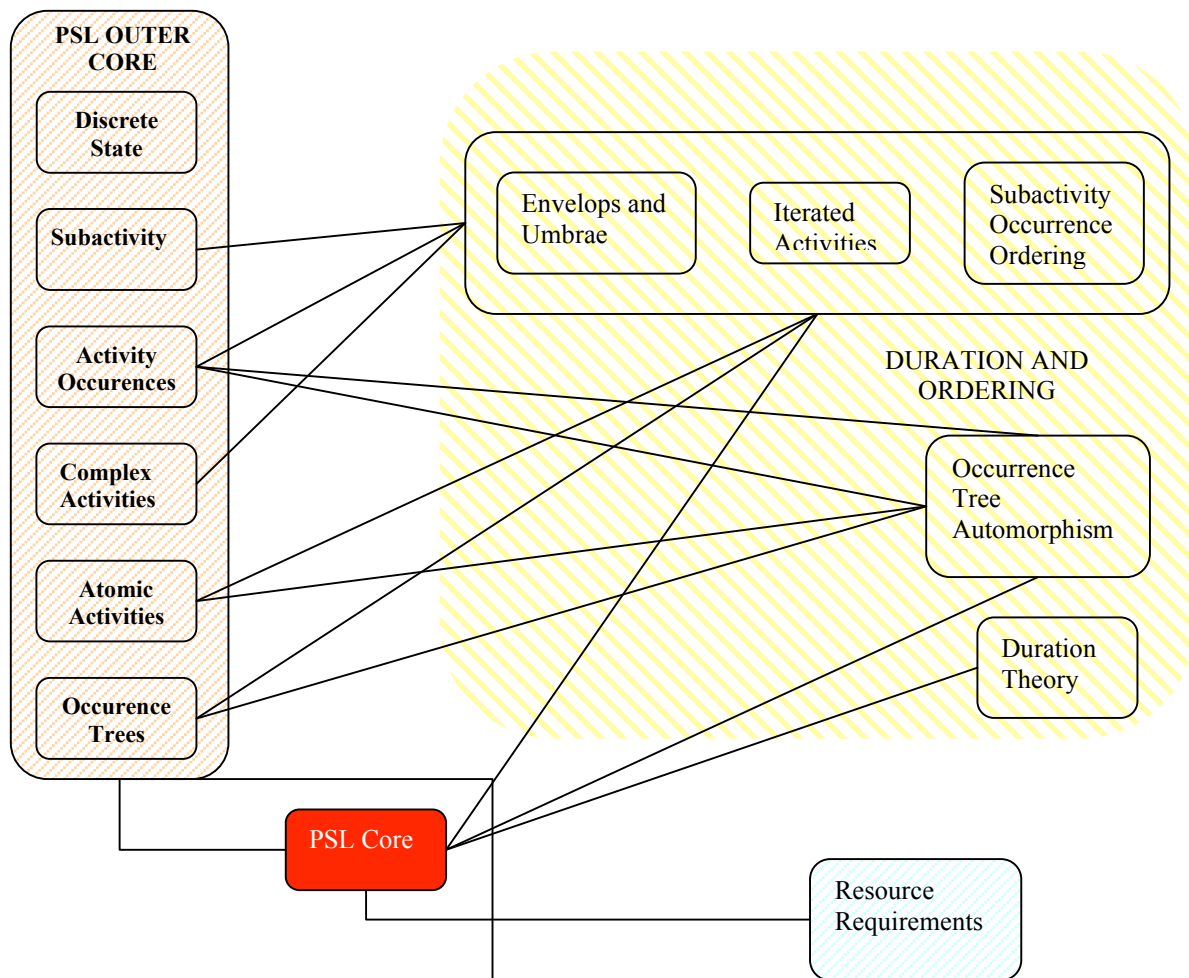
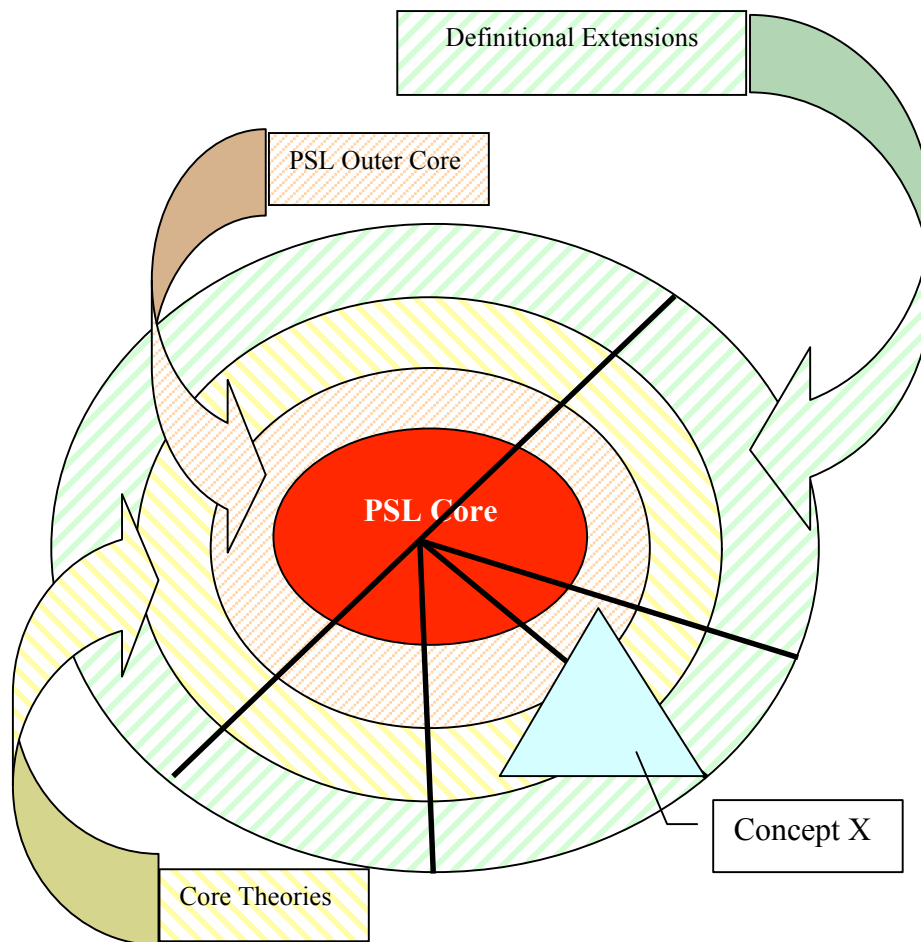


Figure 7.13 : Core and Outer Core Dependencies for Duration and Ordering, and Resource Requirements theories (ISO CD 18629-13), (ISO CD 18629-14)

Domain-specific Definitional Extensions (Parts 4x) : The extensions to the Core and Outer-core are used to represent the actual processes in an application. All terms in the extensions are given definitions using the set of primitive concepts axiomatized in the core theories. This ensures that definitions are consistent with PSL. A software application will typically use the concepts defined in the extensions, rather than the concepts in the Core and Outer core, which are necessary to define the extensions but have little expressivity.

In Figure 7.14, a definitional extension (Parts 4x) is represented as a slice of the pie. It specifies concepts and definitions for all kinds of (practical) concepts and are written using the Core, Outer core, and theories. Some definitional extensions also use concepts defined in other extensions. Figure 7.14 shows that a concept belonging to an extension (blue triangle) is specified using concepts of the Core, Outer Core, and another extension. But the Core and Outer Core alone are not sufficient to represent meaningfully an application’s semantics for the purpose of interoperability.



**Figure 7.14 : Architecture of the PSL ontology**

Table 7.3 gives examples of definitional extensions, and the core theories each extension relies upon. It is to be noted that the organization of the Extensions into Activity Extensions, Temporal Extensions, etc... is here for readability and ease of use of the standard. The organization itself does not affect the concepts in PSL: for instance a concept may be moved from one extension to another without affecting the PSL ontology or the concepts defined in the extension. In other words, to be a valid part of the PSL ontology extensions do not need to belong to one or another of the categories in the left column. However, each concept must conform to the Core Theories in the middle column.

<b>Definitional Extensions (Parts 4x)</b>	<b>Core Theories depended upon</b>	<b>Some examples of definitions</b>
Activity Extensions (Part 41) (ISO IS 18629-41)	Complex Activities	Deterministic and non-deterministic activities Concurrent activities Spectrum of activities
Temporal and State Extensions (Part 42) (ISO IS 18629-42)	Complex Activities, Discrete States	Preconditions, Effects Conditional activities Triggered activities
Activity Ordering and Duration Extensions (Part 43) (ISO IS 18629-43)	Sub-activity Occurrence Ordering, Iterated Occurrence Ordering, Duration	Complex sequences and branching Iterated activities Duration-based constraints
Resource Extensions (Part 44) (ISO IS 18629-44)	Resource Requirements Resource set theory Sub-activity Occurrence Ordering Resource Requirements	Reusable, consumable, renewable, and deteriorating resources, substitutable resources resource pools, Resource paths Processor activities

**Table 7.2: Examples of PSL concepts defined in extensions**

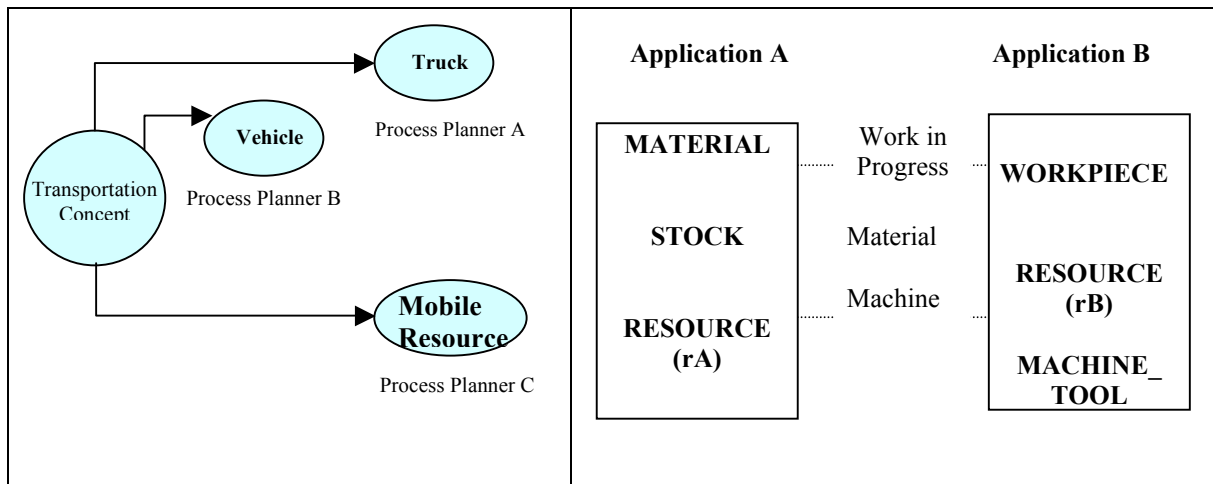
### 7.3.3.2. Interoperability with PSL and conformance to the standard

The main purpose of PSL is to establish a computer language for exchanging processes between software applications such as CAD, and project design software. As a specification language, PSL can be considered as a specification tool of the information and knowledge related to manufacturing management, as modelled by the MANDATE standard (ISO IS 15531-1, 2002).

#### 7.3.3.2.1 The challenges of interoperability

The obstacles to interoperability of data regarding syntax of two applications are common, and usually dealt with parsers. Obstacles due to semantic problems, i.e. problems about the “meaning” of a software object or entity are less visible. Lack of semantic reconciliation may introduce errors even if syntax mapping is correct. Without a standard like PSL, the semantic mapping may be performed in an ad hoc manner by a developer.

Figure 7.15 presents an example from the transportation industry, where a truck is represented as a vehicle, a mobile resource, or a truck. Delivery mechanisms not represented here may also include transportation for some applications. If there is no interoperability of processes, applications that use this terminology may be incompatible. This leads to re-inputting entries manually in the application chain. In the example in Figure 7.16, Material designates two different things: a Resource and a Work in Progress and a Resource. Resource a Material, a Machine-tool, and a Stock.



**Figure 7.15 : Incompatible Content Representation**

**Figure 7.16 : Semantic conflict for resource**

Syntactic interoperability does not resolve these conflicts, and decisions as to which concept in Application A matches a concept in Application B is left to the developer of parsers. The benefit of PSL is to formally encode each application's concept or vocabulary in a rigorous representation language. When two applications sharing data are expressed in PSL, the conflicts and semantic gaps are highlighted and a resolution is proposed. In essence, expressing the concepts of an application with PSL produces a detailed analysis of processes, and on this basis two applications can be reconciled.

#### 7.3.3.2.2 Interoperability and conformance

From the point of view of ISO 18629 two applications can inter-operate if they are conformant with the same set of ISO 18629 extensions. Software applications that claim conformance to PSL will:

specify processes from their application into the KIF language. This is the set of terms used by the application that refer either to processes in the application or relations among these processes.

provide translation definitions between their processes represented in KIF and PSL definitions.

implement syntactic translators between their applications and PSL process descriptions.

Another requirements not discussed in this chapter is that there exists a grammar using the same representation as PSL grammar for the application processes, using the Backus Naur form.

In practice, two applications do not exchange data about all their processes in one exchange. Only one or a set of processes at one time will exchange data. After identifying the concept to be exchanged, the steps outlined in the standard can be followed as :

the processes are defined and expressed using KIF syntax

the concepts contained in the processes (their names, relationship to other processes, conditional expressions) are further defined. In other words, the application's entities are given KIF definitions.

a translation is provided between the application's entities definition and PSL definitions.

At this point in the procedure, Applications A and B's processes have been expressed using

PSL terms and KIF syntax. Each has a one-to-one correspondence between their process definition and a PSL definition. On this basis, data for the relevant process can be exchanged.

Following this procedure does not allow a software application to claim conformance to PSL according to ISO 18629, but it is sufficient for process exchange with another application. To this purpose, the National Institute of Standards has implemented a “question wizard” (PSL, Wizard) to facilitate the expression of any process with PSL definitions and in KIF syntax. A user specifies a process in details by answering questions and checking boxes for their process. The wizard returns a definition for the process using PSL.

#### 7.3.3.2.3 User defined extensions

User defined extensions of PSL are extensions that introduce new primitive concepts. Typically, current extensions are sufficiently rich to express processes in existing software applications. However, the case where an application concept is not represented may arise. In this case, PSL can be extended to include a new extension by expressing it using the PSL Core, Outer Core, and definitions in existing extensions. The axioms in any extension that introduces new primitives must be consistent with the axioms of PSL-Core. User-defined extensions are needed when PSL is applied to domains that have not been yet dealt with in the extensions.

## 6. ISO 13584-PLIB

The ISO 13584 is a series of International Standards for the computer-sensible representation and exchange of part library data (Kemmerer, 2005). The objective is to provide a mechanism capable of transferring parts library data, independent of any application which is using a parts library data system. The nature of this description makes it suitable not only for the exchange of files containing parts, but also as a basis for implementing and sharing databases of parts library data.

Each International Standard in the ISO 13584 series is published as a separate part. These parts are grouped into one of the following series (The numbers between brackets are for unambiguous reference to the documents):

1. Conceptual descriptions (reserved part numbers are 10 to 19),
2. Logical resources (20 to 29),
3. Implementation resources (30 to 39),
4. Description methodology (40 to 49),
5. Conformance testing (50 to 59),
6. View exchange protocol (101 to 199), and
7. Standardized content (500 to 599).

### 7.3.2.1 Purpose

ISO 13584 (13584-1, 1999) specifies the structure of a library system which provides an unambiguous representation and exchange of computer interpretable parts library information. The data held in the library are a description that enables the library system to generate various representations of the parts held in the library. The structure is independent of any particular computer system and permits any kind of part representation. The structure will enable consistent implementations to be made across multiple applications and systems.

ISO 13584 does not specify the content of a supplier library. The content of a supplier library is the responsibility of the library data supplier. The library management system used in the implementation of the structure defined in ISO 13584, and any interface between this system and a user of the system is the responsibility of the library management system vendor and is not specified in ISO 13584.

### 7.3.2.2 Components of a Library System

The components which form a neutral library system may be split into a number of functional areas which are illustrated in Figure 7.6.

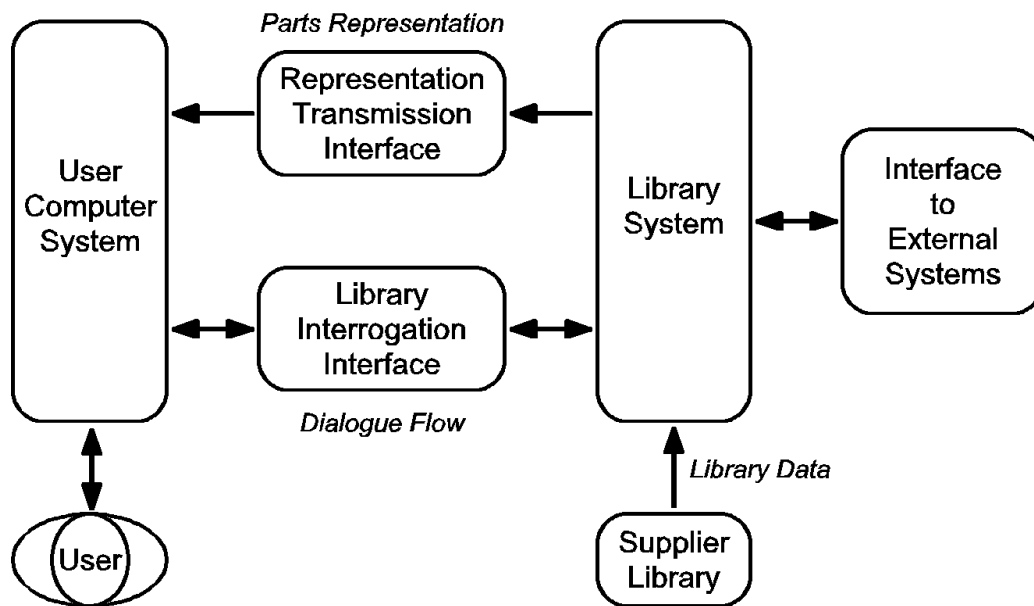


Figure 7.6 Functional areas of library usage

User to computer system communication : The interface between the user and his computer system is not defined in ISO 13584. This would be application dependent and form part of the user interface supplied by a vendor as part of a computer system.

Interface to External Systems : The interface between a library system compliant with ISO 13584 and other software systems are :

- a library Interrogation Interface : not defined in ISO 13584 but would be expected to provide facilities to select parts from the library and to define the orientation, position and representation category of the part selected ;
- a representation transmission interface, enabling the library system to send parts representations to the user computer system ;
- an input interface for library data, enabling the integration of supplier libraries within a library system.

### 7.3.2.3 Internal Structure of a Library System

A Library system consists of a Dictionary, Library Management System and Library Content as shown in Figure 7.7. The standard defines these modules by the requirements placed upon their functional behaviour. ISO 13584 does not standardise their implementation.

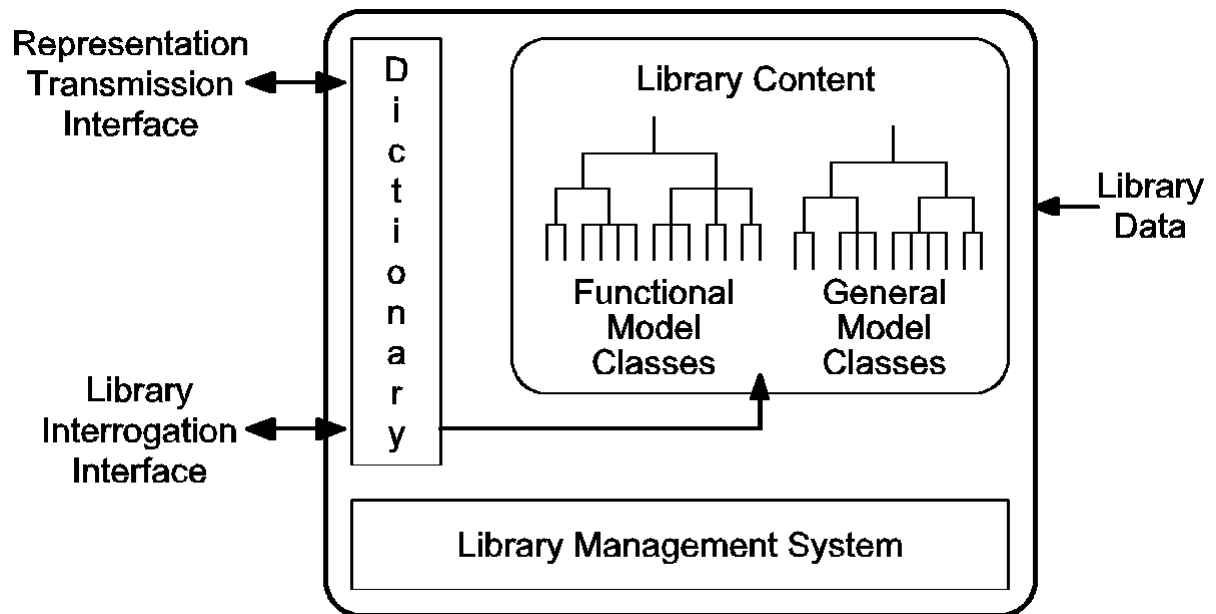


Figure 7.7 Library System

- Dictionary : consisting in a set of entries associated with a human-readable and computer-sensible representation of the meaning associated with each entry. The dictionary may be accessed by the user and referenced from library data. The Dictionary provides a referencing mechanism between library data obtained from different suppliers and enables the user to obtain an understandable view of the parts held in the library. The dictionary structure is specified in ISO 13584-42 (13584-42, 1998). A supplier library may contain only dictionary entries. These entries provide computer-referable identifiers for the concepts involved in some application domain.

- Library Management System : software system that enables the end user of the library to use the content of an integrated library and to load data into that library. The Library Management System is not standardised within ISO 13584.

- Library content : Library data are structured into classes in accordance with the object oriented paradigm. Three kinds of classes are considered in ISO 13584. The contents of the three kinds of classes may be exchanged using the structure and exchange format specified in P-LIB.

General model classes enable library data suppliers to provide the definition of a collection of cognate parts considered as a part family. Functional model classes enable library data suppliers to provide various representations (e.g. geometric, schematics, procurement data etc.) for these collections of cognate parts. Functional view classes enable the specification of the kind of representation provided in the different functional model classes. Some functional view classes are standardised in the view exchange protocol series of ISO 13584. A library data supplier may also provide the definition of their own functional view class. These three kinds of classes are illustrated in Figure 7.8.

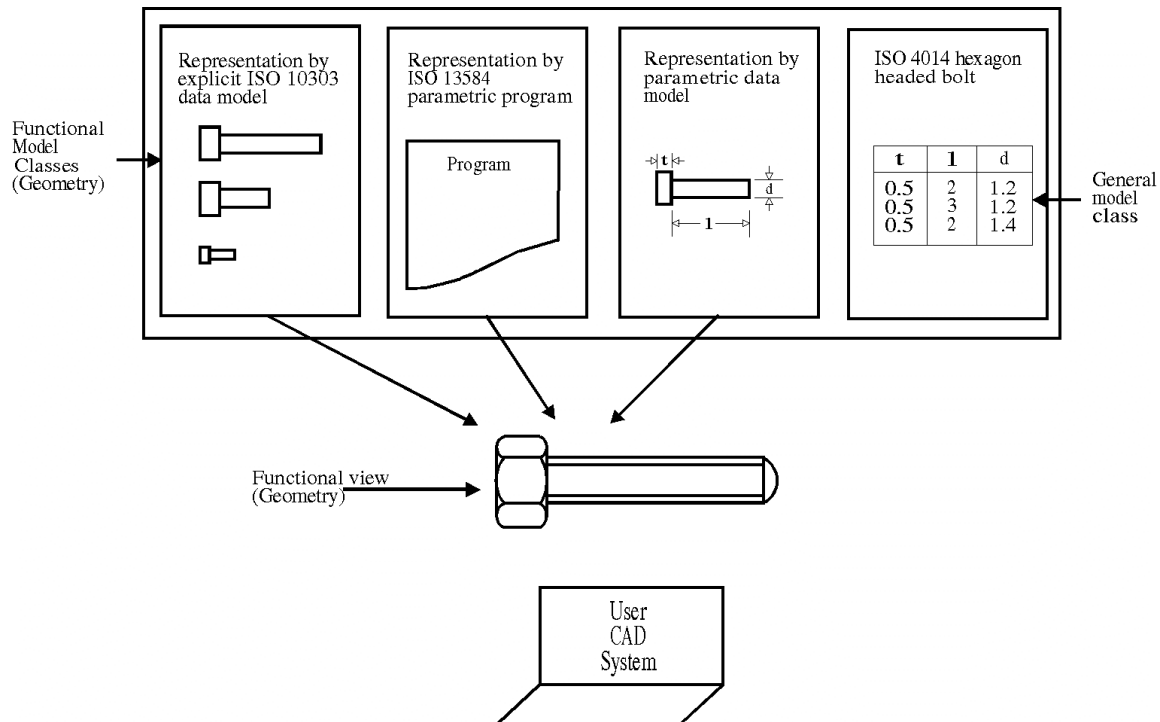


Figure 7.8 Structure of library contents

When a library consists only of a dictionary, it only defines the concept associated with each class and with the properties of each class. When a library also contains a library content, this content defines the set of instances contained by each defined class.

When the user CAD system is compliant with an ISO 10303 STEP application protocol(s), the provisions contained in ISO 13584 ensure that it is possible from a library content to generate a functional view that is compliant with an ISO 10303 application protocol.

#### 7.3.2.4 Fundamental Principles of the standard

ISO 13584 separates the representation of information held in a parts library from the implementation methods used in data exchange. The standard makes use of a formal data specification language, EXPRESS, to specify information about the structure of a library. ISO 13584 separates information about the structure of a parts library from the information about different representations of each part or family of parts in the library. ISO 13584 permits information about part representation to be specified by different standards, and includes mechanisms which enable references to such descriptions (Pierra et al. 1998), (Pierra et al. 2004).

#### 7.3.2.5 Structure of the ISO 13584 series of parts

ISO 13584 is divided into series of parts, each with a unique function. Each series may have one or more parts. The series are listed below with their numbering scheme :

- Conceptual Description - Parts 10 to 19
- Logical resources - Parts 20 to 29
- Implementation resources - Parts 30 to 39
- Description Methodology - Parts 40 to 49
- Conformance Testing - Parts 50 to 59
- View Exchange Protocol - Parts 101 to 199
- Standardised content - Parts 500 to 599

- Conceptual descriptions : they define the global conceptual framework and mechanisms developed to allow the portability of multi-supplier and multi-representation parts libraries, for exchanging and for updating. They present a problem domain analysis of the universe of discourse. They describe the concepts and choices made in the formulation of ISO 13584. The division of the whole task to be performed into a number of logical tasks that may be defined as a separate part of ISO 13584 is accomplished in the conceptual description series of parts.
- Logical resources : The information model of parts library is provided by a set of resources. Each resource is comprised of a set of data descriptions in EXPRESS, known as resource constructs. One set may be dependent on other sets for its definition. Some resources constructs from ISO 10303 may be used to define ISO 13584 resources constructs. All the ISO 13584 resource constructs are defined in one part of the logical resources series. These resources may be used, but not modified, in a view exchange protocol.
- Implementation resources : Each representation category may require a representation transmission interface to be implemented on a receiving CAD system to be able to interpret part models and to generate part views. The implementation resources specify the standardised representation transmission interfaces which may be referenced by a view exchange protocol. Each part of this series either specifies an interface, with the requirements for its implementation, or specifies the requirements for the implementation of one interface specified in other Standards.
- Description methodology : providing rules and guidelines for library data suppliers, who may be standardisation organisations, part suppliers or functional model suppliers. These rules are intended to ensure consistency of a user Library. They are mandatory for the standardisation committees, in charge of specifying standardised dictionary data. They provide optional guidelines for part suppliers or functional model suppliers.
- Conformance testing : providing test cases and a set of requirements that any implementation shall meet before being accepted as conforming to this Standard.
- View exchange protocol : specifying one set of requirements for the exchange of one representation category of parts. Several view exchange protocols may refer to the same representation category. A view exchange protocol may introduce different options that may be selected by an implementation. The options are termed conformance classes. In this case the requirements of the view exchange protocol are specified separately for each conformance class.
- Standardised content : It is intended to progressively define standardised dictionary entries which may be referenced by supplier libraries. This work will be done inside different standardisation committees following the methodology specified in the description methodology series of parts of ISO 13584. The parts of the standardised content specify the standardised dictionary entries corresponding to various application areas.

#### 7.3.2.6 Use of library parts in product data

An ISO 13584 conforming exchange context provides for the exchange of library data intended to be stored in a user library. An ISO 10303 STEP conforming exchange context provides for the exchange of product data.

Three levels of interactions have been identified between these two levels of exchange.

- Level 1 : All information about a part generated in System A will be transferred to System B by means of ISO 10303 (see Figure 7.9).

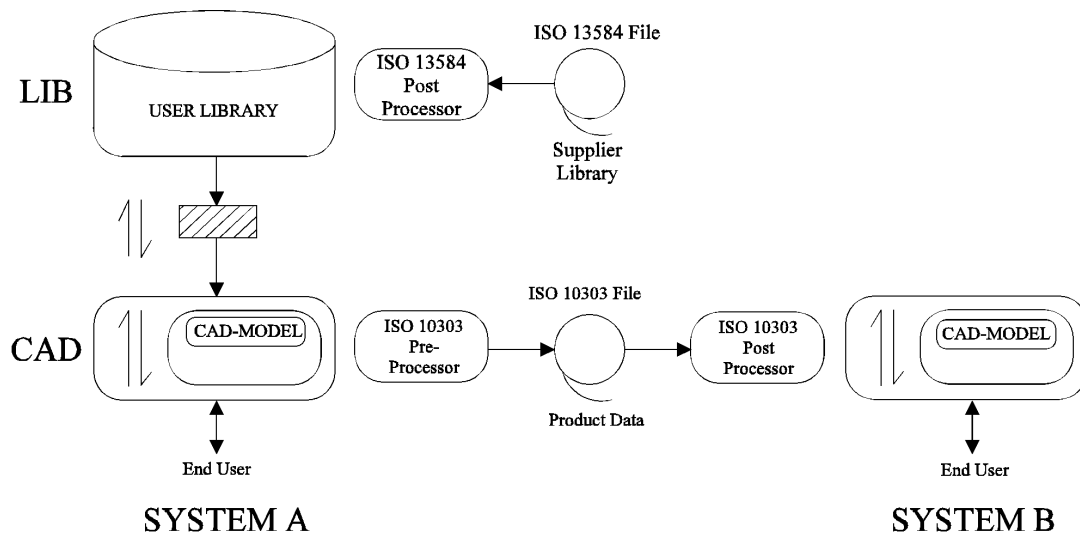


Figure 7.9 Libraries and product data exchange (level 1)

- Level 2 : Only that information is transferred from System A to System B, which is necessary to generate the same part from a Library 2 of the receiving System B at the required position and orientation. Library 1 and Library 2 both contain all the information about the part (see Figure 7.10).

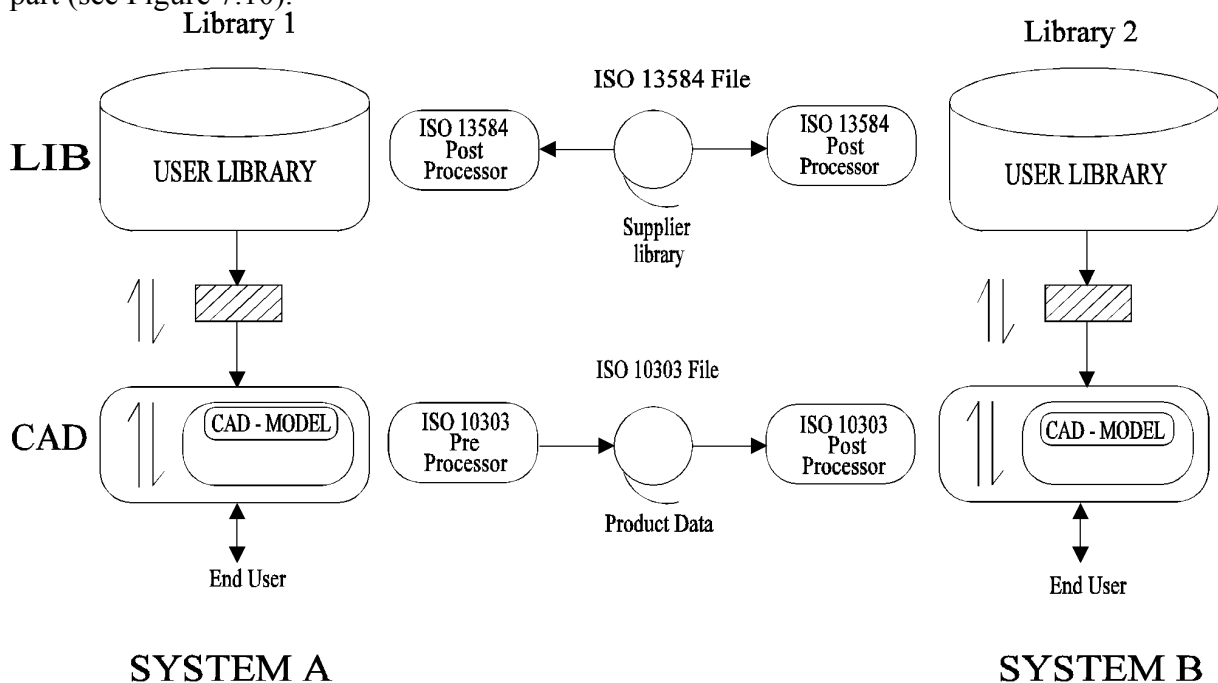


Figure 7.10 Libraries and product data exchange (level 2)

- Level 3: That information is transferred from System A to System B which is necessary to generate the same part information on the receiving System B without any assumption about the content of Library 2. This means that the transferred data also contains a subset of Library

1 (see Figure 7.11).

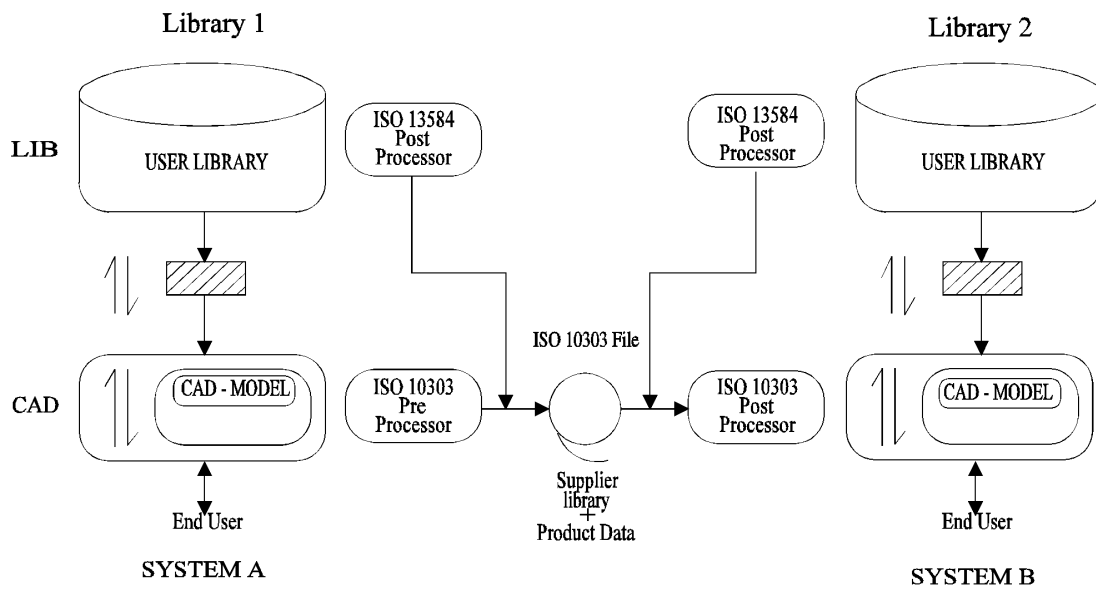


Figure 7.11 Libraries and product data exchange (level 3)

The information models specified in ISO 13584 are intended to enable these three levels of interaction.

### The PLIB ontology model

A PLIB ontology model has the following characteristics (17):

- *Conceptual*: each entity and each property are unique concepts completely defined. The terms (or words) used for describing them are only a part of their formal definitions.
- *Multilingual*: a globally unique identifier (GUI) is assigned to each entity and property of the ontology. Textual aspects of their descriptions can be written in several languages (French, English, Japanese, etc.). The GUI is used to identify *exactly* one concept (property or entity).
- *Modular*: an ontology can reference another one for importing entities and properties without duplicating them. Thus providing for autonomy of various sources that do reference a shared ontology.
- *Consensual*: The conceptual model of PLIB ontology is based on an international consensus and published as international standards (IEC61630-4:1998, ISO13584-42:1998) (for more details see (16)).
- *Unambiguous*: Contrary to linguistic ontology models (16), where partially identical concepts are gathered in the same ontology thesaurus with a similarity ratio (affinity), each concept in PLIB has with any other concepts of the ontology well identified and explicit differences. Some of these differences are computer-interpretable and may be used for processing queries, e.g., difference of measure units, difference of evaluation context of a value.

## References

- ISO 12006-2. 2001. Building construction -- Organization of information about construction works -- Part 2: Framework for classification of information.
- ISO/DIS 12006-3 version 3. 2004. Building construction -- Organization of information about construction works – Part 3: Framework for object-oriented information.
- Kemmerer, S.J. (2005): Exchanging Technical Product Data – the story of ISO TC 184/SC4. NIST <http://www.mel.nist.gov/msidlibrary/doc/ISOprocess.pdf>
- ISO IS 18629-1, “Industrial automation systems and integration – Process specification language – Part 1: Overview and basic principles”, 2004
- ISO IS 18629-11, “Industrial automation systems and integration – Process specification language – Part 11: PSL-Core”, 2004
- ISO IS 18629-12, “Industrial automation systems and integration – Process specification language – Part 12: Outer Core”, 2004
- ISO IS 18629-13, “Industrial automation systems and integration – Process specification language – Part 13: Duration and ordering theories”, 2006
- ISO IS 18629-14, “Industrial automation systems and integration – Process specification language – Part 14: Resource theories”, 2006
- ISO IS 18629-41, “Industrial automation systems and integration – Process specification language – Definitional extensions: Part 41 : Activity extensions”, 2006
- ISO IS 18629-42, “Industrial automation systems and integration – Process specification language – Definitional extensions: Part 42 : Temporal and state extensions”, 2006
- ISO IS 18629-43, “Industrial automation systems and integration – Process specification language – Definitional extensions: Part 43 : Activity ordering and duration extensions”, 2006
- ISO IS 18629-44, “Industrial automation systems and integration – Process specification language – Definitional extensions: Part 44 : Resource extensions”, 2006
- Pierra G. and Sardet E., Potier J. C., Battier G. and Derouet J. C., Willmann N. and Mahir A., “Exchange of component data : the PLIB (ISO 13584) model, standard and tools”, Proceedings of the CALS EUROPE'98 Conference, 16-18 September 1998, Paris, France.
- Pierra G., Dehainsala H., Ait-Ameur Y., Bellatreche L., Chohon J., El-Hadj Mimoune M., "*Base de Données à Base Ontologique : le modèle OntoDB*", Rapport de recherche 2004-02 (soumis à publication), 2004.
- ISO 13584-1, Industrial automation systems and integration – Parts Library : Overview and

fundamental principles, 1999

ISO 13584-42: Industrial Automation Systems and Integration, Parts Library : Methodology for Structuring Parts Families, 1998.

[16] G. Pierra. Context-explication in conceptual ontologies: The plib approach. *in Proceedings of 10th ISPE International Conference on Concurrent Engineering: Research and Applications (CE'03) : Special Track on Data Integration in Engineering*, pages 243–254, July 2003.

[17] L. Bellatreche, G. Pierra, N. Xuan Dung, D. Hondjack. An a Priori Approach for Automatic Integration of Heterogeneous and Autonomous Data Sources, Submitted to: *Computers in Industry*, 2004