

Chapter 4

Specifications for system implementation

Business transaction framework

In real-world business situations, data transfer between two business activities can be carried out by people without the aid of computers; for example, through paper documents containing typed, printed or handwritten information. Today, however, most information exchanges are made via computers and networks. Some are automatic, some require human involvement. In PSLX specifications it is assumed that all information exchanges between business activities require explicit representation schema to enable transfer to be conducted partially or totally by computers. There are two types of implementation methodology for data exchange in a business transaction:

(1) Storage type communication

The first method is storage type communication to exchange data. In this type, data are stored once in a database, and then sent or received via the database. Relational databases are popular tools for all kinds of business information system, and are also useful in conducting this type of data communication between different business activities. The PSLX-RDB common schema is a standard specification for system implementation to facilitate this data exchange.

(2) Messaging type communication

The second method of data exchange is messaging type communication. This is where two application programs operating on different business activities exchange data directly. In this case, the data in each message are generated solely for the purpose of data exchange, and then scrapped. Basically, one message is generated for one purpose, on one business transaction. PSLX defines a standard specification for this type of data exchange in the PSLX-XML common schema.

As a general rule, different business activities need to share common business protocols in order to make a data exchange. "Business protocol" defines rules of communication at business level, including exception processes that apply when communication fails. More importantly, it needs to deal with the issue of the data being able to be properly interpreted, as intended by the initiator of the data exchange.

Both the PSLX-RDB common schema and the PSLX-XML common schema feature a data schema that represents a particular data structure, in which the correct meaning of all elements should be understandable in each business context. To support clear understanding of schema semantics, PSLX defines PSLX Ontology.

PSLX Ontology

An ontology is a fundamental element in representing the semantics of information related to problems in a certain domain. Since the target domain is decision-making processes in planning and scheduling for manufacturing industries, PSLX specifications have defined PSLX Ontology, which provides a basis for common understanding of information about planning and scheduling.

PSLX Ontology is defined by a structure which corresponds to the unique and fundamental structure underlying problems of planning and scheduling. In other words, the semantics of an ontology is represented implicitly by the structure, with respect to its similarity to reality. A term named for an ontology is simply a label used to identify the ontology in the semantic structure, and is not precisely concerned with the correct meaning. An additional intuitive explanation for an ontology is also helpful, but not essential.

In PSLX specifications, an ontology is represented using UML class diagrams. However, this doesn't mean that the ontology is a kind of class defined in object-oriented modeling technologies. The ontology is used to define the semantics underlying classes. In class diagrams, the ontology is classified into several types, which can be observed provisionally in the form of stereotypes. The types of ontology are introduced in order to properly establish the meaning. An ontology does not have any attributes or methods.

Information that represents any elements in planning and scheduling for manufacturing should be defined using PSLX Ontology, to ensure common understanding of problems between different parties. In other words, all information objects dealt with in business activities should have specific relations to the ontology structure. A PSLX specification shows details of the process to describe the relationships, using UML diagrams.

Aspects and structure of ontologies

PSLX Ontology can be classified according to the following aspects, each of which has common features. The classifications are used to decide a stereotype for a class that is directly defined using one of these ontologies.

(1) Engineering aspect

This aspect represents engineering knowledge or know-how that is obtained through long experience in the domain. Information in this aspect is stored in advance of its usage. Then, it is repetitively retrieved and applied to problems. This aspect has four ontologies: Function, Capability, Item, and Event.

(2) Scheduling aspect

The scheduling aspect deals with the relationship of actions with time and space, which are necessary to realizing actions and operations in the physical world. In other words, an object in this aspect is an instance of an object in the engineering aspect with respect to particular time and space. Time includes the past, present and future. This aspect has four ontologies: Operation, Task, Lot, and Action.

(3) Temporal aspect

This aspect represents physical features from the viewpoint of changes in the time horizon. Physical phenomena of an object can be described temporally by objects in this aspect. Physical phenomena are the results of interaction between a physical object and operations defined in the scheduling aspect. In this aspect, three ontologies exist: Capacity, Inventory, and Change.

(4) Physical aspect

The physical aspect is defined for physical objects that exist in reality. This may contain a conceptual object if it is not duplicatable. Usually, objects in this aspect occupy certain places during certain periods of time in the real world. The objects in this aspect do not have a tense, because they are qualified by objects in the temporal aspect. Resource is an ontology that exists in this aspect.

(5) Order aspect

Every action and operation on plant floors is created and executed with respect to orders. This aspect is defined for orders in all kinds of situations. One order may create other orders. All intentional changes in enterprises are provided by orders. Information flows in business processes can be defined as order propagation flows, which execute actions along the flows. This aspect has three ontologies: Work order, Lot order, and Task order.

(6) Plan aspect

Business activities activated by order propagation in a business process are evaluated as to whether or not they are balanced and consistent with others from a medium- or long-term viewpoint. The plan aspect deals with conceptual parameters, objectives and goals associated with a certain period of time. Constraints, preferences and functional relations can be applied to these objects to obtain an optimum solution. Ontologies included in this aspect are Production plan, Capacity plan, and Inventory plan.

(7) Party aspect

The party aspect is defined for objects that can perform autonomously with their own decisions and preferences. The granularity of an object depends on the fact that final goals should be shared within the object. This aspect can also represent a type of ownership of resources in

business processes between enterprises. This aspect has three ontologies: Maker, Supplier and Customer.

(8) Time aspect

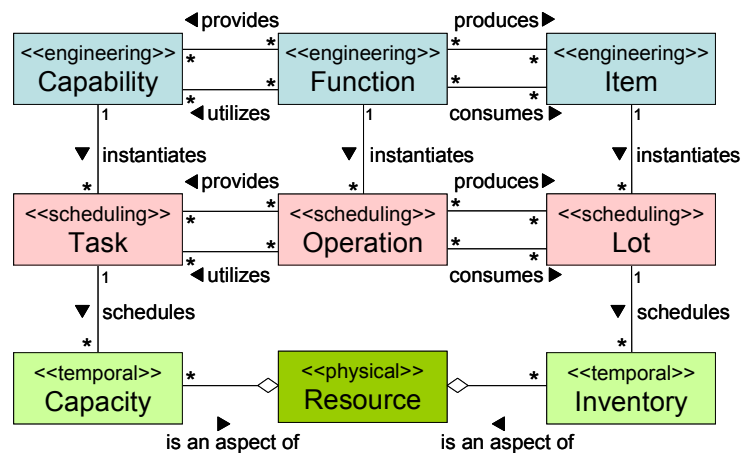
Although it cannot literally define “time”, the time aspect defined by PSLX is simply used to provide useful classifications for concepts related to time, where definition of time is assumed. This aspect has three ontologies: Time point, Time period, and Time span.

(9) Space aspect

Similar to the definition of time, a definition of space is also difficult to establish, but is a highly important aspect of planning and scheduling problems. The space aspect is defined to classify basic elements of the concept related to space. This aspect is useful in determining the granularity of objects from a spatial or geographical view. There are three ontologies in this aspect: Position, Region, and Distance.

PSLX defines 27 ontologies in the nine aspects described above. Ontologies have relations with each other, representing their semantics depending on their positions. The semantics are shown in the following figures, with explanations of the special meaning of each position.

Fig. 4-1
PSLX
ontology (1)



First, Fig. 4-1 shows several basic aspects of ontology and their relations. Engineering, scheduling, temporal, and physical aspects are connected by relations with particular roles and cardinalities. For example, it can be seen that ontologies in the scheduling aspect are instantiated by ontologies in the engineering. Ontologies in the temporal aspect exist between the ontologies in the scheduling and physical aspects.

The right column in Fig. 4-1 represents something that is produced or consumed by production. This is an aspect of an ontology that can fix a particular space in the real world. On the other hand, the left side of Fig. 4-1 represents something that is utilized by production. This is an aspect of an ontology that can fix a particular time, so that the ontologies cannot move across different periods. For example, capacity of resources is utilized within a particular period of time.

Fig. 4-2
PSLX
ontology (2)

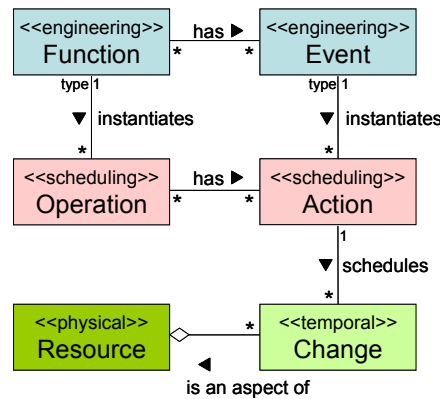


Fig. 4-2 is a complementary chart to Fig. 4-1, in which the right column is added. The additional column represents the most primitive view of those three aspects. While the left-hand column in Fig. 4-2 (center column of Fig. 4-1) associates with the concept of time span, this column associates with time point. For example, a concept of Operation can have a concept of Action between its start and end if it has several different phases.

Fig. 4-3
PSLX
ontology (3)

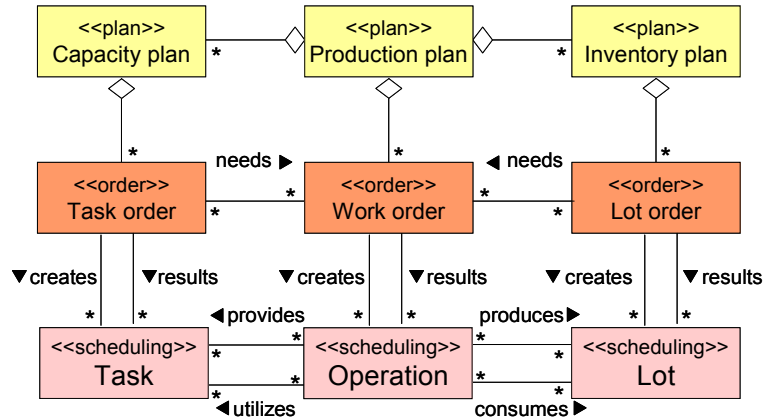


Fig. 4-3 represents relations of the other two aspects: plan and order. As shown in Fig. 4-3, ontologies in the order aspect create an ontology in the scheduling aspect, while ontologies in the plan aspect are aggregated from the ontologies of orders. Similar to Fig. 4-1, this chart also has special meanings for the right column and left column. In addition, it can be noted that there is no ontology in the plan and order aspects that corresponds to the right column of Fig. 4-2.

APS domain model

An ontology is a universal definition that can be shared with different standards in this area. However, it is so abstractive that users who represent something using the ontology take time to understand correctly. Therefore, it is necessary to have more detailed vocabularies for general use in production planning and scheduling that allow an object to be chosen easily.

An abstract data model has a comprehensive set of objects and relations for this purpose. This is independent from the form of information system implementation. This section introduces the APS domain model, which is a common reference model for abstract data models. Each manufacturer can easily define a particular abstract data model according to its unique business model by using the APS domain model.

The APS domain model is represented in UML class charts. All classes in the model should have some connections with PSLX Ontology, which can be drawn as a path in the chart. In conclusion, the definition of class in the APS domain model should be a:

- (1) Class corresponding to an ontology;
- (2) Sub-class of (1);
- (3) Aggregate class of (1); or
- (4) Class that has direct relations with a class of (1), (2), or (3).

In addition to the ontological relations, two classes as defined above (1) - (4) can have particular relations, each of which must have a role and cardinality. Classes in the APS domain model, as well as abstract data models, do not have attributes or methods.

The APS domain model is partially shown in Fig. 4-4 to Fig. 4-7. First, Fig. 4-4 has an operation class that corresponds to the ontology, and its sub-classes: Manufacturing, Setup, Inspection, Transport, Administration, Product design, Plant engineering, Maintenance, Ship goods, Receive goods, Issue inventory, and Store inventory. An asterisk beside a stereotype denotes that the class directory connects to the ontology.

Fig. 4-4
Sub-classes of operation

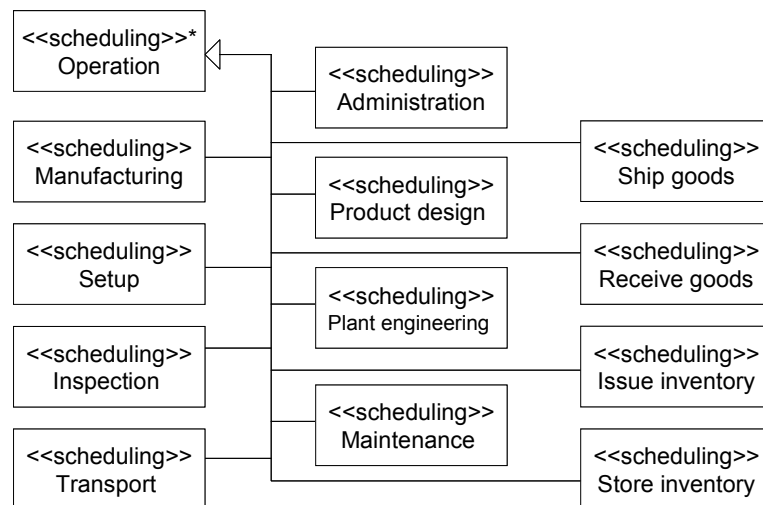
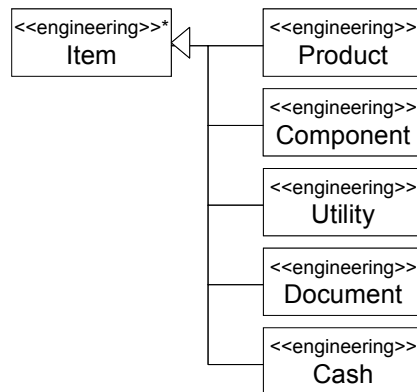


Fig. 4-5 shows an item class and its sub-classes, where the item class has a direct relation to the ontology. All the classes in the chart have the engineering aspect, and represent some features of identical substances or concepts at an abstract level. Sub-classes of item include Product, Component, Utility, Document, and Cash.

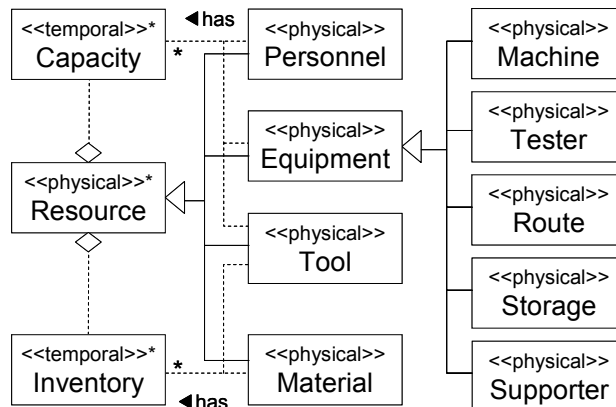
Fig. 4-5
Sub-classes of
item



A class corresponding to the resource ontology and its sub-classes are illustrated in Fig. 4-6. Sub-classes of resource are defined as Personnel, Equipment, Tool, and Material. Equipment has further sub-classes of Machine, Tester, Route, Storage, and Supporter,

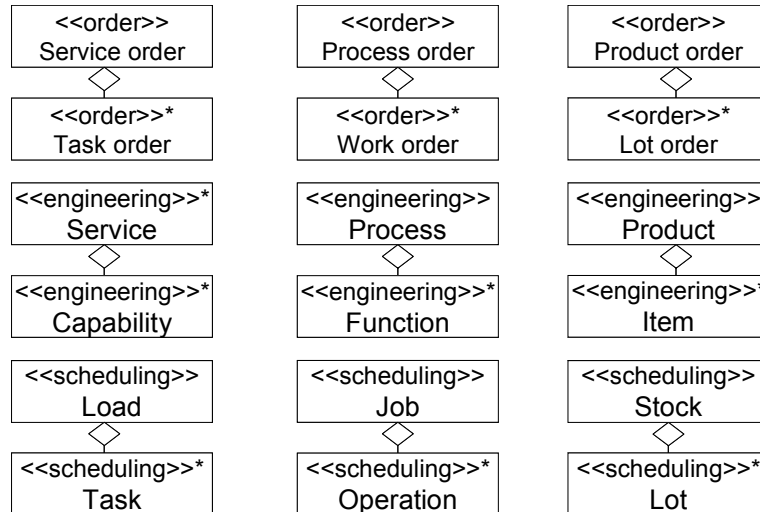
Sub-classes of resource have two different functional characteristics. One is consumable resources such as materials. The other is kinds of non-consumable resources that can remain during production. Most resources can be classified according to one of those characteristics, although a few classes have both. Fig. 4-6 represents the characteristics by connecting to either Capacity or Inventory, while classes connecting to Capacity are non-consumable, and classes connecting to Inventory are consumable.

Fig. 4-6
Sub-classes of resource



Many classes that directly connect to an ontology have an aggregate class, as shown in Fig. 4-7. These aggregate classes are useful when a decision-making process has hierarchy, as in planning and scheduling. In fact, most of the basic classes in Fig. 4-7 are typically used in scheduling problems, and the aggregate classes are useful representation for planning problems.

Fig. 4-7
Aggregation classes for planning



In the order aspect, we find Task order, Work order, and Lot order at the level that directly corresponds to the ontology. These have aggregate classes of Service order, Process order, and Product order, respectively. Similarly, in the engineering aspect, Service, Process, and Product are aggregate classes of Capacity, Function, and Item, respectively. Finally, the scheduling aspect has Load, Job, and Stock as the basic ontological

classes of Task, Operation, and Lot, respectively.

In PSLX specifications, all the classes in the APS domain model are explained in detail using UML class charts.

Standard schema for implementation

While the APS domain model deals with the correctness of knowledge representation in various kinds of practical business activities, the data schema referred to in a computer system needs to have a very application-specific view. In general, each information system has its own implementation schema to represent business data in the form of each application system. In order to realize interoperability of the data, there needs to be another kind of data model that shares the common format of the implementation-specific schemas. This common schema represents information in the APS domain model. PSLX defines two common schemas for implementation: the PSLX-RDB common schema, and the PSLX-XML common schema.

Similar to the APS domain model, the two common schemas defined in PSLX are based on PSLX Ontology. The common schemas should be created using the steps described below. In other words, all the classes in the common schema can be explained by referring to the APS domain model. One difference to the APS domain model is that the common schemas can have attributes.

- (1) Change the name of a class
- (2) Merge two classes
- (3) Create a class instead of a relation
- (4) Create an attribute instead of a relation
- (5) Delete a class
- (6) Add a new class
- (7) Add a relation between two classes

(8) Add an attribute to a class

The steps above have the same sequence as those in the list. Furthermore, steps never have to be redundant. In other words, it should not to have other shorter sequences of steps that get the same schema from the APS domain model. The detailed rules of the schema generation are defined in the PSLX specifications.

PSLX-RDB common schema

A PSLX-RDB common schema is a [shared][common] schema for a business transaction in storage type communication. In this case, the PSLX-RDB schemas correspond to views of a particular RDB, while the views are created by SQLs. A unique feature of a PSLX-RDB common schema is that every class in the schema does not have an explicit relation with others. The relations behind the classes in the schema are defined by referring to the APS domain model. Every class and attribute can be identified by knowing the schema generation steps from the APS domain model.

As mentioned earlier, RDBs implemented in information systems need to provide a view corresponding to a class in the PSLX-RDB schema as a common interface between different applications. A systems engineer who is responsible for the implementation needs to know the relations between a table in the RDB schema and a view that corresponds to a class in the PSLX-RDB common schema. He or she also needs to know about more detailed relations between each field of the table and a field in a view that corresponds to an attribute of the class in PSLX-RDB schema.

There are a huge variety of classes in a PSLX-RDB common schema, because of the number of different business activities. Therefore, some useful categories have been introduced, as follows. The details of the classes are described in the PSLX specifications.

(1) Master classes

Master classes directly relate to classes in the APS domain model. Data in a class act as master data, which is referred to by business activities. This group includes customer master, destination master, supplier master, capacity master, personnel master, equipment master, tool master, storage master, operation master, transport master, process master, product master, material master, item master, area master, work center master, and route master.

(2) Relation classes

Relation classes represent relations between two classes in the APS domain model. Because of the technical constraints of RDB, relations need to be another class if the cardinality is more than one. Relation classes include bill of operation, bill of resource, bill of process, bill of area, bill of material, resource table, process table, precedence table, and loading table.

(3) Order classes

Order classes cover various kinds of orders in business activities, corresponding to the order class and its sub-classes in the APS domain model. The following classes belong to this type: customer order, purchase order, production order, work order, ship order, receive order, issue order, and store order.

(4) Planning classes

Planning classes are used to represent data in terms of plans, such as goal parameters in business activities and aggregated results of those activities. These are defined with respect to a certain unit of time phases. The planning classes include: production plan, master production schedule, material requirement plan, rough-cut capacity plan, capacity

requirement plan, demand plan, inventory plan, and logistics plan.

(5) Scheduling classes

Scheduling classes deal with certain actions on the time horizon, defining information associated with any kind of operation on plant floors. All the classes in this group contain information about a particular time. The scheduling classes include: factory calendar, equipment calendar, personnel calendar, resource capacity, lot data, task data, product inventory, material inventory, resource load, area load, and tracing data.

(6) Performance classes

Finally, performance classes are defined to classify classes that represent results of any business activities. Some of these are obtained by monitoring activities on plant floors, while others are results of commitments, as in a business situation. This group includes: order pegging, lot pegging, work result, production result, resource result, receive result, ship result, store result, issue result, send invoice, and receive invoice.

PSLX-XML common schema

A PSLX-XML common schema is defined for data exchange between different applications in different business activities by means of messaging type communication. This type of communication is becoming popular even between different computer environments. Examples include Web-services technologies for Web application servers and Web clients on the Internet, and XML/EDI for inter-enterprise information exchange using a flexible data format rather than the fixed length format of conventional EDI.

Classes defined in the PSLX-XML common schema have hierarchical

structures, because XML basically is a data specification language for documents. In the schema, network type relations are implicitly defined by the identifiers of each data item and their references. However, those relations should be managed in the APS domain model, outside of the formal definition of the XML schema. The identifier is described as an attribute of a class, while the reference data are also an attribute in the other class. These relations are usually one-way.

The top-level classes in the PSLX-XML common schema correspond to the level of transactions between business activities. A feature of this schema is that a class that corresponds to the same class in the APS domain model can have additional constraints, depending on the context of the business activities. The number of classes in the top level is large, but the number of classes in the second and lower levels is relatively small.

The following are all the classes and their categories of the PSLX-XML common schema, excluding classes in the top level.

(1) Primitive elements

Primitive elements correspond to the PSLX ontology. These are described in the second level of hierarchy (top level of a message). There are nine elements: customer, supplier, item, resource, function, operation, order, lot, and task.

(2) Administrative elements

Administrative elements are also described in the top level of a message. These are used as common subsidiary data among the primitive elements. In this group, scale and stone (milestone) are defined.

(3) Administrative properties

Administrative properties, on the other hand, define some subsidiary data

as an attribute of the primitive elements. These data are required by decision-making processes. This group includes description, priority, and display.

(4) Relational elements

Relational elements represent relations between the primitive elements. Ten classes are included: produce, produced, consume, consumed, assign, assigned, predecessor, successor, pegging, and partof.

(5) Property elements

This category is for any classes that represent attributes of the primitive elements. All of the attributes can have a temporal aspect. There are seven classes: spec, location, progress, load, stock, available, and calendar.

(6) Temporal elements

Temporal elements define a class or sub-class of event[s] in the APS domain model. There are five classes in this category. Classes named start, end and event are related to an operation, while release and duetime are temporal elements related to an order.

(7) Basic data elements

The basic data elements are defined for any basic data that are necessary to describe a value of parameters. The basic data elements have five classes: qty, price, char, duration, and time.

(8) Data auxiliary elements

The data auxiliary elements are additional information for the basic data elements. Classes in this category define a range of data. Seven classes exist: min, max, earliest, latest, shortest, longest, and enumerate.

The classes in the top level of the PSLX-XML common schema are defined in PSLX specifications in accordance with the business collaborations and transactions. One example is a class named “pslx”, which can incorporate every class described above. The detailed information of classes in the second and lower levels, which are the main concern of XML tag representation, is also defined in the PSLX specifications.