Representation of Data in Industrial Systems Using STEP – a Case Study

Claudia RAIBULET and Claudio DEMARTINI
E-mail: raibulet@athena.polito.it, demartini@polito.it
Dip. di Automatica ed Informatica, Politecnico di Torino, Corso Duca degli Abruzzi, 24, 10129, Turin, Italy

Abstract **- One of the actual problems identified in industrial systems is the representation of data related to heterogeneous devices with different proprietary characteristics. A data model designed to provide a common view of information associated to various types of devices is strongly required. An immediate advantage of such an issue would be a common view of data for activities that regard installation, configuration, maintenance, and management of the devices in a system. This paper describes the Distributed Data Repository (DDR) model that aims to provide a uniform perspective for the distributed components of an industrial system, independent of any implementation detail. To achieve this goal, STEP standard has been used. The specification of the conceptual model is based on STEP AP 212 that defines the elements necessary to describe an industrial system. The paper presents an object-oriented implementation approach that makes use of STEP Part 21 file format and STEP Standard Data Access Interface.**

I. INTRODUCTION

Heterogeneous devices with different proprietary description and representation of data, interconnected via heterogeneous protocols in distributed systems, require specific mechanisms for their manipulation. This fact generates difficulties for the access and the management of their related data within distributed industrial systems.

In this context, a fundamental requirement of the Distributed Data Model (DDR) model has been the definition of a common information model that provides a uniform view of the devices' data for any application related to the installation, configuration, maintenance, and technical management of the system's distributed components.

DDR aims to define a conceptual model that describes heterogeneous device data, independent of any model of storage and/or user application. DDR model acts as a middleware between the physical model and the application level. For the specification of DDR, STEP/ISO 10303 [1, 6, 7, 8] standard has been used. STEP is a standard for the representation and the exchange of product data in industrial automation systems. DDR conceptual model is based on STEP AP 212 [3], which defines a set of entities necessary to describe an industrial plant. The specification of the model makes use of STEP EXPRESS [2] formal language.

An object-oriented implementation of DDR is also presented in this paper. It is based on two of the STEP implementation forms: Part 21 file format (Part 21) [4] and Standard Data access Interface (SDAI) [5]. Part 21 represents a model of storage, while SDAI is a standard way for the access and the manipulation of data, which has been specified using EXPRESS language.

The remainder of this paper is organized as follows. Section 2 contains a description of DDR model. Section 3 presents the representation of the devices' data within the DDR model. Section 4 describes an object-oriented implementation approach for DDR. The conclusions and the further work are dealt within Section 5.

II. THE DDR MODEL

The objective of the DDR is to provide a general information model that can describe distributed systems during the different stages of their life cycle. The DDR model unifies the actual representation of the devices within complex distributed systems. Heterogeneous devices and their links are described according to a common and unified approach, regardless of their specific implementation.

STEP/ISO 10303 AP 212 [3] has been chosen for the specification of the DDR model because it includes all the concepts necessary to describe a plant (e.g., devices, interfaces, networks, functions, variables, etc). The objective of the AP 212 is to specify the tasks of the operational activities of each plant and the relationships among them. Attention is directed to the abstraction of generic activities and data structures that provide the basis on which efficient activity models and data models can be built up. AP 212 is used not only to create data models, to represent and to exchange data, but also to build distributed data repositories.

A. The Conceptual Model

The conceptual model of DDR defines an abstract view of the data associated to an industrial plant in terms of an object-oriented paradigm: the model describes objects, their attributes, and the relationships among them. The basic abstract objects identified for the description of DDR are:

Function block: it is a software functional unit comprising an individual, named copy of a data structure

and associated operations specified by a corresponding function block type [3]. It has associated an algorithm described by an external reference (e.g., a state chart representation). The input of a function block is specified by zero or more input parameters. The output of a function block is specified by zero or more local variables used by its algorithm. It is implemented by one or more physical devices [3].

Local variables and parameters: they are software entities associated with a function block that keep the values used by the function block's algorithm. Local variables are defined within the scope of a function block. Parameters are input/output events that do not carry associated data (i.e., boolean values) or input/output variables [3]. From the physical point of view they are considered input/output ports.

Logical structure: it represents the way in which function blocks are connected within the system. The connections are specified in terms of connected parameters. There are two types of connections: one-to-one (a direct connection between an output port and an input port) and one-to-many (a direct connection between an output port and two or more input ports). Connections are implemented by physical networks.

Localization: it describes the physical position and arrangement of the physical entities, in particular, devices and network segments. In addition, the physical position of a device group that forms a logical entity can be represented.

Physical structure: it describes the way in which devices are connected within the system. One-to-one and one-to-many connections are specified in terms of ports.

Allocation: it describes aspects regarding the intended use of the equipment within the system, expressing relationships between function blocks and devices, between logical and physical structures.

B. ISO 10303 AP 212 – Subset Definition

DDR uses a subset of ISO 10303 AP 212 entities in order to describe the objects and the relationships identified in the conceptual model. A subset of the AP 212 entities that have been selected for the design of the DDR model is briefly presented below:

Allocation – defines information that specifies the relationship between the objects designated in both the function_structure UoF and the product_structure UoF [3], describing the allocation of function blocks to devices.

Classification – describes concepts needed to categorize the packages of information contained in the product data.

Designation – describes concepts used to identify equipment items and function blocks.

External_reference – provides a way to specify information not included in AP 212.

Function_structure – specifies the concepts used to describe the functional structure of the system. The functions may be either primitive or composed from other functions.

Functional_connectivity – specifies all those concepts that are required to specify the logical connectivity of the system, i.e., connectivity among function blocks.

Network_allocation – describes information that specifies the allocation of functional connectivity to physical connectivity established by the equipment used in the system.

Physical_connectivity – specifies information required in order to describe the connectivity of the equipment used in the system.

Product_structure – specifies the concepts needed for the description of the equipment used in the system implementation.

.III. THE DESCRIPTION OF DEVICES IN THE DDR MODEL

STEP EXPRESS language [2] has been chosen for a formal description of the DDR model. The language focuses on the definition of entities, which represent the objects of interest. The definition of an entity is in terms of its properties, which are characterized by the specification of a domain and the constraints on that domain. Relationships are implemented as entity-valued attributes. Both entities of a relation contain an attribute that has the type of the partner entity. To define a common scope for a collection of entities and/or type definitions, EXPRESS provides the schema concept. A data model contains one or more schemas.

```
ENTITY device ABSTRACT SUBTYPE OF
        ( ONEOF (single_device) );
    id: object_reference_designation;
    definition: instance_definition_select;
    description: OPTIONAL STRING;
END_ENTITY;

ENTITY single_device SUBTYPE OF  ( device );
END_ENTITY;

ENTITY design_discipline_item_definition
        SUPERTYPE OF (assembly_definition);
    name: OPTIONAL STRING;
    associated_item_version: item_version;
    id: STRING;
END_ENTITY;

ENTITY item_version;
    description: OPTIONAL STRING;
    version_id: STRING;
    associated_item: item;
END_ENTITY;

ENTITY item_identification;
    id: STRING;
END_ENTITY;
```

Figure 1. EXPRESS Definition for the Device Model

Figure 1. contains the EXPRESS schema for the representation of a device.

In the DDR model, a device is represented by the *single_device* [3] entity, which is a subtype of the abstract *device* entity. The *device* object must be associated with a unique identifier, which is represented by its *id* attribute. A device definition has to be specified using the *design_discipline_item_definition* [3] entity. The definition carries information related to the *device* type. More than one device can share the same definition, that is, more than one device of the same type can be present in the system. The *design_discipline_item_definition* entity specifies the version information by means of the *item_version* [3] object associated with it. An *item_version* must have associated an *item* [3] object whose unique identifier is represented by the *item_identification* [3] object related to the *item* by means of its *id* attribute.

IV. AN OBJECT-ORIENTED IMPLEMENTATION APPROACH FOR DDR

An object-oriented implementation of DDR makes use of the Java environment. The EXPRESS model associated to the DDR has been translated in terms of Java concepts: packages, interfaces, and classes. The effective data associated to the EXPRESS model has been stored in a STEP Part 21 file format [4] (see § 4.1).

The translation of EXPRESS entities into Java concepts has been fulfilled using ST-Developer Tool [9]. The tool provides also a Java - Standard Data Access Interface (SDAI) [5] binding that supports the access to STEP data (stored for example in a Part 21 file format) within Java applications. SDAI bindings to Java consists of a set of session level interfaces, which are common for all EXPRESS schemas and a mapping used to generate a set of early-bound container classes for each particular EXPRESS schema. Session level classes provide the ability to interface with SDAI repositories, to open, and to save SDAI models. Early-bound classes provide fine-grained data access.

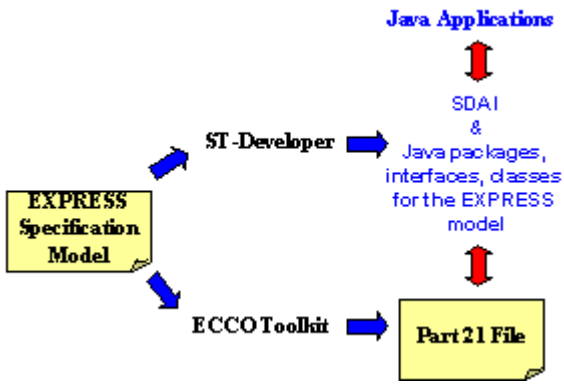

Figure 2. An Object Oriented Implementation Approach for the DDR

A Java model equivalent to an EXPRESS model is obtained as follows. An EXPRESS schema [2] is translated into a Java package that consists of interfaces and classes corresponding to the entities defined within the schema. Each entity [2] is translated into an interface that contains public methods for reading and writing the values of the attributes defined within the entity, and a Java class that implements this interface. The entity attributes correspond to protected variables in the Java classes and can be accessed using the *get* and *set* associated methods (see Figure 3.).

Part 21 files represent the physical database. It is transparent for the applications the way in which the effective data is stored. For the user, all the information is stored in terms of objects and classes. The access to Part 21 data is done through the SDAI. A Part 21 file can be generated using ECCO Tollkit. [10].

Figure 2. resumes the way in which a Java implementation of the DDR EXPRESS model has been successfully fulfilled.

A. Encoding the Data Model

The EXPRESS specification of the DDR represents the formal description of the data model, while Part 21 is the physical format for storing the effective data associated to the model. Part 21 is one of the implementation methods provided by STEP which describes the structure of an ASCII file that contains the effective values of the elements presented in an EXPRESS data model. The file is organized in such a way that both reading and writing of data, related to a specific data model, are permitted. The structure and the content of a Part 21 file are created according to the EXPRESS data model associated to a specific example.

B. Standard Data Access Interface

STEP provides the specification of a SDAI, which allows the access and the manipulation of data stored in a database through a standard interface. The SDAI provides the facility to access data as if the physical model used for data storage were identical to the conceptual model.

The SDAI provides:

- a standard Application Program Interface (API) for the data defined using the EXPRESS specification language;
- a consistent data access environment for use in software development; the details of the underlying database are hidden and of no concern, because data is available via the SDAI;
- independence of the underlying database technology; the interface defined by the SDAI is based on the conceptual data model, so that different database

technologies can be used for data storage without having to alter the interfaces used between the applications and the database.

The Java interface and the Java class equivalent to the *device* entity (see Figure 1.) specified using EXPRESS language, are presented in Figure 3 below.

```
package SDAI.DDR;

public interface Device extends App_inst
{
  SDAI.DDR.Object_reference_designation
   getId();
  void setId
   (SDAI.DDR.Object_reference_designation v);
  SDAI.DDR.Instance_definition_select
    getDefinition();
  void setDefinition
   (SDAI.DDR.Instance_definition_select v);
  java.lang.String getDescription();
  void  setDescription(java.lang.String v);
}

public class DeviceC extends App_instC
  implements Device
{
  public DeviceC() {}
  public DeviceC(Model_contents mc)
    { mc.addInstance(this); }
  public final java.lang.String
   getDescription() { return Description; }
  public final void
   setDescription(java.lang.String v)
    { Description = v;}
  public final
   SDAI.DDR.Instance_definition_select
    getDefinition() { return Definition; }
  public final void setDefinition
   (SDAI.DDR.Instance_definition_select v)
    { Definition = v; }
  public final
   SDAI.DDR.Object_reference_designation
   getId() { return Id; }
  public final void setId
   (SDAI.DDRS.Object_reference_designation v)
    { Id = v; }
  protected java.lang.String
    Description =null;
  protected
   SDAI.DDR.Instance_definition_select
     Definition = null;
  protected
   SDAI.DDR.Object_reference_designation
    Id = null;
}
```

Figure 3. Java Interface and Java Class Equivalent to the Device Entity

V. CONCLUSIONS AND FURTHER WORK

This paper has presented a data model designed to provide a common view of the information associated to various devices of an industrial system. The main objective of DDR has been to find a common model for the representation, the access, and the manipulation of heterogeneous devices interconnected via different protocols in distributed systems. The model has been also verified and validated on an actual example in the IAM-Pilot laboratory in ENEL in Milan.

DDR makes use of STEP standard technology. The DDR conceptual model has been specified using a formal language named STEP EXPRESS. The model is based on a set of entities defined within STEP AP 212. Starting from this specification, an object-oriented approach has been successfully implemented.

An entity-relationship implementation of DDR is under development. Further work will be related to the development of a Java interface that allows the manipulation of DDR data in both implementations. The interface will be used in order to compare the performances of the two different implementation approaches of the DDR conceptual model.

REFERENCES

[1] ISO 10303-1: 1994, Industrial automation systems and integration – Product data representation and exchange – Part 1: Overview and fundamental principles.

[2] ISO 10303-11:1994, Industrial automation systems and integration – Product data representation and exchange – Part 11: Description methods: The EXPRESS language reference manual.

[3] ISO/IEC DIS 10303 – 212: 1998, Product Data Representation and Exchange, Application Protocol: Electrotechnical Design and Installation.

[4] ISO 10303-21: 1994, Industrial automation systems and integration – Product data representation and exchange – Part 21: Clear text encoding of the exchange structure.

[5] ISO/WD 10303-22: 1996, Industrial automation systems and integration – Product data representation and exchange – Part 22: Implementation methods: Standard data access interface specification.

[6] Udo Nink, “Using the STEP Standard and Databases in Science”, Proceedings of the Ninth International Conference on Scientific and Statistical Database Management, 1997.

[7] Thu-Hua Liu, Amy J. C. Trappey, Fu-Wei Chan, “A Scheduling System for IC Packaging Industry Using STEP Enabling Technology”, IEEE Transactions on Components, Packaging, and Manufacturing Technology, 1997.

[8] STEP, http://www.nist.gov/sc4/www/stepdocs.htm

[9] ST-Developer Tools, http://www.steptools.com/

[10] ECCO Tool Kit, http://www.pdtec.de/