

AC.1 – Research, Strategic Advice and Implementation Planning Group

REFERENCE GUIDE – “The Next Generation of UN/EDIFACT”

SOURCE: AC.1
STATUS: Committee Document
ACTION: For review and action by GE.1

AC.1- RESEARCH, STRATEGIC ADVICE AND
IMPLEMENTATION PLANNING GROUP

REFERENCE GUIDE

“THE NEXT GENERATION OF
UN/EDIFACT”

- AN OPEN-EDI APPROACH USING IDEF MODELS & OOT -

* REVISION 10 *

TABLE OF CONTENT

MANAGEMENT SUMMARY 4

0 FOREWORD 6

1 INTRODUCTION..... 8

1.1 GENERAL 8

1.2 DEFINE A STANDARDS DEVELOPMENT PROCESS TO MEET CUSTOMER NEEDS..... 9

 1.2.1 *The Enterprise Architecture* 9

 1.2.2 *Scope* 10

 1.2.3 *Modeling*..... 11

 1.2.4 *Object Oriented Technology*..... 11

1.3 CO-ORDINATION AND LIAISON..... 12

1.4 NORMATIVE REFERENCES 12

2 BACKGROUND..... 13

2.1 OPEN-EDI 13

 2.1.1 *Shift the Focus* 15

2.2 MODELING THE BUSINESS PROCESS 15

 2.2.1 *Introduction* 15

 2.2.2 *Activity Modeling*..... 16

 2.2.3 *Process Modeling*..... 17

 2.2.4 *Data Modeling*..... 19

2.3 OBJECT ORIENTED TECHNOLOGY 20

 2.3.1 *Introduction* 20

 2.3.2 *Thinking of Objects in an Analogy* 21

 2.3.3 *Objects and Classes* 21

 2.3.4 *Attributes and Behavior* 22

 2.3.5 *Object Relationships* 23

 2.3.5.1 *Associations*23

 2.3.5.2 *Aggregations*24

 2.3.5.3 *Inheritance*24

 2.3.5.3.1 *Creating a Class Hierarchy*.....25

 2.3.5.3.2 *How Inheritance Works*.....26

2.4 THE SHIFT TO OBJECT-ORIENTED TECHNOLOGY 28

3 OPEN-EDI FRAMEWORK & SCENARIO EXAMPLES..... 29

3.1 OVERVIEW OF FRAMEWORKS..... 29

3.2 CATALOG ORDER FRAMEWORK 30

 3.2.1 *Detailed Explanation of BUSINESS FUNCTION A1* 33

 3.2.2 *Detailed Explanation of BUSINESS FUNCTION A2* 34

 3.2.3 *Detailed Explanation of BUSINESS FUNCTION A3* 36

 3.2.4 *Detailed Explanation of BUSINESS FUNCTION A4* 37

 3.2.5 *Detailed Explanation of BUSINESS FUNCTION a5*..... 39

3.3 IDEF DIAGRAMS FOR CATALOGUE ORDER FRAMEWORK 40

 3.3.1 *IDEF0 (Activity) Diagrams* 40

3.4 IDEF DIAGRAMS FOR CATALOGUE ORDER SCENARIO 42

 3.4.1 *IDEF0 (Activity) Diagrams* 42

 3.4.2 *IDEF3 (Process) Diagrams* 43

 3.4.2.1 *IDEF3, Process Flow Description (PFD) Diagrams*43

 3.4.2.2 *IDEF3, Object State Transition Network (OSTN) Diagrams*.....43

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

3.4.3 IDEFIX (Data) Diagram (Page 12)..... 44

4 “THE WAY FORWARD”..... 45

4.1 INTRODUCTION..... 45

4.2 MOVING FROM THE DATA MODEL TO TRADITIONAL EDI..... 48

4.3 MOVING FROM THE DATA MODEL TO OBJECT ORIENTED EDI 52

4.3.1 Class Diagram 52

4.3.2 Class Library 52

4.3.2.1 Primitive and Foundation Classes and Hierarchy.....53

4.3.2.2 Core Business Classes53

4.3.2.3 Common Business Classes.....53

5 GLOSSARY..... 54

6 ACRONYMS 58

APPENDIX A – CATALOG ORDER FRAMEWORK..... 59

APPENDIX B – CATALOG ORDER SCENARIO EXAMPLE 69

IDEF0 ACTIVITY MODEL (DIAGRAM PAGES 1-4)..... 70

IDEF3 PROCESS FLOW MODEL (DIAGRAM PAGES 5-7)..... 75

IDEF3 OBJECT STATE TRANSITION NETWORK (DIAGRAM PAGES 8-11)..... 79

IDEFIX DATA MODEL (DIAGRAM PAGE 12)..... 84

OBJECT CLASS MODEL (DIAGRAM PAGE 13) 86

APPENDIX C – EDI CLASS LIBRARIES 88

APPENDIX D – AC.1’S MEMBERSHIP LIST 91

MANAGEMENT SUMMARY

The number of countries and regions involved in UN/EDIFACT has risen considerably over the years. However, it is a fact that EDI has not fulfilled the expected potential. The Small and Medium sized Enterprise (SME) community still finds EDI to be too costly and complex to implement with no off-the-shelf integrated software solution available.

Though no simple 'plug-and-play' solution yet exists for EDI, several technical and economic trends indicate that the time is right: quantum advances in technology; the growing interest in affordable and reusable information systems; and, the increase in the number of SMEs involved in international trade. All of these factors should be considered for the next generation of EDI.

AC.1 believes that a modeled approach to implementing Open-edi provides the most promising opportunity for organizations to establish short-term relationships quickly and cost-effectively. The Open-edi reference model identifies two views to describe business transactions: the Business Operational View (BOV) and the Functional Service View (FSV). The former addresses the semantics of business data and the rules for business transactions. The latter focuses on the information technology requirements to support the BOV. The application of modeling techniques provides an unambiguous identification of business requirements. In turn, the use of object oriented techniques permits the information technology requirements to be differentiated from the semantics of business data, thus providing the ability to interface different FSV implementations (produced by software vendors) to support the BOV. Accordingly, the work of AC.1 is primarily focused on the BOV.

If Open-edi is to overcome the problems currently associated with implementing EDI, a paradigm shift is required. The focus for the development of EDI standards should be shifted from the interchange file to the information contained within the business process.

In order to examine business processes and decompose them into smaller, standard reusable components, modeling techniques must be employed. Through the production of well-defined models it is possible to reduce the number of ways business transactions are interpreted. This will separate the analysis phase from the application design and programming phase and may lead to the production of commercially available 'off-the-shelf' Object Oriented EDI software.

The Integrated Definition Language (IDEF) modeling techniques have been chosen by UN/EDIFACT to develop the next generation of EDI standards.

Using these modeling techniques it will be possible either to map to the way we design messages currently, or to move to a new way to design EDI messages with Object Oriented Technology.

The problem with mapping from IDEF models to the *current* way of designing messages is that the directories have grown in an ad hoc fashion, often without the benefit of modeling. This means that there is frequently no exact match from the structures that the model would suggest to the existing segments in the EDI directories. In trying to re-use current structures, often there are several combinations that could suffice, none being perfect, leading to diverse solutions and inevitably, diverse implementations. There is no automatic method of getting from the model to current EDI directories so there is a heavy reliance on EDI expertise to remember how and why a segment was developed in a certain way. The trouble is that if you ask a different expert you get a different answer.

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

From the IDEF models, AC.1 proposes using Object Oriented Technology (OOT) for the development of standard reusable parts - object classes -where implementation aspects are ‘hidden’ from the user. An Object Class is a template for multiple object instances with similar features. For example, a class could be a ‘person’ and the instance would be ‘John Smith’.

Current EDI practice is based upon incorporating business information requirements within implementation conventions. If OOT is adopted for the next generation of EDI, the business requirements will be met within the Object Classes. This is a significant shift in thinking. It should be noted that the use of the object-oriented approach is not likely to simplify the EDI standards development process - much work is required to model business activities in order to identify the classes. Through the production of a standardized framework, which is the set of all possible functions that meet a common business goal, it will be possible to select a subset of the functions (a scenario) for a given situation. The scenarios should be registered to indicate which functions from the framework have been implemented by trading partners.

If this OOT approach is adopted, it will allow for commercial EDI serving software to be built for various syntax representations and transport mechanisms, including UN/EDIFACT. What is expected, however, is that much of the experience gained from the development of current EDI standards can be transferred to the OOT environment.

If business needs are to be properly considered, there may need to be a change to how EDI solutions are currently designed. In order to separate the business analysis phase from the technical implementation phase it is necessary to replace the current message development groups with business modeling groups, responsible for developing business requirements. A model review group would be necessary to provide constructive feedback to the business modeling groups. Finally, a single production group would be necessary to produce the OOT solutions for EDI.

In order to manage the UN/EDIFACT standards development process for the future, the EDIFACT Steering Group (ESG) has identified two strategic objectives.

Strategic Objective 1, Mainstream UN/EDIFACT, requires little introduction. The current installed user base is using standards that have been developed over a ten-year period, with investment of significant levels of resource. Mainstream UN/EDIFACT is now a mature and stable standard, which will continue to evolve for as long as its users wish to do so. In line with this user support, CEFACt needs to devote resources to the effective maintenance and publication of this standard. Recently, considerable effort has also been put into meeting the requirements of SMEs and this will be continued. Initiatives such as EDI-LITE and a UN/EDIFACT approach to aligned forms have obvious potential for Web-based applications, bringing mainstream EDI to a wider audience.

Strategic Objective 2 is the Object Oriented approach to EDI that is covered in depth in the body of the Reference Guide.

In proposing the two strategic objectives, it is important to appreciate that no one objective is more relevant than the others to CEFACt’s long term goals. Under the CEFACt structure, users will be the main contributors to the work. Therefore, users will be able to choose which objective to pursue, when, if ever, there is a requirement to move to a different objective.

0 FOREWORD

This Reference Guide for the Next Generation of UN/EDIFACT, an Open-edi Approach using IDEF Models and Object Oriented Technology (OOT), is a working document developed by CEFACT's¹ Research, Strategic Advice and Implementation Planning Group, known as AC.1. The intent of this document is to illustrate how Integrated Definition Language (IDEF) modeling techniques can identify the data requirements and data flows of a business process and be implemented using OOT. A well-understood business function, catalog ordering, was chosen to demonstrate the modeling technique.

As a result of input from the UN/EDIFACT Joint Rapporteurs Team, the X12 Strategic Implementation Task Group (SITG) and ISO/IEC JTC 1/SC 30/WG 1, this document has evolved to become a stand-alone guide for the next generation of EDI, encompassing the results of this international work effort. It now represents the recently proposed strategic objective for CEFACT, referred to as Object Oriented UN/EDIFACT. As the research progresses this document will serve as a living reference guide. As such, sections will be added to include proposals for new UN/EDIFACT standards, a proposal for a new standards development process (if necessary), and an assessment of how the problems associated with traditional EDI may be resolved by this new approach. It will be used to evaluate all inputs on the future of UN/EDIFACT, which if accepted will be incorporated in this reference guide. Additionally, it has become a primary reference for SC 30/WG 1, and, as such, was instrumental in the development of the Open-edi scenario template, which is based on the details provided in the catalog ordering example.

AC.1 assumes that this approach to Open-edi is compatible with the ISO/IEC 14662 Open-edi Reference Model patterned after Zachman's Enterprise Architecture². The Open-edi Reference Model provides a baseline for all levels of standards that are needed for the specification of Open-edi scenarios and their implementation. AC.1 further believes that this path to implementation of Open-edi is consistent with the Memorandum of Understanding between the UN/ECE and ISO/IEC in conformance to the Open-edi Reference Model. Within the Open-edi framework semantic models and object class models correspond respectively to enterprise and system perspectives of DATA. Activity and process models correspond respectively to enterprise and system perspectives of FUNCTION within the Open-edi framework. Thus, as indicated in the title of this document, Object Oriented UN/EDIFACT is proposed to be an implementation of Open-edi.

This document specifically addresses the Business Operational View (BOV) of Open-edi, not the Functional Services View (FSV). Historically, traditional EDI standards have addressed the information structure and format, not the transport layer. AC.1 will continue to concentrate on the BOV, similar to the activity of ISO/IEC JTC 1/SC 30 to date. The prospect of FSV-related standards to be forthcoming within SC 30 appears to be unlikely at this time as it deals with message transport and not data structures. It is assumed, instead, that FSV de facto standards that enable distributed object computing environments will be developed by commercial software vendors. This is not to say the FSV will be ignored in Open-edi. Prototype implementation must account for all issues, and will require integration of all aspects of the business process of information exchange.

¹ Center for the Facilitation of Procedures and Practices for Administration, Commerce and Transport (CEFACT) - the reengineered United Nations organization for the administration of UN/EDIFACT Standards Development.

² J. A. Zachman, "A Framework for Information Systems Architecture," *IBM Systems Journal* 26, No.3, (1987)

As implementation progresses, alliances for non-BOV-related standards, e.g., FSV and legal, must be made to ensure the pieces will fit together.

AC.1 has taken the position that modeling of the business process is fundamental to EDI standards, whether it be for current EDI or for next generation EDI. The approach is to model the entire business function, i.e., the activity being performed, and not to focus on the information structure that might be contained in an interchange message. The IDEF modeling techniques were chosen because they integrate the business activity with the business process along with the semantics. Further, this decision was based on 15 years of extensive international application experience, i.e., CALS, and on the likelihood that IDEF will be fast tracked to become an ISO standard through the auspices of the Institute of Electronics and Electrical Engineers (IEEE).

Object Oriented Technology was chosen by AC.1 as an avenue of research for the system perspective, i.e., data implementation within the Open-edi framework, primarily because, in the words of the Chairman of the ESG, as "an area of intense activity in major organizations, it has already demonstrated its ability to deliver significant advances in business performance." OOT is clearly where the IT industry is placing its bets on accomplishing application interoperability. As major software suppliers develop an OOT infrastructure for interoperability, data constructs will naturally follow in the form of various levels of object classes, defined according to the proprietary interests of each supplier, and without input of true business requirements. AC.1 contends that true interoperability can only be achieved through standardized business object classes that are accessible in the public domain through standard class libraries. Only within the consultative process of EDI standards bodies, where business users participate, can the business requirements be adequately gathered and correctly modeled for quality implementation. Work must begin now to provide standardized business classes before EDI software providers have no recourse but to proceed with proprietary "business" class implementations.

AC.1 proposes standardizing Open-edi frameworks. A framework will consist of object classes and IDEF models. Scenarios will be built from the framework contents, and publicly registered. Companies wishing to exchange data with a user of a registered scenario can then establish data interchange without prior negotiations or agreements. The framework contents will provide reusable building blocks for development of scenarios and other frameworks.

The business analysis required to interchange information will not be sidestepped, or even simplified, by this new approach to standards. Instead of the current message design methodology that incorporates the business information within implementation conventions after the EDI messages have been standardized, this Open-edi approach requires rigorous analysis of the business process and incorporation of the information transfer details up-front. Principles such as *information is exchanged the moment it is created as part of the overall business function life cycle without duplication of information previously exchanged* are fundamental to the analysis. The payoff will be unambiguous Open-edi scenarios that can either be reused or quickly assembled from standard building blocks in the construction of commercially available and affordable EDI servers.

As a final note, it must be emphasized that business modeling is fundamental to EDI standards development, regardless of the implementation technology. While OO-edi prescribes business modeling as the first step, OOT is only one of many system perspectives of the enterprise model perspective. Indeed, traditional EDI message development will also benefit from business process modeling. As the parallel strategic objectives for existing and new generation UN/EDIFACT progress, they will be grounded in common business process models.

1 INTRODUCTION

The economic advantages of Electronic Commerce (EC), including Electronic Data Interchange (EDI), are widely recognized. Clearly, it is expanding into many new areas of business and administration. To date, however, only large multi-national companies in the manufacturing and service sectors have taken advantage of this functionality. A variety of reasons, including cost, complexity, legal, administrative, and security issues, have slowed EC/EDI implementation within the Small and Medium sized Enterprise (SME) community.

Concurrently, a series of technical and economic trends are driving changes to traditional business practices in that:

- Quantum advances in technology are providing substantial opportunities to reshape business landscape through enhanced productivity, competition and product definition.
- The demand is fierce for affordable and reusable information systems that can keep pace with evolving technology and support the rapidly changing global business environment.
- Global economic and trade considerations to maintain and expand market share are dictating an increased requirement for SMEs to expand trade within and across borders.

Numerous ventures are well underway to meet this extraordinarily powerful demand that now haunts the marketplace. EC and EDI have a definite role in this process. However, if UN/EDIFACT is to meet the requirements for use by the potential range of users and move forward in a leadership role into the next millennium, it must move rapidly to identify and develop the next generation of message standards.

1.1 GENERAL

In direct response to requests to address directory quality and production, Group of Experts (GE.1) created the Research, Strategic Advice and Implementation Planning Group (AC.1) to “define the next generation of processes and procedures necessary for the management of the UN/EDIFACT message development and maintenance life-cycle, meeting user needs for quality standards.” This document summarizes the status of AC.1’s research and serves as a reference guide. It should be viewed as a “living document” which will be updated periodically as work progresses.

AC.1’s overall approach has been to use as much “out of the box” thinking as possible in researching the current and projected business environment, the evolution of technology and the ability to leverage the ongoing work targeted at meeting market demand. Without question UN/EDIFACT subscribes to the Open-edi position that “significant opportunities...for EDI arise from using...information and communication technologies as **enablers**, (not drivers) in ‘reengineering’ business processes...and simplifying business rules and relations with...business partners.”³ Today emphasis is placed on automating and implementing “as is” business processes.

³ JTW63/92-058 ISO/IEC JTC1/WG 3 N032 of a992-08-05, *The “open-edi” Conceptual Model: Why, How, What and Where to From Here?*, Personal Contribution from Dr. Jake V. Th. Knoppers, Canada, to the ISO/IEC JTC 1/WG 3 meeting of September 1 - 4, 1992.

AC.1 believes that by transferring the resource expenditure to the beginning of the business process substantially more productivity can be achieved.

Among AC.1's conclusions thus far are that:

- Significant opportunities accrue with the use of modeling techniques to identify data requirements and data flows associated with a particular business process.
- Object-oriented technology (OOT) provides one of the most powerful alternatives to identify and convey common business information.

AC.1's underlying assumption is that models unambiguously represent the activity and information involved in information interactions, and, thus, lend themselves to a clear understanding of what must be provided for in EDI standards design. Likewise, AC.1 believes that common business objects are the essential building blocks from which application developers can create affordable software representing sharable frameworks which can keep pace with rapidly evolving business and technology needs.

Because this document presents a series of concepts and technology innovations with which not everyone will be familiar, a level of expertise should be available in order to benefit fully from the discussion. To this end, AC.1 commend *Object-Oriented Technology: A Manager's Guide*, David A. Taylor, Ph.D. Addison-Wesley Publishing Company, September 1995, ISBN 0-201-56358-4 as one introductory text for laying a basic foundation in this regard. It is expected that this, or any other text, might be used for education and understanding OOT relationships. However, because of the evolving interpretations associated with OOT, AC.1 will rely on the glossary provided herein as the basis for all definitions to be presented.

1.2 DEFINE A STANDARDS DEVELOPMENT PROCESS TO MEET CUSTOMER NEEDS

AC.1 has investigated several methodologies for the management of the procedures and processes necessary for the handling of information requirements related to business transactions. In addition, methods for the organization and management of the information itself have been investigated.

The conclusion reached by AC.1 is that a construct is required to provide both the process and the deliverables concurrently. Since the UN/EDIFACT process needs to incorporate multiple perspectives (for example, the business and technical perspective) and use a variety of tools (such as analysis and modeling), the construct should also take this into consideration. It must allow for the correct tools and methodologies to be used by the appropriate people at the relevant stage, show how each relates to the other, and improve communication within the UN/EDIFACT community. AC.1 believes that the construct that will satisfy the above requirements is the Zachman Enterprise Architecture.

1.2.1 THE ENTERPRISE ARCHITECTURE

The Enterprise Architecture (Figure 1) is based on the Information System Architecture (ISA)⁴ and shall be used to identify and resolve user requirements in the future. Each of the rows in Figure 1 identifies a model, and each of the columns represents a unique perspective.

⁴ A Framework for Information Systems Architecture, IBM Systems Journal Volume 26, No 3, 1987, page 276-292

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

“The logic structure or rules for the framework are generic. They can be used for structuring the description of any complex object. The framework was first discovered by observing how the manufacturing discipline divides the descriptions of complex engineering products for the purposes of design and manufacture. It would appear that the use of the design artifacts are several including:

- Partitioning the design trade-off decisions into manageable, independent variables
- Ascribing appropriate design formalisms for each variable.
- Establishing a base line of descriptive representations for managing changes in the product during and after its production.”⁵

AC.1 is working on the detail of the Enterprise Architecture and how each of the cells in Figure 1 relates to UN/EDIFACT. However, it is evident that different tools can be used at different stages of development to describe the activities. Indeed, much of the lower portion of the Enterprise Architecture can be automated using, for example, CASE (Computer Aided Software Engineering) tools. Also, some of the output of the Business and Information Modeling Group⁶ (BIM) and the International Trade Transaction Group⁷ (ITT) could be used at various levels in the Enterprise Architecture.

| | What? DATA | How? FUNCTION | Where? NETWORK | Who? PEOPLE | When? TIME | Why? MOTIVATION |
|--|--------------------------|--------------------------|---------------------------------|------------------------------|-----------------------|----------------------------|
| Scope (contextual) | List of things important | List of processes | List of locations | List of organizations Agents | List of events | List of goals |
| Enterprise model (conceptual) | Semantic Model | Process flow diagram | Logistics network | Organizations chart | Master schedule | Business plan |
| System model (logical) | Data model | Data flow diagram | Distributed system architecture | interface architecture | Processing structure | Knowledge architecture |
| Technology model (physical) | Data design | Structure chart | System architecture | Technology interface | Control architecture | Knowledge design |
| Components pre-presentations (out-of-context) | Data definition | Program | Network architecture | Security architecture | Timing definition | Knowledge definition |
| Functional system | Data | Function | Network | Organization | Schedule | Strategy |

Figure 1 - The Enterprise Architecture

1.2.2 SCOPE

The scope is used to describe the overall picture of the domain in question. The scope should describe the boundaries of influence, listing the important aspects, processes, locations, organizations events and goals. It corresponds to an executive overview of a company or organization.

⁵ Extending and formalizing the framework for information system architecture, IBM Systems Journal Volume 31, No. 3, 1992, pages 613-614

⁶ Joint Rapporteurs Team - Business and Information Modeling Technical Work Group (T5)

⁷ UN/ECE/TRADE/WP.4/GE.2 - International Trade Transaction Modeling Group (ITT)

The determination of the scope for an organization is a top-down process identifying the range of major business processes, supporting data, etc. to be undertaken by the organization. In the current situation, with the implementation of re-engineering report, it will be necessary that the leadership of CEFACT, via its Steering Group, assume the responsibility for empowering the appropriate Working Groups to take on the three levels of specifications as needed. Below is a proposed structure:

| | <i>Traditional responsibility</i> | <i>Proposed responsibility</i> |
|--|--|---------------------------------------|
| <i>Scope specification</i> | WP.4 | CEFACT P&P WG |
| <i>Enterprise specification</i> | GE.1/ESG | CEFACT Framework WG |
| <i>System specification</i> | Each JRT/JM | CEFACT EDIFACT WG |

Figure 2 – Responsibility for Architecture Specification

1.2.3 MODELING

The Enterprise Architecture has three rows that relate to modeling: the enterprise model, the system model and the technology model. Each of these rows deals with a different aspect of the same overall problem. The enterprise model is used to describe the usability constraints that relate to the conceptual view of the end product. The system model provides a mechanism for describing the design constraints. The construction restraints are given in the technology model. Taking a real life example, when buying a car, the customer may decide that safety and comfort are two important factors. The designer of the car will have to consider how safety and comfort can be integrated and may have some safety laws to comply with (such as metal density and seat-belt requirements). The manufacturer of the car is constrained by resources and profitability. All of these views are different aspects of the same issue.

For the UN/EDIFACT organization, the main issue is the use of EDI to facilitate trade. There are differing views and requirements of this issue that must be modeled. EDI is a business tool that uses technology and cuts across geographical and political boundaries. The business, technological and political views should be carefully documented so that the UN/EDIFACT process can become more logical, hence more predictable.

Much of AC.1’s findings regarding modeling are not new. The BIM group, whose work is concentrated on current message development activities, has developed a framework that could be incorporated into the Enterprise Architecture framework, with minor modifications. At the very least, the commonalities between the BIM findings and the Enterprise Architecture could be the bridge from the old to the new.

1.2.4 OBJECT ORIENTED TECHNOLOGY

Data maintenance flexibility is key to UN/EDIFACT’s future success; this is evident from the criticisms directed towards UN/EDIFACT. The new method must be open to change to allow new requirements and should have the following principles:

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

- the ability to hide different implementations behind a common interface;
- a mechanism whereby one set of data can be defined as a special case of a more generic set;
- a technique of packaging data with its corresponding process/procedures (context);
- and a way to ensure semantic business information is captured.

AC.1 recognizes that the technology exists to provide for these principles. The technology is known as Object Oriented Technology⁸ (OOT) and supports the concepts of:

- Polymorphism, the ability to hide different implementations behind a common interface. This allows many business and administration sectors to take advantage of data objects.
- Inheritance, the mechanism whereby one class of objects can be defined as a special subclass of a more generic class. This allows for efficiency in the standards.
- Encapsulation, the technique of packaging data with its corresponding process/procedures. This means that data is linked to context, hence reducing the need for complex implementation guides.
- Using Activity, Process and Data Modeling captures abstraction, a way to ensure semantic business information.

1.3 CO-ORDINATION AND LIAISON

Numerous corporations and organizations are involved in comparable initiatives targeting the advancement of new methodologies and technologies to enhance data exchange. AC.1 has established a series of informal liaisons with organizations working in this area. In particular, AC.1 is working with ISO/IEC JTC 1/SC 30/WG 1 Open-edi, utilizing the liaison structure of GE.1, to expand basic Open-edi parameters and identify a series of object oriented concepts. AC.1 envisions that these concepts could result in the creation of standard building blocks that are readily usable in the development of EDI compatible software supporting interoperable processes.

In the way of clarification AC.1 notes that the scope of this collaboration is limited to the area of business processes, including only those activities that involve information exchange with external trading partners. Cooperation with providers of FSV functionality is required to ensure interoperable processes. Prototype activity will require the cooperation of software developers.

1.4 NORMATIVE REFERENCES

The following standards contain provisions, which through reference in this text constitute provisions of this document. All standards are subject to revision, and parties to agreements based on this document are encouraged to investigate the possibility of applying the most recent editions of

⁸Object Oriented Technology: A Manager's Guide, David A. Taylor, ISBN 0-201-56358-4

the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/IEC 14662 Information technology - Open-edi reference model Document

2 BACKGROUND

This paper introduces what to many will be a new approach to considering the standards environment, and it will include a number of new terms and concepts. AC.1's objective is to present the business and standards implications and not delve deeply into technology for its own sake. Accordingly, an overview of the key points is provided below as a means of better understanding the results of AC.1's research and recommendations.

2.1 OPEN-EDI

Open-edi is an ISO/IEC vision of future EDI. ISO/IEC 14662 provides a baseline for all levels of standards that are needed for the specification of Open-edi scenarios and their implementation. AC.1 has explored and recommended various modeling techniques including IDEF for business modeling and OOT for information modeling.

Open-edi takes a generic approach. It enables organizations to establish short-term relationships quickly and cost effectively. Open-edi provides the opportunity to lower significantly the barriers to electronic data exchange by introducing standard business scenarios and the necessary services to support them. In principle, once a business scenario is agreed upon, and implementations conform to the Open-edi standards, there is no need for prior agreement among trading partners, other than the decision to engage in the Open-edi transaction in compliance with the business scenario.

The field of application of Open-edi is the electronic processing of business transactions among autonomous multiple organizations within and across sectors (e.g., public, private, industrial, geographic). It includes business transactions that involve multiple data types such as numbers, characters, images and sound. The Open-edi Reference Model provides the standards required for the inter-working of organizations, through interconnected information technology systems, and is independent of specific information technology (IT) implementations, business content or conventions, business activities and organizations.

The Open-edi Reference Model places existing EDI standards in perspective using two views to describe the relevant aspects of business transactions: the Business Operational View (BOV) and the Functional Service View (FSV).

The BOV addresses the aspects of a) the semantics of business data in business transactions and associated data interchanges, and b) the rules for business transactions which apply to the business needs of Open-edi, including:

- operational conventions,
- agreements,
- mutual obligations.

The FSV addresses the supporting services meeting the mechanistic needs of Open-edi. It focuses on the Information Technology aspects of functional capabilities, service interfaces, and protocols, including

- capability of initiating, operating and tracking the progress of Open-edi transactions,
- user application interface,
- transfer infrastructure interface,
- security mechanism handling,
- protocols for inter working of information technology systems of different organizations,
- translation mechanisms.

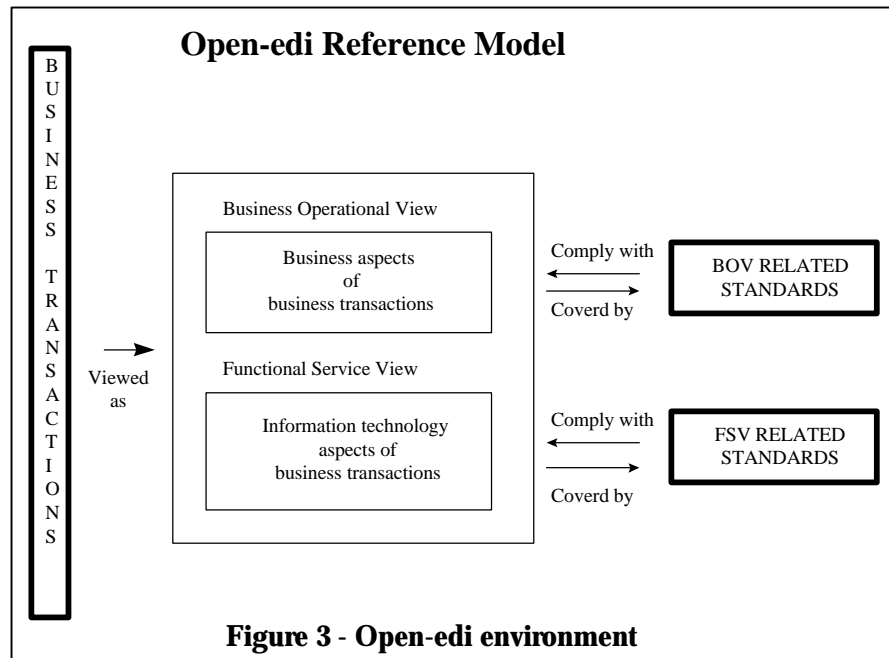


Figure 3 - Open-edi environment

Figure 3 sets out the relationship between the Open-edi reference model and these views. While the Model itself is developed as a standard, other standards are required. Standards are driven by, and must satisfy, the requirements identified within the Open-edi Reference Model.

Although AC.1’s work must satisfy the overall requirements of the Model, the primary focus shall reside with the BOV and shall be independent of the supporting FSV solution. AC.1’s assumption is that the FSV will be developed by commercial software vendors which enable distributed object computing environments, and ensure backward compatibility to traditional EDI messages. As such, the resultant BOV related standards will provide the business and object class models to construct Open-edi scenarios.

2.1.1 SHIFT THE FOCUS

Interoperation among application programs requires that there be “common ground” in their exchange of information, so that there can be common understanding and agreement on the information being jointly processed. Common ground in this exchange of information is accomplished in current EDI methodology through a neutral, application independent syntax, i.e., typically for business data a translated UN/EDIFACT interchange file. All consideration of application programs, how to facilitate their interoperation, functionality variations, and the business practices behind them are deliberately ignored. Instead, the current EDI standardization process in UN/EDIFACT concentrates solely on the structure and content of the translated interchange file. The problems associated with UN/EDIFACT standards, and the standard development process, are well documented and are not repeated here.

However, it is essential to understand that for Open-edi to overcome the current impediments to implementing EDI, a new paradigm must be envisioned that shifts the focus on EDI standards from the interchange file to the information contained in the business processes. While business practices from one business organization to another are highly variable, depending on competitive strategies, experience and management style, activities can be decomposed into business processes that are more generic to the type of business. This analysis through the modeling process will identify activities and object classes that are likely candidates for standardization. AC.1 looks for standard reusable components from which to construct EDI compatible software. Such a goal is a core concept of object technology.

2.2 MODELING THE BUSINESS PROCESS

2.2.1 INTRODUCTION

As AC.1 considers the necessity to decompose business processes to their more generic components, UN/EDIFACT must utilize a consistent methodology (modeling techniques) for conducting the analysis. Thus, it is important to explore the benefits of using modeling techniques to identify the data requirements and data flows of a particular business process. These models assist in providing an interface specification that enables non-standard data, internal to a business process, to be mapped and translated to a representation of standardized data.

It is proposed that models, which provide the interface specification, will constitute the new EDI standards, once they are certified as satisfying the business requirements. These new EDI standards will be independent of the interchange data syntax, transport infrastructure, and EDI server software.

AC.1’s primary objective in this new focus on EDI standards is to make EDI technology widely available, non-obtrusive to the business process, and cost effective for all organizations, including SMEs. The requirements to make this objective a reality include:

- Production of well-defined, consistent standards for interoperability, i.e., reduces the number of ways of doing things.

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

- Development of off-the-shelf tools that are commercially available for analysis and implementation.
- Separation of analysis from application design and programming.
- Availability of training and reference sources (i.e., take advantage of a mainstream methodology for new projects in industry).
- Provision for automatic generation of EDI interactions.
- Separation of data definition and format from the transport layer.

AC.1 selected Integration Definition Language (IDEF) as an integrated suite of modeling techniques to characterize these new EDI standards. Three IDEF modeling tools are used in the modeling process, and each produces distinct models as follows:

- Activity model (IDEF0)
- Process model (IDEF3)
- Data model (IDEF1X)

2.2.2 ACTIVITY MODELING

An activity model shows the decisions, actions and activities of an organization (or organizations). It is a system for expressing what is done, not how it is done. The activity model provides a complete view of a business process by showing an activity in relation to its inputs, outputs, controls and mechanisms. Activities shall be decomposed into component activities up to the point that they are basic enough to be reusable and are thus candidates for standardization. The same modeling technique can be used to model current and future business processes (in different models) as well as alternative processes for achieving business objectives.

IDEF0 is a method for the rigorous definition of the business activities. It does not describe an organizational structure, a business program, a product or set of products. An activity must always have at least one output and at least one control. There can be zero or many inputs and zero or many mechanisms. An input is transformed by an activity into an output under the controls (i.e., conditions required) specified and using the mechanisms (i.e., the means to perform the function) stated.

Activities are represented by boxes; all inputs enter the left of the box; all controls enter into the top of the box, all outputs exit from the right side of the box and all mechanisms enter into the bottom of the box. See Figure 4.

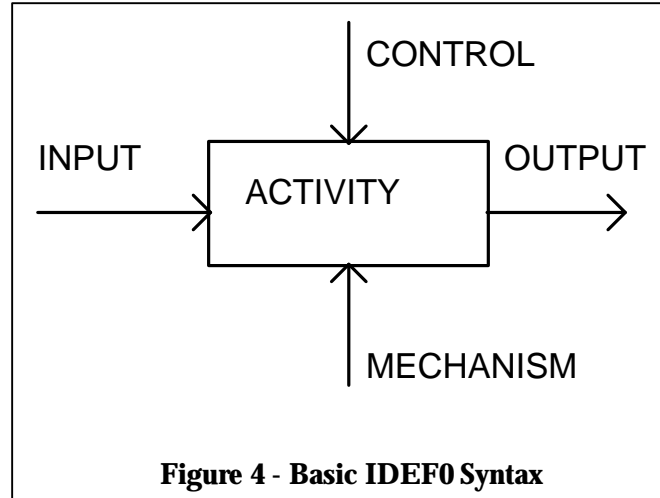


Figure 4 - Basic IDEF0 Syntax

Applying the IDEF0 method results in an organized representation of the activities and important relations between them. IDEF0 is designed to “tell the story” of what an organization does (or organizations do); it does not support a process. A box in an IDEF0 model represents the boundaries drawn around some activity. Inside that box is the breakdown of that activity into smaller activities, which together comprise the box at the higher level. This hierarchical structure helps the practitioner keep the scope of the model within the boundaries represented by the decomposition of the activity.

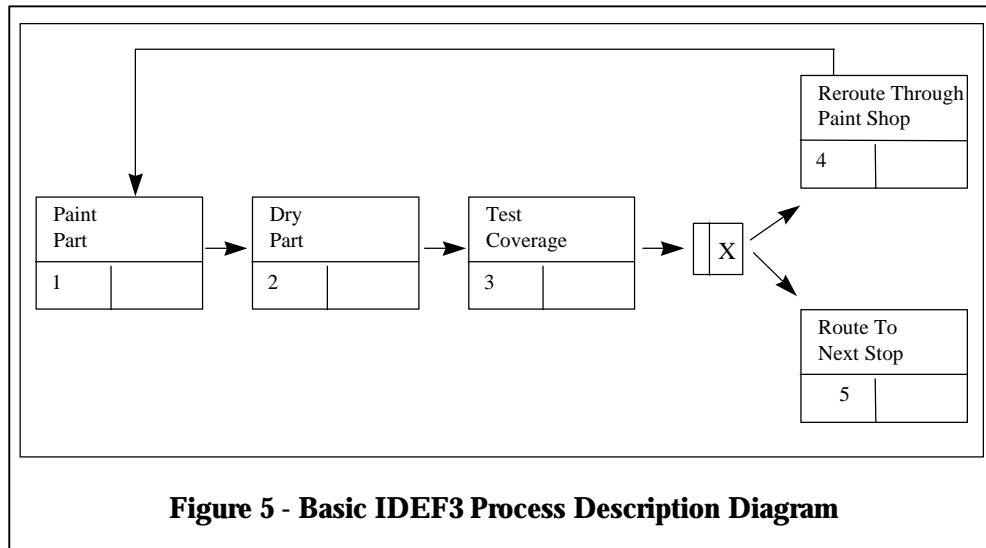
IDEF0 is very effective in identifying the core activities and secondary functions of the organization. The actual act of identifying what the organization does will often result in answering the more important question of why this is so. This represents the first step of many reengineering efforts focused on identifying candidate organizational areas for reengineering.

2.2.3 PROCESS MODELING

The IDEF3 Process model provides a mechanism for collecting and documenting processes. It captures precedence and causality relations between situations and events in a structured format for expressing knowledge about how a system, process or organization works. IDEF3 captures the behavioural aspects of an existing or proposed system. This captured process knowledge is structured within the context of a framework. The resulting IDEF3 descriptions provide a structured knowledge base for constructing analytical and design models.

Process modeling (IDEF3) has two components: process flow descriptions and object state transition network (OSTN) descriptions. A process flow description captures the knowledge of how things work in an organization. It identifies the time sequence and decision points relating to each activity within the context of the entire business process. In short, a process flow description model takes the activities identified in an activity model (IDEF0) and adds the sequence in which the activities are performed and shows which activities may be optional. For example, an activity called “bake” may have as inputs flour, water, eggs, and sugar, and bread and cake as outputs. A process decision must be made as to how to utilize the inputs to create the desired output, i.e., bread or cake.

The basic unit of the process flow description is the Unit of Behavior (UOB), represented by a box. A UOB can be further classified as a function, activity, action, act, process, event, decision or a procedure. Arrows are used to indicate sequence. A junction box, denoted by an “X” or “&” inside a smaller box on the diagram, indicates requirements and is used as a mechanism for introducing logic to the flows being represented. An X in a junction box denotes a set of activities that are mutually exclusive. An “&” in a junction box denotes a set of activities all of which are required. See Figure 5.



Object states and state transition arcs are the key elements of an OSTN diagram. An OSTN describes the world from an object’s viewpoint recording all of the changes that the object undergoes and the processes that cause the changes to occur. Object states are defined in terms of the facts and constraints that need to be true for the continued existence of the object in that state and are characterized by entry and exit conditions. They are represented in the diagram by circles. The entry conditions specify the requirements that need to be met before an object can transition into a state. The exit conditions characterize the conditions under which an object can transition out of a state. The constraints are specified by a simple list of property/value pairs or by a constraint statement. The values of the attributes must match the specified values for the requirements to be met.

The transition from one state to another is represented by the lines (known as state transition arcs) that connect the circles. The process that assists in the state transition is represented by specifying the relevant UOB to the transition arc between the two object states.

The OSTN description summarizes the allowable transitions an object may undergo throughout a particular process: it describes the world from the perspective of an object. See Figure 6.

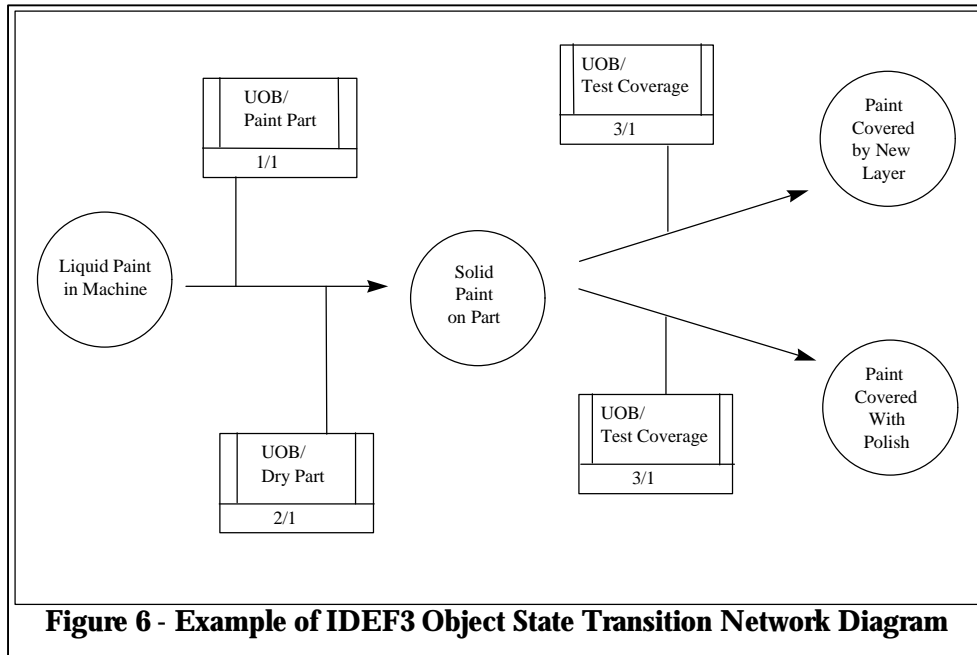


Figure 6 - Example of IDEF3 Object State Transition Network Diagram

2.2.4 DATA MODELING

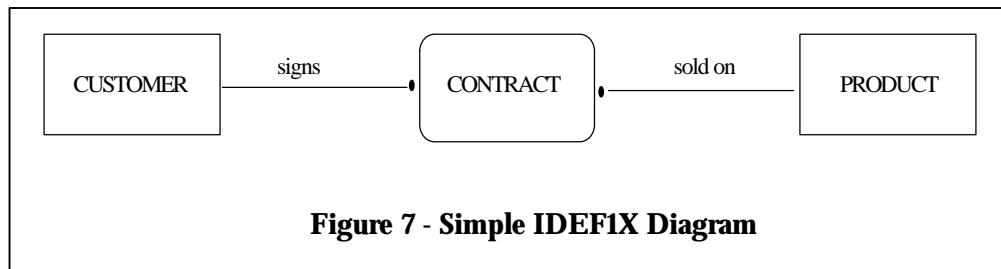
Data modeling is a technique for describing information structures. Logical models specify business information requirements and rules. Physical models specify solutions. As such, data modeling can be viewed as a way to develop information system requirements and solutions. Models are used to design a physical system database that represents a balance between the specific needs of a particular implementation project and the general needs of the business area in which that project is being conducted. Most data models are representations of things (entities), properties of things (attributes) and associations among things (relationships).

A data model is used to understand the inherent nature of the data identified in the activity model, and, in fact, details the data requirements identified in the activity and process models. The intent of data modeling is to model business rules and the relationship among the rules, e.g., a company has many departments, a department has many employees, an employee can work for one department only

IDEF1X is used to understand and diagram the nature of the data identified in the inputs, controls, outputs, and mechanisms of the IDEF0 model. The IDEF1X model is constructed at the conceptual level and is referred to as “the model of the business”.

Entities in data models are abstractions of real-world things. For example, a CUSTOMER entity is an abstraction of the real-world concept of a customer. Individual pieces of data that describe an entity are called attributes. Relationships denote how entities relate to one another. Figure 7 depicts the relationship between two entities the interaction of which results in the generation of a new entity.

IDEF1X is designed to depict a model of an information system using relational technology. However, much of the work of AC.1 is based upon object technology, thus limiting the usefulness of IDEF1X. Because of these limitations and because within the EDI environment the requirement is to be independent of the application, we have chosen to disable many IDEF1X features so that the model could be implemented in any technology including relational and object-oriented. Specifically, these suppressed features are the Identification of key Identifiers and non-Identifier versus Identifier relationships. In addition, an IDEF1X model does not permit many-to-many relationships. Since they are unresolvable they cannot be processed.



2.3 OBJECT ORIENTED TECHNOLOGY

2.3.1 INTRODUCTION

Object Oriented Technology (OOT)⁹ is one of the most talked about concepts of recent years. It all comes down to organizing things in ways that echo how things are put together and relate in the real world.

This section¹⁰ serves as an overview of how OOT concepts relate to EDI and will address:

- What classes and objects are, and how they relate to each other;
- The two main parts of a class or object, its behaviors and its attributes;
- Class inheritance and how inheritance affects the way one designs Information Parcels;
- Some information about packages and interfaces;

⁹ Numerous sources have alternative naming requirements for object technology. This document uses the term Object Oriented Technology (OOT) rather than terms such as Object Oriented Analysis (OOA) and Object Oriented Design (OOD).

¹⁰ The general outline of this section is based on text published by Laura Lemay .It has been changed to serve as introduction to users, implementers and designers of EDI, as well as to permit specific EDI examples.

- The authorized access to operations and attributes within a class, making them available those operations through a public interface.

2.3.2 THINKING OF OBJECTS IN AN ANALOGY

Consider the children’s toy, Lego®, small plastic building blocks in various colors and sizes. They have small round pegs on one side that fit into small round holes on other Lego pieces so that they fit together snugly to create larger shapes. With different Lego pieces (Lego wheels, Lego engines, Lego hinges, and Lego pulleys), it is possible to build castles, trailer tractors, giant robots that swallow cities, or just about anything else you can imagine. Each Lego piece is a small object that fits together with other small objects in predefined ways to create other larger objects. In turn, these fit together and form even larger objects. For example, a tractor and trailer will fit together to form a wagon.

As a second example suppose it is possible to walk into a computer store and, with a little background and often some help, assemble an entire personal computer system from various components: a motherboard, a CPU chip, a video card, a hard disk, a keyboard, and so on. Ideally, when all the various self-contained units have been assembled, the result is a system where all its units work together to create a larger system that can solve the computational problems for which it was designed.

Internally, each of those components may be vastly complicated and engineered by different companies with different methods of design. But it is not important to know how the component works, what every chip on the board does, or how, when you press the A key, an “A” gets sent to the computer. As the assembler of the overall system, each component you use is a self-contained unit. The main interest is how the units interact with each other. Will this video card fit into the slots on the motherboard, will this monitor work with this video card, will each particular component convey the right commands to the other components it interacts with so that each part of the computer is understood by every other part, etc.? Once it is known what the interactions are between the components and can match the interactions, putting together the overall system is easy.

OOT works in exactly this same way. Using OOT, the overall design (model) is made up of lots of different self-contained components (objects), each of which has a specific role in the model and all of which can talk to each other in predefined ways.

2.3.3 OBJECTS AND CLASSES

OOT is modeled on how, in the real world, occurrences of objects (called object instances) are often made up of many kinds of smaller objects. This capability of combining objects, however, is only one very general aspect of OOT. Several other concepts and features make creating and using objects easier and more flexible. The most important of these features is that of classes.

A **class** is a template for multiple objects with similar features. Classes embody all the features of a particular set of object instances. When a model is designed in an

Object Oriented environment, actual objects are not defined, but rather classes of objects are defined. An **instance** of a class is a synonym for an actual object. If class is the general (generic) representation of an object, an instance is its concrete representation. For example, a class could be “person” and the instance would be “John Smith.”

As an EDI design example, a class may be created for a physical location called “address”. The Address class defines the features of an address (its label, its components, its format) and how it behaves (what can be done at that address {receive, send, etc.}, does it need a postal code or a city name, does it change formats because of a special command request, etc.?). Once the Address class is defined, one can then easily create instances of that address, that is, address objects, that all take on the basic features of the address as defined by the class, but which may have different appearances and behavior based on the requirement for that particular address. By creating an Address class, components for each individual address do not require redefinition based upon usage in a model. The Address class can be reused to create different kinds of addresses as needed in any model.

2.3.4 ATTRIBUTES AND BEHAVIOR

Every class is generally made up of two components: attributes and behavior. The following example demonstrates how each applies to a theoretical class called “Email Address”.

Attributes are the individual things that differentiate one instance and class from another and determine the appearance, state, or other qualities of that object. The attributes of an Email Address might include the following:

- User Identification: John, KDN01, 1001-0998
- Domain Name: company.com, [COUNTRY=US, ADMIN_DOMAIN=XYZ]
- Type: Internet, X.400, etc.

Attributes of a class can also include information about its state, e.g., status condition (In Service), last time used, etc.

Attributes are defined by variables, instance and class variables. Each instance of a class can have different values for its variables, called an instance variable. Instance variables instantiate the attributes of an object. The class defines the type of attribute, and each instance stores its own value for that attribute. Each attribute, as the term is used here, has a single corresponding instance variable. Changing the value of a variable changes the specific attribute of that object. Instance variables may be set when an object is created and stay constant throughout the life of the object. Changes to instance variables may occur through external control or due to other internal changes in state.

Class variables apply to the class itself and to all its instances. Unlike instance variables, whose values are associated in the instance, class variables’ values are associated in the class itself.

Behavior is the only way objects can do anything to themselves or have anything done to them. A class’s behavior determines what instances of that class do to change their internal state or when that instance is asked to do something by another class or object. For example, the theoretical Email Address class could have the following behavior:

- Receive information;
- Send information;
- Acknowledge receive;
- Forward information;
- Identify owner.

Operations or methods are functions defined inside classes that operate on instances of those classes. Methods define behavior within a class. Methods always affect only a single external object, although objects communicate with each other using methods that are defined in the object’s public interface. Methods exist as instance and class methods. Instance methods apply and operate on an instance of a class; class methods apply and operate on the class itself.

2.3.5 OBJECT RELATIONSHIPS

There are many ways in which object classes interrelate and this section discusses some of more commonly used relationships. A relationship is basically a way to represent a logical link between objects. Each of these relationships will likely be used in the development of object class diagrams that represent the scenarios in the Open-edi BOV. The three main relationships discussed are association, aggregations, and inheritance.

2.3.5.1 Associations

An association is a semantic relationship between two or more classes. Typically the common label for the relationship are verbs in the problem statement such as “uses a”, “supplies”, “owns”. The example (Figure 8) below shows an association between product and supplier where a supplier “supplies” many products and products are “supplied by” many suppliers.

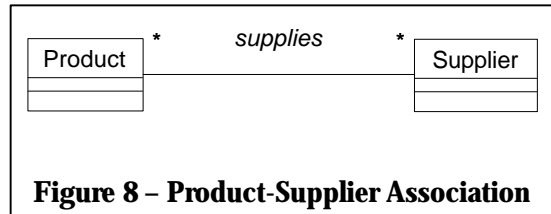
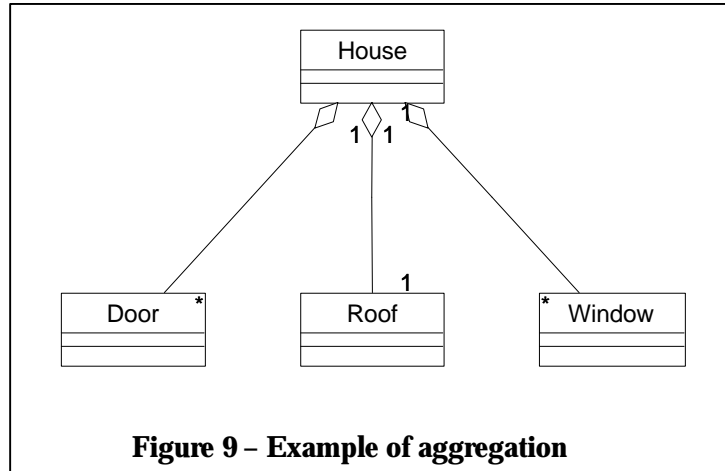


Figure 8 – Product-Supplier Association

2.3.5.2 Aggregations

An aggregation is a special form of association between a class that describes the aggregate whole and one or more classes that describes the composition of the parts that make up the whole. The common labels for the relationship is “part of”, “contains”, “consists of”. Instantiated objects that are composed of many parts or components are often complex objects. The example (Figure 9) below shows a complex object called House that contains many components such as doors, windows and one roof. The House object contains references for the creation of lists of doors and windows.



2.3.5.3 Inheritance

Inheritance is one of the most crucial concepts in OOT in that it has a very direct effect on how EDI classes are designed. Inheritance is a powerful mechanism that permits the development of a derived class by only specifying how that class is different from its parent class. Inheritance permits automatic access to the information contained in the parent class, but not to its sibling classes.

With inheritance, all classes, whether developed locally, used from other class libraries and those from the standard business classes as well, are arranged in a strict hierarchy (see Figure 10).

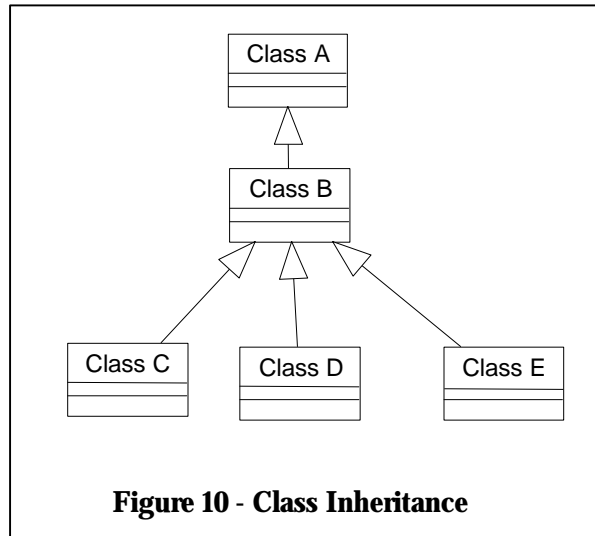


Figure 10 - Class Inheritance

Each class has a **superclass** (the class above it in the hierarchy), and each class can have one or more subclasses (classes below that class in the hierarchy). Classes further down in the hierarchy are said to inherit from classes further up in the hierarchy.

Derived classes or subclasses inherit the public and protected methods and variables from their parent classes or superclasses, that is, in any particular class, if the superclass defines needed behavior, then that behavior need not be redefined in some other subclass. Any given class automatically possesses behavior from super classes above it in the hierarchy. Each class farther down in the hierarchy adds more information and becomes more tailored to a specific purpose. In this way, a class hierarchy can be thought of as defining very abstract concepts at the top of the hierarchy with those ideas becoming more concrete the farther down the chain of superclasses you progress.

In designing new EDI classes it is desirable to create a class that has all the information some other class has, plus some extra information. For example, create a version of Address, but with its own built-in identification. To get all the Address information, define the new class to inherit from Address, and then specify the incremental differences specific to the requirements of this new address.

Subclassing is the mechanism for defining new classes as the differences between them and their superclasses. It involves creating a new class that inherits from some other class in the class hierarchy. Using subclassing, one only needs to define the differences between a class and its parent; the additional behavior is all available through inheritance.

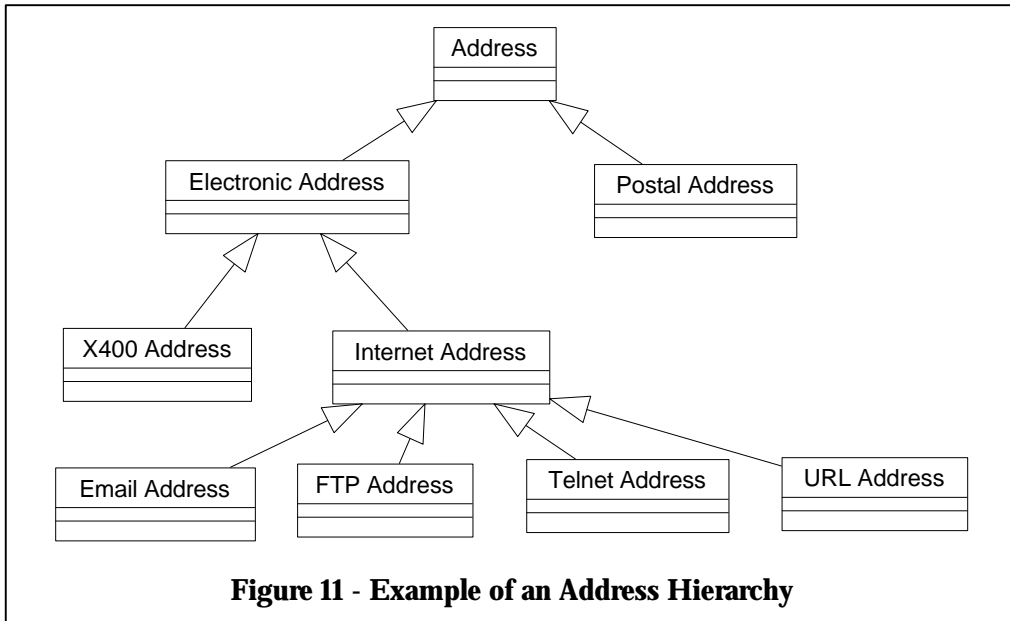
2.3.5.3.1 Creating a Class Hierarchy

In creating a larger set of classes, it makes sense for classes not only to inherit from the existing class hierarchy, but also to develop a hierarchy themselves. This will take some planning beforehand when trying to determine how to organise an information model. Nevertheless, the advantages are significant once done:

- When developing a class hierarchy, factor out information common to multiple classes in superclasses and reuse that superclass’s information.
- Changing (or inserting) a class further up in the hierarchy automatically changes the operation of the lower classes, eliminating a need to change any of the lower classes.

For example, consider the earlier Email Address class. Assume a program to implement all the features of an Email Address has been created. Now assume that the model is to create an EDI class called Telnet Address. Email Address and Telnet Address have many similar features: both are addresses used to deliver and/or receive from, both have user names and types, etc.

One alternative would be to open up the Email Address class and copy over much of the previously defined information. A far better plan is to factor out the common information for Email Address and Telnet Address into a more general class. This may be a lot of work just for the classes Email Address and Telnet Address, but once FTP Address, URL Address, Gopher Address, Archie Address, etc. are added, having common operation in a reusable superclass significantly reduces the amount of work done overall. Figure 11 provides an example of a class hierarchy beginning with “Address” as the top class.



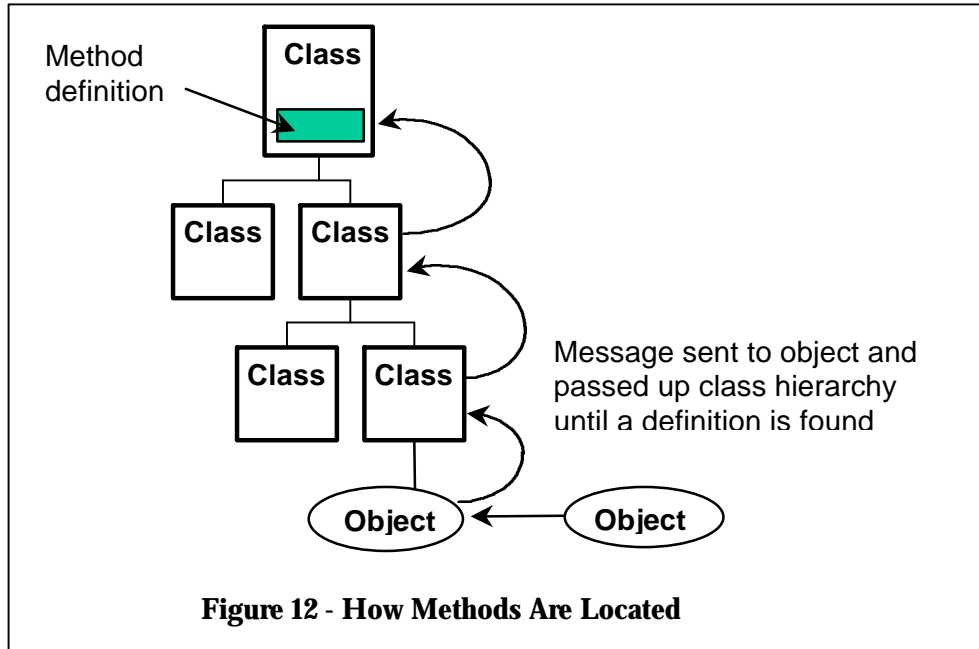
2.3.5.3.2 How Inheritance Works

How is it that instances of one class can automatically get variables and methods from the classes further up in the hierarchy?

For instance variables, when a new instance of a class is created, a “slot” is also created for each variable defined in the current class and for each variable defined in all its superclasses. In this way, all the classes combine to form a template for the

current object and then each object fills in the information appropriate to its situation.

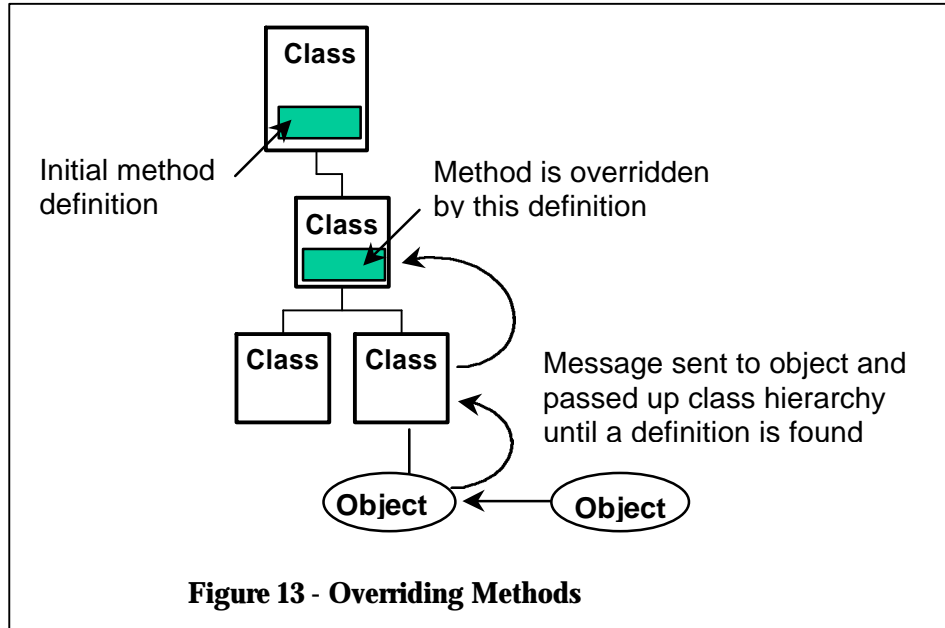
New objects have access to all the methods of its class and its superclasses. The same method may be defined differently at different levels within the same class hierarchy. The definition used will be the lowest found in the hierarchy. That is, if a method is called on a particular object, the object’s class is first checked for the definition of that method. If the method is not defined in the object’s class, that class’s superclass is checked, and so on up the chain until the method definition is found (see Figure 12).



Things get complicated when a subclass defines a method that has the same signature (name and number and type of arguments) as a method defined in a superclass. In this case, the method definition that is found first (starting at the bottom and working upward toward the top of the hierarchy) is the one that is actually executed. Because of this, a method can purposefully be defined in a subclass that has the same signature as a method in a superclass, which then “hides” the superclass’s method. This is called overriding a method.

Overriding a method is creating a method in a subclass that has the same signature (name, number and type of arguments) as a method in a superclass. See Figure 13.

All object-oriented programming languages allow some form of inheritance although their implementation may differ. Some use one form called single inheritance that means that each class can have only one superclass (although any given superclass can have multiple subclasses). Others permit more than one superclass. In these cases a subclass inherits combined variables and methods from all those classes.



2.4 THE SHIFT TO OBJECT-ORIENTED TECHNOLOGY

The Object Oriented approach to EDI standards will specify the requirements of an Open-edi framework that describes many potential scenarios within a distributed computing environment, using standard building blocks or components. These components are the class diagrams which reflect the actual business need. The class diagrams are defined and agreed upon in the standards process. This is a significant shift from today's message design standards process in which most of the business information requirements are captured in the EDI implementation conventions versus the EDI standard itself. It should be emphasized that this shift to an objected oriented approach is not likely to simplify the EDI standards development process. Many of the business representatives in the EDI standards process will be required to relearn how EDI standards are developed in order to capture their business knowledge in developing the new Open-edi frameworks.

The EDI business representatives need to take the lead in developing Open-edi frameworks and components, since many software developers and FSV-oriented organizations who know very little of the business environment are now attempting to develop and package these as "de facto standard" component-based solutions. These ad-hoc solutions are targeted to run on existing commercially available distributed computing options (most of which rely of the use of objects), the same options that AC.1 believes are the implementations of the FSV portion of Open-edi. Therefore IDEF and OOT models are the mechanisms to capture this knowledge and allow the construction of EDI components, while international standardization will ensure that business needs are accurately communicated and the EDI components conform to Open-edi frameworks.

The specification of an Open-edi framework will require:

- rigorous modeling of activities to identify the object classes that are required;
- development of detailed process models and state diagrams to describe the time sequence of events within the scenarios;

- the selection of existing classes or definition and registration of new classes for the EDI class library;
- identification of possible combinations of activities that describe a scenario(s).

The analysis of the business process and incorporation of the information transfer details will occur in the standards development process by engaging the EDI community (the content experts) and various modeling experts (who capture the knowledge required to specify the frameworks). The deliverables from standards development process include:

- framework development and maintenance;
- standard object development and maintenance;
- scenario registration and maintenance;
- certification specifications for conformance to the Open-edi frameworks.

The scenarios will detail the objects used, and the object interfaces will allow for the development of commercial EDI serving software to be built for various syntax representation and transport mechanisms including UN/EDIFACT. The registration of classes and scenarios will allow them to be reused without prior agreement as long as the trading partners' EDI servers support them.

3 OPEN-EDI FRAMEWORK & SCENARIO EXAMPLES

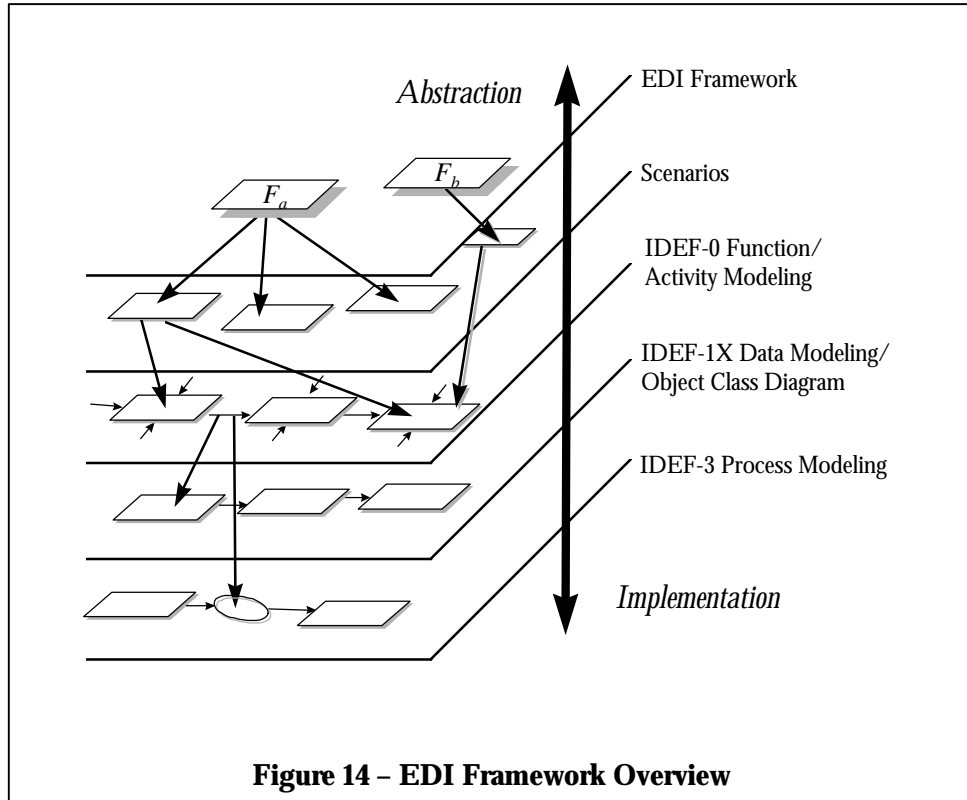
3.1 OVERVIEW OF FRAMEWORKS

A framework in essence provides a statement of the boundary of the problem domain; i.e., the purpose, what viewpoint and which context the underlying model is trying to communicate. The purpose is a statement of the goals of the modeling activities (e.g., what information needs to be assembled, what decisions this information is supposed to support, what consensus is to be achieved, etc.). The viewpoint establishes how the reader will interpret the model and how the modeler will constrain the idealization or abstraction of the activities that occur in the system under study. The context establishes the interpretation and scope of the model as part of a larger scope.¹¹

The EDI Framework development process first defines the EDI problem that describes an inter-company business goal and create a high level IDEF0 Function diagram (A-0 level), typically as a single function (goal) representing the EDI Framework. The EDI Framework goal in the diagram is further decomposed into a more granular level of IDEF0 functions (A0 level). These functions are then further decomposed into activities until outputs to the activities are obvious data structures that can be transformed into IDEF1X and object class representations. This would conclude the static portion of the modeling effort. The next step is dynamic IDEF3 Process Modeling, all within the scope of the problem domain. The scenarios within the EDI Framework are the last to be developed, in which several IDEF0 functions (at the A0 level) and combinations of functions are selected based on the interdependencies between functions. A selected scenario includes the underlying IDEF1X, IDEF3, and object classes, which are more implementation specific models.

¹¹ Excerpts from Mayer, Painter, deWhite, “IDEF Family of Methods for Concurrent Engineering and Business Re-engineering Applications”, KBSI (1992).

The illustration below (Figure 14) attempts describe how a framework is a high level abstraction of the IDEF and OOT models while the lower models are more implementation specific. Two EDI Frameworks are illustrated on the highest level plane. It shows that scenarios themselves can reuse various IDEF-0 functions that can be somewhat generic; an example of this is the A5 Request Order Status function in Appendix A discussed below. The illustration does not show the full decomposition of the IDEF-0 model that could be several additional planes but are not included for the sake of simplicity.



Scenarios bridge the gap between the function/activity models and the EDI framework by selecting and describing mandatory and conditional activities that accomplish the business goal of performing a given function within the scope of the EDI framework. Within the context of a scenario, business objects 1) are created, 2) interact, and 3) perform operations that conform to the specifications of the scenario. Specifically, there may be many possible scenarios within an EDI framework, and trading partners will advertise which scenarios they are able to participate in. The requesting trading partner will have no knowledge of how the scenarios and business objects are implemented, but through the specifications of the scenarios (via the object interfaces) they will know what data structures are returned and the behavior of the objects within the scenario.

3.2 CATALOG ORDER FRAMEWORK

The catalog order framework is shown in Appendix A. The A-0 diagram or node of the IDEF-0 function model defines the context of the framework in which the complete model is specified. It describes a very high level statement which is the act of ordering from a catalog and receiving a confirmed order. It defines the key role players (buyer and seller) involved as mechanisms to the function/activity and various business practices as constraints or controls to the function/activity.

The A0 node is the first level decomposition of framework providing a more granular viewpoint of the business function involved when ordering from a catalog. Each box defines the boundaries for the specified function, which are further decomposed as activities at the second level in nodes A1 through A5. The A0 node also illustrates that activities are **interdependent** within the scope of the specified EDI framework; e.g., node A2 Process Buyer Information, and node A4 Process Order Information where A4 requires a valid buyer ID. Variations of the base scenario are possible through combinations of the conditional activities A1 Get Catalog, A3 Request Price Quotation, and A5 Request Order Status. A1 Get Catalog can be executed independently or conditionally if either A4 Process Order Information or A3 Request Price Quotation are rejected due to an invalid catalog error. A more complete IDEF model is included in Appendix B.

Five business functions depicting the process of a Buyer executing a Catalog Order with a Seller are shown. Business function A1 is an optional business function a Seller may offer to any potential Buyer to provide an electronic version of the Seller’s catalog on request. Business Function A2 depicts a first time Buyer initiating a relationship with a Seller by providing required buyer information, confirmed by receiving a Seller’s Buyer ID from the Seller. Business function A3 (provide a price quote to the Buyer for selected product(s) on request) is another optional business function that the Seller may offer to a Buyer after a valid Seller’s Buyer ID has been assigned. Business function A4 depicts the process of a Buyer ordering items from a catalog, having previously established a relationship with the Seller by providing Buyer information and receiving a Buyer ID. Business function A5 is another optional business function where the Seller provides order status information to the Buyer on request.

Certain assumptions are made as part of these business functions. The Buyer has a copy of the Seller’s catalog, provided electronically via business function A1 or otherwise by reference. Both Buyer and Seller have a private/public security key pair using the same security method. How the public keys are made available is not important for this discussion. The cost of each item is not always included in business function A4, as this is calculated from the listed Product Unit Price and the Quantity the Buyer orders. Note also that the Price Quote(s) for the Buyer may have already been established through the execution of business function A3. A Price Quote would be provided in that case. However, absent business function A3, product price and currency are required when the catalog offers a product in multiple currencies.

The Seller requires the buyer information before any Catalog Order is placed. As soon as the required buyer information is transmitted, the Buyer will indicate to the Seller a time in which a response is expected. Receiving no response in the specified time period terminates the business function. The benefit of a Buyer, having provided information about itself prior to ordering, is that the amount of information to be exchanged and the number of steps required to subsequently place a Catalog Order are reduced. This results in saving both the Buyer and the Seller processing time that reduces the cost of doing business.

Certain interdependencies are assumed in the execution of these business functions. Whenever business function A1 (get Catalog) is invoked, business functions A3 (Request Price Quotation) and A4 (Process Order Information) require Buyer reference to the catalog to determine what to get price quotes on, or what to order. Also, whenever business function A3 (Request Price Quotation) is invoked, business function A4 (Process Order Information) will be executed only if the Buyer agrees with the Price Quote. Thirdly, business function A5 (Request Order Status) will only be invoked in connection with business function A4 (Process Order Information). Therefore, while business functions that may be packaged as are (A1, A2, A4), (A1, A2, A3, A4), (A1, A2, A4, A5), (A1, A2, A3, A4, A5), (A2, A3, A4), (A2, A4), (A2, A3, A4, A5), and (A2, A4, A5) business

functions that may be logically executed are A1, A2, A3, A4, and A5 individually, or combinations A1+A2, A2+A3, A2+A4, A2+A4+A5, and A4+A5.

It should be noted that additional business functions could be added to the Catalog Order Framework, e.g. exchange of security keys, invoice and payment. Such an expansion of the business case would provide further illustration of the power of reuse of business modeling and object class components.

Business function A1: a potential Buyer requests an electronic version of the Seller’s catalog. The request is honored unconditionally by the Seller, assuming the Buyer is capable of receiving the catalog information based on invocation of this business function.

- Buyer requests a catalog from the Seller
- Seller provides the Buyer with the requested catalog

Business function A2: a Buyer wanting to receive price quote(s) on product(s) offered by the Seller, or wanting to place a Catalog Order must have a Seller assigned Buyer Identification. This business function must be used to obtain such an ID. The Buyer will supply the Seller with up to five information parcels before receiving a Seller assigned Buyer Identification:

- Buyer provides Buyer Information, Billing Information, Ship-to Information (if different from Billing), and Accounting Information. In order to ensure that the Buyer can be validated and to keep the Account Information secure, the Account Information is signed and encrypted using a private/public security keying method.
- Buyer provides to Seller a time period in which to expect the Buyer Identification response. If the Buyer Identification is not received in that time period, the business function is terminated.
- Seller either assigns the Seller’s Buyer ID or rejects the request (stating the reason) and sends it to Buyer to complete the business function.

Business function A3: a Buyer requesting price quote(s) on a Seller’s product(s) will supply the Seller with two information parcels before the request can be acknowledged. The Seller has Buyer information on file that is keyed to the Seller’s Buyer ID. There is no need to resend Buyer information unless the Buyer has changed this information since previously sending it to the Seller. Knowing all that is necessary about the Buyer and the Line Item Quantity(ies) of the product(s) to be price quoted, the Seller has sufficient to information to quote the price(s).

- Buyer sends to Seller a Price Quote Request containing Seller’s Buyer ID and price quote request information
- Buyer provides to Seller a time period in which to expect the Price Quote Response. If an acknowledgement is not received in that time period the business function is terminated.
- Seller either provides a Price Quote Response or rejects the request (stating the reason) and sends it to the Buyer to complete the business function.

Business function A4: a Buyer placing a Catalog Order with a Seller will supply the Seller with three information parcels before the Order can be acknowledged. This is because the Seller has Buyer information on file that is keyed to the Seller’s Buyer ID. There is no need to resend Buyer information unless the Buyer has changed this information since previously sending it to the Seller.

- Buyer sends to Seller a Catalog Order containing Seller’s Buyer ID and Order Information.
- Buyer provides to Seller a time period in which to expect the Order Identification response. If an acknowledgement is not received in that time period the business function is terminated.
- Seller either assigns the Order Identification or rejects the request (stating the reason) and sends it to the Buyer, to complete the business function.

Business function A5: along with placing a Catalog Order with a Seller the Buyer also wants to determine the status of the order. An Order Reference ID, provided in the Order Identification Information response from the Seller in business function A4, must be referenced in the Order Status Request.

- Buyer provides to Seller an Order Status Request containing Order Identification.
- Buyer provides to Seller a time period in which to expect the Order Status response. If an acknowledgement is not received in that time period the business function is terminated.
- Seller either provides the Order Status Information or rejects the request (stating the reason) and sends it to the Buyer, to complete the business function.

3.2.1 DETAILED EXPLANATION OF BUSINESS FUNCTION A1

A Buyer is interested in finding out information about products from a supplier, referred to as the Seller. The Buyer decides to request an electronic version of a product catalog from the Seller. The Buyer expects that the Seller will unconditionally honor a request.

The Buyer starts the business function by sending to the Seller the following information:

- a. Buyer Information to include:
 - Buyer’s name [Business name by which the Buyer wants to be known by the Seller]
- b. Catalog Identification to include
 - Catalog ID (if required; is required when the Seller has multiple catalogs) [Unique identification of a Seller’s catalog]

The Seller responds by sending the requested catalog to the Buyer to complete the business function:

c. Catalog Information

- Product Identification [Unique identification of a product]
- Unit Price Amount [Monetary amount cost per single item (or per unit of measure) of the product]
- Unit Price Currency Code (if required; is required when the catalog offers the product in multiple currencies) [Identification of the currency of the Unit Price Amount]
- Line Item Unit of Measure ID (if required) [Unit of measure for selling bulk products]
- Product Characteristics Type (if required) [Product variation such as color, size, etc.]
- Product Characteristics Code (if required) [Product offerings within a product variation]
- Delivery Method [Seller provided options for means and timing of delivery per order]

3.2.2 DETAILED EXPLANATION OF BUSINESS FUNCTION A2

A Buyer finds one or more items in a Catalog which the Buyer needs. The Buyer has never conducted business with the Seller before. Thus the Buyer expects that the Seller will need specific information from the Buyer before the Catalog Order can be placed using business function A4.

The Buyer starts the business function by sending to the Seller the following information:

a. Buyer Information to include:

- Buyer's name [Business name by which the Buyer wants to be known by the Seller]

b. Billing Information to include:

- Address [Address to which the Buyer wants the bill to be sent]
- Contact Name [Name of the person to whom the Buyer wants billing inquiries to be directed]

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

- Phone Number [Telephone number of the person to whom the Buyer wants billing inquiries to be directed]
- c. Ship-to Information to include (only required if different from Billing Information):
- Address [Address to which the Buyer wants ordered goods to be shipped]
 - Contact Name [Name of the person to whom the Buyer wants shipping inquiries to be directed]
 - Phone Number [Telephone number of the person to whom the Buyer wants shipping inquiries to be directed]
- d. Accounting Information (signed and encrypted) to include:
- Credit Card Number [Identification of the credit account that the Buyer chooses to provide to the Seller as a credit reference]
 - Credit Card Holder Name [Name of the owner of the credit account the Buyer chooses to provide to the Seller as a credit reference]
 - Credit Card Issuer Name [Name of the financial institution issuing the card]
 - Type of Credit Card [Type of credit account, recognized by the credit card industry, that the Buyer chooses to provide to the Seller as a credit reference]
 - Expiration Date [Date on which the credit account that the Buyer chooses to provide to the Seller as a credit reference is no longer valid]

To conclude the exchange the Buyer sends to the Seller the Response Time Information to indicate by what date a response is expected from the Seller. If no response is received the business function is terminated.

- e. Buyer ID Respond-By Date Information to include:
- Respond-by Date [Final date on which the Seller can respond to the Buyer with a Buyer ID before the business function is terminated]

When the Respond-By Date Information is received, the Seller assigns a Seller's Buyer ID or rejects the request (stating the reason) and sends it to Buyer to complete the business function:

- f. Buyer Identification Information to include:
- Seller's Buyer ID [Seller assigned identification by the which the Seller uniquely recognizes a Buyer]

OR

g. Buyer ID Rejection Notice to include:

- Type [Code for Seller’s reason for not assigning a Seller’s Buyer ID]
- Reason [Seller stated reason for not assigning a Seller’s Buyer ID to the Buyer, e.g. insufficient Billing Information, invalid credit account, etc.]

3.2.3 DETAILED EXPLANATION OF BUSINESS FUNCTION A3

A Buyer, having a Seller assigned Seller’s Buyer ID, is aware that the Seller offers special purchase prices for regular (frequent) customers as well as volume discounts. Also, when Unit Price Amount is not specified in the Seller’s catalog for certain products, the Buyer is expected to request a Price Quote. Since the Seller knows the Buyer, the Buyer only needs to provide the Seller’s Buyer ID, the products for which a Price Quote is requested, and the Price Quote Respond-By Date Information. The Seller will respond either with Price Quote(s) or reject the request (stating the reason) and send it to the Buyer to complete the business function.

The Buyer starts the business function by sending to the Seller the following information:

a. Buyer ID

- Seller’s Buyer ID [Previously defined]

b. Price Quote Request Information

- Product Identification [Previously defined]
- Unit Price Currency Code (if required); is required when the catalog offers the product in multiple currencies [Previously defined]
- Product Quantity [Number of items (or units of measure) of the product to be price quoted]
- Line Item Unit of Measure ID (if required) [Previously defined]
- Product Characteristics Type (if required) [Previously defined]
- Product Characteristics Code (if required) [Previously defined]
- Delivery Method [Previously defined]

To conclude the exchange the Buyer sends to the Seller the Price Quote Response Time Information to indicate by what date a response is expected from the Seller. If no response is received the business function is terminated.

c. Price Quote Respond-By Date Information to include:

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

- Price Quote Respond-by Date [Final date on which the Seller can respond to the Buyer with a Price Quote before the business function is terminated]

When the Price Quote Respond-By Date Information is received, the Seller provides a Price Quote Response or rejects the request (stating the reason) and sends it to the Buyer to complete the business function:

d. Price Quote Response to include:

- Product Identification [Previously defined]
- Price Quote [Monetary amount]
- Price Quote Currency Code (if required; is required when an order is based on a Price Quote and the product is offered in multiple currencies) [Identification of the currency of the Price Quote]
- Product Characteristics Type (if required) [Previously defined]
- Product Characteristics Code (if required) [Previously defined]
- Delivery Method [Previously defined]
- Price Quote Reference Number [Seller assigned identification of a Price Quote for a product, to be used by the Buyer as a reference when ordering]
- Price Quote Expiration Date [Date on which a Price Quote for a product is no longer valid]

OR

e. Reject Price Quote Notice to include:

- Price Quote Reject Type [Code for the Seller’s reason for rejecting the request]
- Quote Rejection Reason [Seller stated reason for not providing a Price Quote Response, e.g., no Seller’s Buyer ID, insufficient Price Quote Request Information, etc.]

3.2.4 DETAILED EXPLANATION OF BUSINESS FUNCTION A4

A Buyer, having a Seller assigned Seller’s Buyer ID finds one or more items in a Seller’s Catalog that the Buyer needs. Since the Seller knows the Buyer, the Buyer only needs to provide the Seller assigned Seller’s Buyer ID, the Ordering Information and Order ID Respond-By Date Information. The Seller will respond either with an assigned Order Identification or reject the request (stating the reason) and send it to the Buyer to complete the business function

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

The Buyer starts the business function by sending to the Seller the following information:

a. Buyer ID:

- Seller’s Buyer ID [Previously defined]

b. Order Information to include:

- Product Identification [Unique identification of a product in the Seller’s catalog]
- Price Quote Reference Number (if required; is required when an order is based on a Price Quote) [Previously defined]
- Unit Price Amount (if required; is required when the catalog offers the product in multiple currencies) [Monetary amount cost per single item (or per unit of measure) of the product as stated in the Seller’s catalog]
- Unit Price Currency Code (if required; is required when the catalog offers the product in multiple currencies) [Previously defined]
- Line Item Quantity [Number of items (or units of measure) of the product to be ordered]
- Line Item Unit of Measure ID (if required) [Unit of measure as stated in the Seller’s catalog for selling bulk products]
- Product Characteristics Type (if required) [Previously defined]
- Product Characteristics Code (if required) [Previously defined]
- Delivery Method [Means and timing of delivery per order as selected by the Buyer from Seller provided options]

To conclude the exchange the Buyer sends to the Seller the Response Time Information to indicate by what date a response is expected from the Seller. If no response is received the business function is terminated.

c. Order ID Respond-By Date Information to include:

- Respond-by Date [Final date on which the Seller can respond to the Buyer with an Order ID before the business function is terminated]

When the Respond-By Date Information is received, the Seller assigns an Order Identification or rejects the request (stating the reason) and sends it to the Buyer to complete the business function:

d. Order Identification Information to include:

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

- Order Reference ID [Seller assigned order identification for tracking the status of an order in a Buyer’s account until payment is made]

OR

e. Order Rejection Notice to include:

- Type [Code for the Seller’s reason for rejecting the order]
- Reason for rejection [Seller stated reason for not assigning an order from the Buyer, e.g. no Seller’s Buyer ID, insufficient Order Information, insufficient credit, etc.]

3.2.5 DETAILED EXPLANATION OF BUSINESS FUNCTION A5

A Buyer, having received an Order Reference ID in the Order Identification Information, wants to determine the status of the order. Upon requesting the order status, the Seller will respond either with Order Status Information or reject the request (stating the reason) and send it to the Buyer to complete the business function.

The Buyer starts the business function by sending the Seller the following information:

a. Order Status Request:

- Seller’s Buyer ID [Previously defined]
- Order Reference ID [Previously defined]

To conclude the exchange the Buyer sends to the Seller the Response Time Information to indicate by what date a response is expected from the Seller. If no response is received the business function is terminated.

b. Order Status Request Respond-By Date Information to include:

- Respond-by Date [Final date on which the Seller can respond to the Buyer with Order Status Information before the business function is terminated]

When the Order Status Respond-By Date Information is received, the Seller provides Order Status Information or rejects the request (stating the reason) and sends it to the Buyer to complete the business function:

c. Order Status Information to include:

- Order Reference ID [Previously defined]
- Product Identification [Previously defined]

- Product Status [Seller’s code for the status of processing each product ordered, e.g., logged, out of stock, percent filled if partially complete, complete, etc]

OR

d. Reject Order Status Notice to include:

- Order Status Reject Type [Code for the Seller’s reason for rejecting the request]
- Reason for rejection [Seller stated reason for not providing Order Status Information]

3.3 IDEF DIAGRAMS FOR CATALOGUE ORDER FRAMEWORK

3.3.1 IDEF0 (ACTIVITY) DIAGRAMS

A-0 Diagram: Catalog Order Framework (*Diagram Page 1*)

Purpose: This model describes a buyer ordering from a seller’s catalog. The buyer may or may not have an established business relationship with the seller.

Viewpoint: the model is described from the viewpoint of the buyer.

A0 Diagram: Ordering from Catalog (*Diagram Page 2*)

Having determined that there is a requirement to order, minimum use of the framework provides the business functions for two possibilities, depending upon whether or not the Buyer has a Seller’s Buyer ID (i.e., whether or not the Buyer and the Seller already have a business relationship).

If the Buyer does not have a Seller’s Buyer ID, the Buyer provides information to the Seller after which the Buyer receives a Buyer Identification number from the Seller (Seller’s Buyer ID) or in case of rejection a notice with reason for the rejection (shown as Activity 2).

If the Buyer has a Seller’s Buyer ID (i.e., a business relationship is established), the order information can be provided from the catalog. Ultimately, the Seller provides the buyer with an Order Reference ID or a rejection notice stating the reason for rejection (shown as Activity 4).

Three additional business functions (shown as Activities 1, 3, and 5) are included in the Catalog Order Framework that optionally provide for Get Catalog, Request Price Quotation, and Request Order Status.

A1 Diagram: Get Catalog (*Diagram Page 3*)

The Buyer requests an electronic version of the Seller’s catalog from the Seller. The Seller honors the request unconditionally by providing the Buyer with an electronic version of the requested catalog.

A2 Diagram: Process Buyer Information (*Diagram Page 4*)

In order to establish a business relationship with the Seller, the Buyer provides all the required Buyer Information to the Seller, along with a time period in which to expect the Buyer Identification response. The Seller either assigns the Seller’s Buyer ID or rejects the request (stating the reason) with a Reject Buyer Information Notice.

A3 Diagram: Request Price Quotation (*Diagram Page 5*)

The buyer submits a Price Quote Request containing the s ID to the Seller along with a time period in which to expect the Price Quote Response. The Seller either provides a Price Quote or rejects the request with a Reject Price Quote Notice.

A4 Diagram: Process Order Information (*Diagram Page 6*)

The Buyer submits all of the Order Information, including a Price Quote if requested and upon which the order is based, along with the Seller’s Buyer ID to the Seller. Again, the Buyer specifies a time period in which to expect the order to be acknowledged. The Seller either provides the Order Identification confirming the order, or rejects the order with a Reject Order Notice.

A5 Diagram: Request Order Status (*Diagram Page 7*)

In order to determine the status of the order the Buyer sends an Order Status Request to the Seller, including the Seller’s Buyer ID and the Order Identification. The Buyer also specifies a time period in which to expect the Order Status Information. The Seller either provides the Order Status or a Reject Order Status Notice.

A21 Diagram: Submit Buyer Information (*Diagram Page 8*)

The Buyer information provided to the Seller in order to receive a Seller’s Buyer ID is the Buyer’s Name, Billing Information (Address, Contact Name and Phone Number), Ship-to Information (Address, Contact Name and Phone Number) if different from the Billing Information, and Account Information (Credit Card Number, Credit Car Holder Name, Type of Credit Card and Expiration Date). Note, the account information is encrypted for security and signed to validate the information source.

A41 Diagram: Submit Order Information (*Diagram Page 9*)

The order information provided by the Buyer to the Seller is: the Seller’s Buyer ID, the Delivery Method, Product Identification, Quantity, Unit Price Amount and Unit Price Currency Code (if multiple Unit Prices) and Product Characteristics Type and Code. The Product Identification, Unit Price Amount with Currency Code and Product Characteristics Type and Code are extracted from the Seller’s catalog.

3.4 IDEF DIAGRAMS FOR CATALOGUE ORDER SCENARIO

3.4.1 IDEF0 (ACTIVITY) DIAGRAMS

A-0 Diagram: Catalog Order (*Diagram Page 1*)

Purpose: This model describes a buyer ordering from a seller’s catalog. The buyer may or may not have an established business relationship with the seller.

Viewpoint: the model is described from the viewpoint of the buyer.

A0 Diagram: Order from Catalog (*Diagram Page 2*)

Having determined that there is a requirement to order, there are two possibilities depending upon whether or not the buyer has a Seller’s Buyer ID (i.e., depending on whether or not the buyer and the seller already have a business relationship).

If the buyer does not have a Seller’s Buyer ID, the buyer provides information to the seller after which the buyer receives a Buyer Identification number from the seller (Seller’s Buyer ID) or in case of rejection a notice with reason for the rejection.

If the buyer has a Seller’s Buyer ID (i.e., a business relationship is established), the order information can be provided from the catalog.

Ultimately, the seller provides the buyer with an Order Reference ID or a rejection notice stating the reason for rejection.

A1 Diagram: Provide Buyer Information (*Diagram Page 3*)

The buyer information provided to the seller in order to receive a Seller’s Buyer ID is the Buyer’s Name, Billing Information (Address, Contact Name and Phone Number), Ship-to Information (Address, Contact Name and Phone Number) if different from the Billing Information, and Account Information (Credit Card Number, Credit Card Holder Name, Type of Credit Card and Expiration Date). Note, the account information is encrypted for security and signed to validate the information source.

A3 Diagram: Provide Order Information (*Diagram Page 4*)

The order information provided by the buyer to the seller is: the Seller’s Buyer ID, the Delivery Method, Product Identification, Quantity, Unit Price Amount and Unit Price Currency Code (if multiple Unit Prices) and Product Characteristics Type and Code. The Product Identification, Unit Price Amount with Currency Code and Product Characteristics Type and Code are extracted from the Seller’s catalog.

3.4.2 IDEF3 (PROCESS) DIAGRAMS

3.4.2.1 IDEF3, Process Flow Description (PFD) Diagrams

Top level Diagram (Ordering from catalog) (*Diagram Page 5*)

In the example, complete buyer data must be supplied at some point resulting in the seller assigning a Seller's Buyer ID. Then, the order information can be provided. Finally, the seller returns an Order Reference ID to the buyer.

Level Two Diagram (Provide Buyer Information) (*Diagram Page 6*)

The buyer information includes the Buyer's Name, Billing Information, Account information, and Ship-To Information. For simplicity, it is assumed that there is only one delivery point for the entire order.

Level Three Diagram (Provide Order Information) (*Diagram Page 7*)

The order information includes the Seller's Buyer ID, product information, delivery method and unit price amount and currency code. Products are identified by supplying their identification and quantities, and, if applicable, additional characteristics. The delivery method and unit price amount and currency code are optional data which complete the order.

3.4.2.2 IDEF3, Object State Transition Network (OSTN) Diagrams

OSTN Diagram for Object: Provide Buyer Information (*Diagram Page 8*)

The state of *No buyer Information* is moved to the state of *Buyer Information received* through the provision of Buyer's Name, Billing Information, Ship-to Information and Account Information.

OSTN Diagram for Object: Provide Order Information (*Diagram Page 9*)

The Order is moved from the state of *Blank Order* to the state of *Order Placed* by the provision of the Seller's Buyer ID, the Delivery Method, Product Identification, Quantity, Unit Price Amount and Currency Code, and Product Characteristics.

OSTN Diagram for Object: Provide Order Reference Number (*Diagram Page 10*)

Moving from the state of *No Order Reference ID* to *Order Reference ID Received* requires order information to be provided.

OSTN Diagram for Object: Provide Seller's Buyer ID (*Diagram Page 11*)

Moving from the state of *No Seller's Buyer ID* to the state of *Seller's Buyer ID Requested* required the provision of Buyer Information. In order to move to the state of *Seller's Buyer ID on File* the Seller's Buyer ID has to be received.

3.4.3 IDEF1X (DATA) DIAGRAM (PAGE 12)

Reading an IDEF1X Diagram

The boxes are “entities”. These are persons, places, things, concepts, et cetera that are of importance in this context. Entities are “object classes” without behavior. The name of the entity is written above the box. (Boxes with rounded corners are considered “child entities” and have no relevance in this model. Along with other features mentioned above, they are an artifact of the normal use of IDEF1X for providing structure to relational databases.)

The links among entities are “relationships”. The relationships are labelled with a verb phrase written along the line. The relationship is normally read toward the end with the dot. The number of entity instances to be expected at each end of the relationship is called “cardinality”. A dot on the end indicates zero, one, or more. A dot with a P means one or more. No dot means one.

“Attributes” are equivalent to “data elements” in EDI. The names of the attributes describing an entity are listed in each box.

There are other features of a data model that are not covered here. Many other icons can be used in an E-R diagram, but are not of use in this structure. In addition, to be complete, there should be a description of each entity and attribute including its definition.

In this model:

- A *Buyer* has a *Seller's Buyer ID* by which the *Buyer* is known by the *Seller*.
- A *Buyer* has a “shipped to” *Site* and a “billed at” *Site*.
- The *Site* has a *Contact* person who can be telephoned for information.
- A *Buyer* has an *Account* and the *Account* has an expiration *Date*.
- The *Buyer* can place an *Order* with a *Seller*.
- The *Order* has a *Delivery* that has a *Date*.
- The *Order* has one to many *Line Items*.
- Each *Line Item* has a *Product*.
- Each *Product* has zero to many *Product Characteristics* such as color, size, et cetera.

4.1 INTRODUCTION¹²

At the March 1997 session of GE.1, two CRPs were discussed which contained substantive proposals for the future of UN/EDIFACT (TRADE/WP.4/CRP.123 and TRADE/WP.4/CRP.135). It was agreed that following consultation with the JRT at its April 1997 Singapore meeting, and a general consultative period concluding on 30 June 1997, the relevant aspects of both CRP's - amended by the comments received - would be merged into one document for approval at the September 1997 session of CEFACT. Therefore, this document, which is submitted on behalf of the EDIFACT Steering Group (ESG), contains substantive sections on strategy, organisation and empowerment which are for approval.

The Strategy for UN/EDIFACT

It is now nearly ten years since the UN/EDIFACT syntax became an ISO standard and eight years since the UN/ECE approved its first messages. The intervening period has seen;

- a substantial increase in the number of countries and regions participating in the work;
- a rapid growth in the number of messages in the UNTDID directory -- there are over 150 covering many different fields of application;
- a maturing of the development, production and publication process with good quality directories now being produced twice a year;
- large scale implementations based on UN/EDIFACT world wide -- mainly by major public and private organisations.

Further, all of the technical objectives set for UN/EDIFACT by WP.4 in 1987, have been successfully met.

However, at the strategic level, UN/EDIFACT has not had the impact that was predicted for it in two significant areas. The first is in the general rationalisation and simplification of business and administrative processes and the second is in the take up and use of the standard by Small and Medium sized Enterprises. (SME's)

Whilst the lack of impact in these areas is general to EDI, irrespective of the standard on which it is based, CEFACT needs to fully address these issues because of UN/EDIFACT's position as the global trading standard and because of their potential impact on our primary objective, which is Trade Facilitation.

Three years ago, in March 1994, WP.4 established the AC.1 group to act as a research and development group for UN/EDIFACT. In its interim report of March 1996

¹² This section contains the "Report from the Chairman of the EDIFACT Steering Group (ESG)" presented to GE.1 during the March 1997 session.

(TRADE/WP.4/R.1189), AC.1 identified a number of specific reasons which were and, still are, constraining implementation by SME's of UN/EDIFACT and EDI in general, whatever the standard used. The report went on to point to a number of emerging technologies and techniques, in particular object technology and information modeling, which they believed could be especially useful both to UN/EDIFACT and trade facilitation.

AC.1's report also coincided with a number of other developments including the:

- a. a growing understanding of the limits of the "bottom-up approach" to message design and the potential value of information modeling techniques;
- b. moves to simplify UN/EDIFACT messages and the agreement to harmonise implementation guidelines;
- c. recognition of the need to radically simplify business and administrative processes if SME's are to become the driving force behind the expansion of world trade;
- d. use of EDI and UN/EDIFACT over the Internet;
- e. evolution of the World Wide Web, the arrival of Java software, and the recognition of their potential for introducing SME's to EDI and Electronic Commerce;

Given the AC.1 report and these developments, the ESG has been working to prepare a coherent forward strategy for UN/EDIFACT which takes these into account within the context of CEFACT's Mission Statement. Accordingly, and after consultation, particularly with the JRT, the ESG now proposes, for incorporation into CEFACT's Work Programme, the following items based upon two inter-linked, parallel strategic objectives.

- 1) Continuation of the development and maintenance of UN/EDIFACT as the global message standard based on both batch and interactive syntaxes. (Strategic Objective 1 - Mainstream UN/EDIFACT)
- 2) Full development of the Object Oriented approach to the design of future messages. (Strategic Objective 2 - Object Oriented UN/EDIFACT)

Both strategic objectives shall fully take into account the special requirements of SME's for simple and stable messages which can be incorporated into effective and inexpensive applications.

Strategic Objective 1 - Mainstream UN/EDIFACT, requires no introduction. The current user base is made up of many significant and stable organisations from all the main regions of the world and implementations are continuing to grow (particularly in Asia). Messages continue to be developed (there are more than 55 new messages in development) and the efficiency of the maintenance process will improve significantly when the DMR process has been automated. Mainstream UN/EDIFACT is now a mature and stable standard which will continue to evolve for as long as its users wish it to do so. In line with the support provided by users, CEFACT needs to devote resources to the maintenance and publication of the standard although this may decrease as full automation of the DMR process comes into effect.

Recently, considerable effort has also been put into meeting the requirements of SME's and this must be continued. For example, currently there are projects underway in:

REFERENCE GUIDE "THE NEXT GENERATION OF UN/EDIFACT"

- a. the Message Development Groups of the JRT centred around the Harmonised Implementation Guidelines Group which appear very promising;
- b. EBES (the successor to the Western European EDIFACT Board) known as EDI-LITE, which, if developed, could provide SME's with a convenient and comprehensive UN/EDIFACT based Internet Web solution;
- c. SITPRO on the development of a UN/EDIFACT based approach to aligned forms which may be very relevant to the wider international trade arena and, therefore, to CEFACT;
- d. UKCEDIS on a fundamental proposal to exchange master files before the commencement of trading transactions to radically simplify the subsequent message exchange. This is part of an equally radical approach to the simplification of the whole business process based on a combination of simpler UN/EDIFACT messages and "value chain" analysis.

All of these developments are of significant interest and, in the cases of the EDI-LITE and aligned forms initiatives, have obvious potential for Java based Internet Web applications. Here, CEFACT has a direct interest through the previous work of WP.4, which did much to successfully develop and promote the UN aligned documentation system. Therefore, the possibility of Web Java Applets containing a relevant set of UN/EDIFACT based aligned documents is exciting.

Consequently, central to Strategic Objective 1, is the need to continue to encourage strongly the development of simpler UN/EDIFACT messages and sub-sets. However, proposed work should not duplicate that which is being carried out elsewhere. To this end, CEFACT may benefit from entering into understandings with specific organisations in order to both associate itself with relevant developments in other organisations and assist in the world wide dissemination of the results. Equally, the work done within the JRT's in support of SME's should be strongly supported and become an important part of CEFACT's Work Programme under the proposed mandate for a Permanent Working Group to empower the JRT (See section 4 below).

Strategic Objective 2, Object Oriented UN/EDIFACT

As noted earlier, apart from supporting SMEs, the other area where UN/EDIFACT (and indeed EDI) has not achieved the impact that was predicted for it ten years ago, is in the rationalisation and simplification of business and administrative processes. There are many reasons for this, but the fundamental one is that EDI and UN/EDIFACT did not start at the business process level. Rather, it was assumed that the considerable benefits of rationalising and automating the transaction data (e.g. a purchase order) which was part of a process, (e.g. purchasing) would stimulate a review of the process itself. In most cases this has not happened because, by approaching the issue "bottom - up," the highest level, being that of the process itself, was not reached.

While business practices from one business organisation to another are highly variable, depending on competitive strategies, experience and management style, they can be decomposed into business processes that are much more generic.

As the emerging work of AC. 1 is showing, the greatest benefit to be gained from an object oriented approach coupled with the rigorous use of modeling techniques, is that of a top-down approach. This means starting at the level of the business process, and decomposing this process, through a number of logical steps, all the way down to the semantic level. This decomposition is from business processes down to business functions (transactions) and then to activities or objects, data or classes, and, finally, abstract data or core business objects. Object Technology, currently an area of intense activity in major organisations, has already demonstrated its ability to deliver significant advances in business performance for all sizes of organisations.

For CEFACT, the implications are that it is now possible to begin to confront the high level issue of the rationalisation and simplification of the general business and administrative process while building upon and using the existing semantic content of UN/EDIFACT. [One of the great strengths of UN/EDIFACT has been the development , over many years, of its semantic content. In terms of the investment of world wide business and administrative expertise, it probably represents hundreds of man years. Under no circumstances should that investment be put in jeopardy]

This approach, if harnessed effectively and coupled with the practical experience of business and other techniques such as value chain analysis, could lead to major advances for SME's and for the wider area of Trade Facilitation. That is why the ESG believes that it extremely important to build upon and further develop the work of AC.1. Therefore, they propose a new but parallel objective in the strategy - Object Oriented UN/EDIFACT.

If this strategy is approved, users will be able to choose to follow one or both objectives, in accordance with their own needs. For example, some current users may wish to stay with Objective 1 for the foreseeable future; others may wish to move partially or entirely to objective 2 when it becomes a reality. New users, particularly SME's, will probably have their requirements initially met within Objective 1 but may seamlessly move to Objective 2 when products are available and when clear business advantages have been demonstrated.

Conclusion and Recommendation

In proposing this strategy with these two parallel objectives, it is important to appreciate that one objective is not more relevant than the other to CEFACT's long term goals. However, it should also be noted that under the CEFACT structure, users will be the main contributors to the work. Indeed, it must be emphasized that the successful completion of future work items developed under this proposed strategy will directly depend on users reactions and their commitment of resources to achieve the goals. In particular, despite its long term potential, Objective 2 – Object Oriented UN/EDIFACT, is unlikely to receive additional Secretariat support or resources. It will only become a reality if users support and resource it - which they appear to want to do. Therefore, the ESG believes that the two objectives do present a coherent strategy for the next phase of UN/EDIFACT which will take it into the new millennium. and they recommend it to the Plenary for approval.

4.2 MOVING FROM THE DATA MODEL TO TRADITIONAL EDI

Certain trends are observed by comparing the IDEF1X model (see Appendix A) with current X12 and UN/EDIFACT practices. However, these observations do not imply fixed rules and require an expert in current EDI to “interpret” the models in light of these standards. Since different experts would develop different message/transaction set from the same model representation, the objective

REFERENCE GUIDE “THE NEXT GENERATION OF UN/EDIFACT”

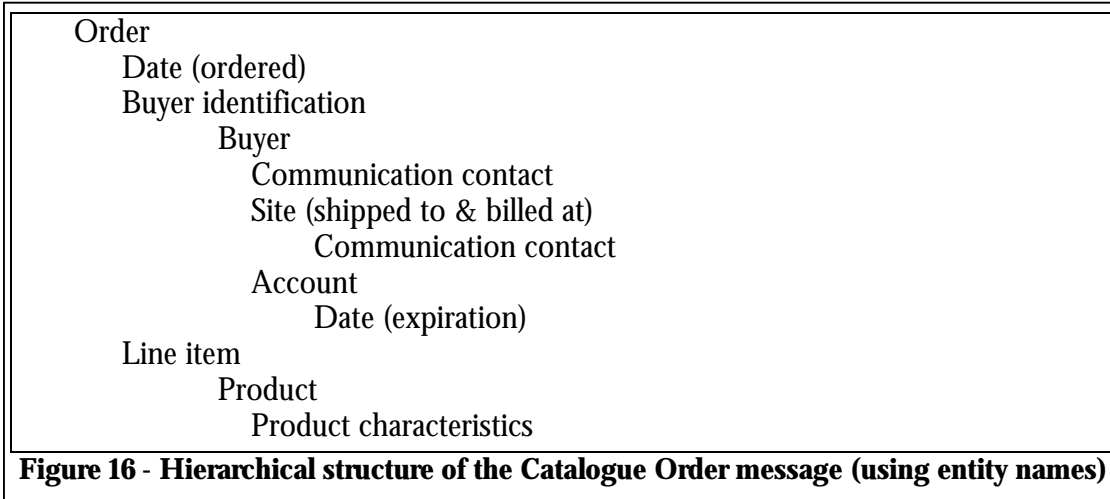
is to use these techniques to create future EDI Implementations. In additions the following problems have be identified with the current way of developing messages/transaction sets:

- a) the same information is represented in different ways in different messages/transaction sets;
- b) there is no well defined meaning for data elements, segments and messages/transaction sets. This makes the design of message/transaction set structures a black art;
- c) mapping from an IDEF1X model to the message/transaction set structure must be done manually.

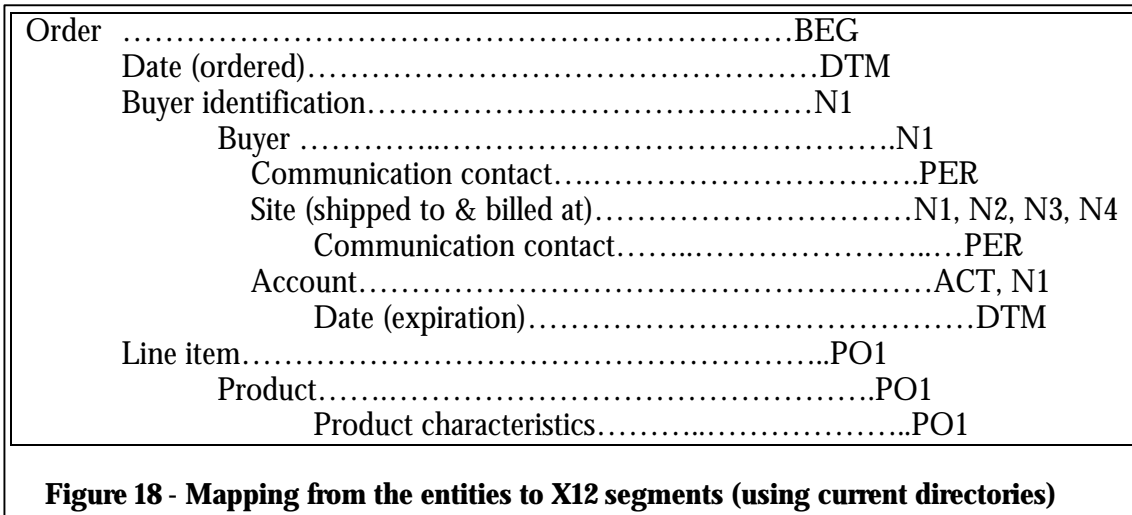
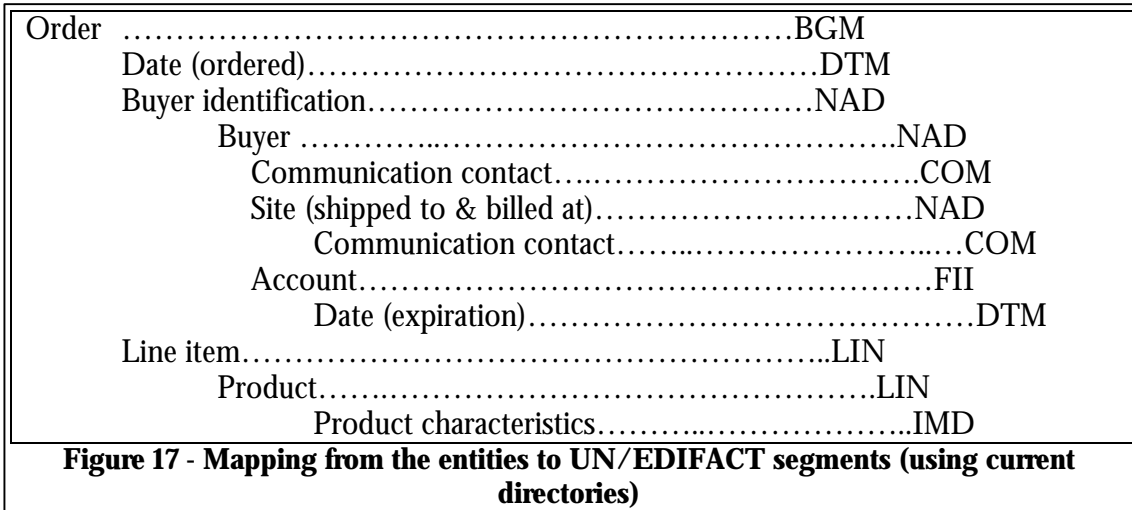
| IDEF1X to X12 | IDEF1X to EDIFACT |
|--|---|
| Each attribute is an element. | Identical |
| Each combination of entities having a one to one relationship is a segment. | Each entity is a segment. |
| Each entity that acts in multiple roles (has more than one relationship arrow coming into it in the diagram) is a generic segment (requires a segment qualifier). | Identical |
| Entities referred as one to many are segment groups (loops). | Identical |
| This model could be used to further define the EDI design discipline such as: | If each entity is a segment, composites and repeating data element can not be defined by the model. |
| Each entity is a composite data element. Each entity which can occur more than once is a repeating data element. | |
| IDEF3 to X12 | IDEF3 to EDIFACT |
| Each scenario represents one or more transaction sets (messages). The number of transaction sets (messages) is determined by an examination of the IDEF3 diagrams (Flow and Object Transition). If all data flow is in one direction without a transition point, there could be just one message. There must be at least one message for each data flow. | Identical |
| Figure 15 – Mapping from IDEF to EDI Standard Components | |

There are two steps in moving from the data model to EDI using the data directories from the current UN/EDIFACT or X12 directory sets. The first step - producing a hierarchical structure of the Catalogue Order message (using entity names) - is common, irrespective of whether the end goal is to map to UN/EDIFACT or X12. The second step - Mapping from the entities to segments (using current directories) - differs according to the EDI standard being used.

The first step is to draft the hierarchical structure of the message using the entity names from the model (rather than the UN/EDIFACT or X12 segment tags). This results in the structure shown in figure 16.



The second step in moving from the Data Model to EDI structures using today's directories is to map the entity names to segments by using the rules defined in table 1 above. In the case of UN/EDIFACT, the result is shown in figure 17 below. In the case of X12, the result is shown in figure 18.



An analysis of this two stage process immediately illustrates the problems faced by message designers and implementers: Having modeled the data requirements accurately using IDEF1X it is not then possible to map directly to either UN/EDIFACT or X12 segments since many of the segments are complex, hybrid structures performing several different (albeit related) tasks. For example, the hierarchical relationship between the line item and the product as shown in the model is lost within the EDIFACT structures (figure 17) because both are located in the same LIN segment. The X12 mapping (figure 18) highlights the opposite problem - sometimes the attributes of the entities from the data model require to be mapped to more than one segment. It is the view of the group that although these are very simple examples, if the conceptual (IDEF1X) model had been of a more complex operation then many more poor examples would have followed.

Since it is not possible to map automatically to simple EDI structures this means that there is a heavy reliance on knowledge of UN/EDIFACT and X12. The problem is augmented by the fact that often there may be two or three possible alternative segments to which an entity may be mapped which introduces a high degree of subjectivity and the increased possibility of inconsistent implementations. Hence the allusion to the 'Magic' of mapping from data models to EDI: if you ask a different expert you get a different answer.

This leads to the conclusion that the only way of introducing more predictability and 'science' (rather than art) into the process of mapping from models to EDI, is to adopt a new way of designing segments (though, not necessarily a new syntax). Whether the benefits of taking this step outweigh the cost has yet to be proved.

A note on the model review process

There is a view that if there is a business model showing both the conceptual view of the data and the relationships between the data, then whatever message is produced (either using current EDI directories or some different directory in the future) will be the definitive answer. But this would lead to the conclusion that every specific model would produce many specific segments.

A more considered approach may be to change all of the current message design groups into business modeling groups, establish a model review panel and a single message production group. (The premise is that each of the groups have the required skills and tools to perform the tasks.)

The business modeling groups would produce models of their business requirements.

The model review panel would assess all of the models for consistency, for example, to check the relationships between entities - this may result in some of the models being modified.

The message production group would turn the conceptual data view into an implementation view by creating messages. The message production group would not amend the data models. Each entity in the conceptual view (of the data model) would not necessarily result in the creation of a specific segment in the implementation view (i.e. the EDI messages). For example the entity 'buyer' and the entity 'seller' in the conceptual view would probably result in the creation of a segment called 'party' rather than two specific segments.

4.3 MOVING FROM THE DATA MODEL TO OBJECT ORIENTED EDI

4.3.1 CLASS DIAGRAM

The object technology approach to EDI is illustrated in this section using a class diagram based on a class library. A class diagram for the catalogue order scenario is represented in Diagram Page 13 (Appendix A) using an adaptation of the catalogue order conceptual model of the Diagram Page 12. In Diagram Page 13, object classes, or classes, replace “entities” modeled in Diagram Page 12. Relationships between the “entities” are replaced by associations between the classes in Diagram Page 13. Each class bubble is labelled in Diagram Page 13 with a class name. Internal to each class bubble is listed the attributes and methods belonging to the class, as described in Section 2.3. Variables are listed for each class above the horizontal line in the class bubble. Methods are listed below the line.

4.3.2 CLASS LIBRARY

In order for a class to be specified in a class diagram, it must first be defined in a class library. This section provides an illustrative example of a class library that provides the classes referred to by the classes in the catalogue order class diagram of Diagram Page 13. The class library in this illustration is comprised of primitive classes, foundation classes, core business classes and common business classes.

Primitive classes are at the lowest level in terms of attributes and methods, and deal with basic data types, (e.g., character, integer, Boolean values *true* and *false*) and run time operations performed directly by the language. Foundation classes use the primitive classes (and foundation classes) to build fundamental, reusable classes, (e.g., name, description, code list, identifier, quantity, currency) and are referred to as attributes in a class diagram for a scenario. (Note, the catalogue order class diagram of Diagram Page 13 specifies the attributes for each business class, but does not depict the primitive and foundation classes as bubbles along with their associations.) Sometimes referred to as abstract data types, foundation classes are prime candidates for standardization. Within the context of core business classes, foundation classes provide references for permitted values of attributes and formatting rules against which values of instantiated attributes are validated when a scenario is executed.

A core business class is a business superclass defined at a generalized level, allowing for specialised business subclasses, (i.e., common business classes) to inherit its attributes and methods. A core business class consists of one or more foundation class(es) and methods and/or one or more other core business class(es). Common business classes are defined as required for a scenario. Sometimes they can be reused within a scenario or other scenarios, and thus may also be candidates for standardization.

Classes in this illustrative class library are defined within class hierarchies (as described in Section 2.3) whenever possible in order to use the power of inheritance. Although not illustrated in this example, object technology provides additional flexibility in the construction of a class diagram by allowing attributes and methods to be added to the attributes and methods already contained in

standard classes in the class library. Conversely, attributes and methods that are part of standard classes may be nulled out in class diagrams.

This class library is based solely on the scenario and class diagram of the catalogue order scenario of Diagram Page 12 and Page 13. It is recognized that if some of the classes were modeled more generically, additional attributes would probably be present, and that also some of the classes may be subdivided. The classes were categorized as:

Primitive Classes

Foundation Classes

Core Business Classes

Common Business Classes

4.3.2.1 Primitive and Foundation Classes and Hierarchy

Foundation Classes are principally used to hold data, and the methods associated with them generally are quite simple such as Setting data and Retrieving data. Primitive and foundation classes support core and common business classes. See Appendix B for the Primitive and Foundation classes supporting the catalogue order scenario.

4.3.2.2 Core Business Classes

Core Business Classes are reusable classes in either the same or different scenarios. See Appendix B for the Core Business Classes supporting the catalogue order scenario.

4.3.2.3 Common Business Classes

Common business classes are defined as required for a scenario as specializations or subclasses of core business classes. See Appendix B for the Common Business Classes supporting the catalogue order scenario.

5 GLOSSARY

| | |
|---|--|
| Abstraction: | 1. Any model that includes the most important, essential, or distinguishing aspects of something while suppressing or ignoring less important, immaterial, or diversionary details. 2. Any relationship between a group of object types such that one object type represents a set of characteristics that are shared by other object types. |
| Activity: | 1. The term used during state modeling for any operation that takes significant time to execute. 2. A task that transforms inputs into outputs via the work of mechanisms under the instruction of controls. Activities occur over time and have identifiable outputs. |
| Activity model (IDEF0): | Describing decisions, actions and activities of an organization or system. |
| Activity unit: | An activity, that with its input, control, output, mechanism (ICOM) concepts, is a business process component at a low enough level that it is recognized as reusable, and is thus a candidate for standardization. (An activity unit can be represented as an IDEF0 model, and may be further analyzed using an IDEF3 process model.) |
| Aggregation: | 1. The whole-part relationship from an aggregate to its component parts. 2. The process of forming an aggregate object (class) from other objects (classes) as its component parts. |
| Association: | Any relationship between two objects or classes describing their interaction. |
| Attribute: | Any named property used as a data abstraction to describe its enclosing object, class, or extent. |
| Behavior: | The externally visible dynamics of an object or class in terms of its interactions with others (i.e., in terms of incoming and outgoing messages and exceptions). |
| Business life cycle: | The duration of existence or period of time over which a commercial enterprise operates. |
| Business operational view (BOV): | The view of the open-edi reference model which addresses the aspects of a) the semantics of business data in business transactions and associated data interchanges and b) the rules for business transactions including operational conventions, agreements, and mutual obligations which apply to the business needs of open-edi. |
| Business practices: | The major management and control systems operated by a business organization. |
| Business processes: | The means by which one or more activities are accomplished in operating business practices. |
| Class: | Any uniquely-identified abstraction of a set of logically-related instances that share the same or similar characteristics. |
| Class diagram: | Any diagram used to show the existence of classes, class categories, and their relationships. |
| Class hierarchy: | Any hierarchy of classes that results from a single inheritance. |

| | |
|--|---|
| Class inheritance: | Any inheritance among classes in which a new class (a.k.a. the <i>child</i>) is defined in terms of one or more existing classes (a.k.a., its <i>parents</i>), whereby the child inherits the features of its parents. |
| Class library: | Any collection of compatible, reusable classes. |
| Class variable: | 1. Any variable attribute of a class as a whole. 2. Any variable whose value is shared by all instances of a class. |
| Common business class: | A specialized business class, defined as required for a scenario, consisting of one or more foundation class(es) and methods and/or one or more core business class(es). |
| Complex object: | Any aggregate object containing one or more objects as component parts. |
| Core class (core business class): | An essential class of model objects that captures a key abstraction of the application domain. |
| Data model (IDEF1X): | 1. Any model of a collection of entities, operators, and consistency rules. 2. Describing information structures managed by an organization or system, the rules governing the management of the information, and logical relationships within the organization or system reflected in the information. |
| Encapsulation: | The physical localization of features (e.g., properties, behaviors) into a single blackbox abstraction that hides their implementation (and associated design decisions) behind a public interface. |
| Entity: | Any name used by a class text to refer to values that may at run-time become associated in some way to the instances of the class. In a class context, four kinds of entities may appear: a) attributes of a class; b) local entities of routines, including the redefined entity result for functions; c) formal routine arguments; and d) current, the predefined entity used to represent a reference to the current object. |
| Foundation class: | The combination of primitive classes (and other foundation classes) to build fundamental, reusable classes, referred to as attributes in a class diagram for a scenario. |
| Framework: | Any set of prebuilt classes and methods that define the basic structure of some end user functions, leaving the application-specific details to be filled in by developers. |
| Function: | 1. Any input/output mapping resulting from some object's behavior. 2. Any operation that maps an object of one set into a set of objects in the same or different set. |
| Functional service view (FSV): | The view of the open-edi reference model that addresses the supporting services meeting the mechanistic needs of open-edi. It focuses on the Information Technology aspects of functional capabilities, service interfaces and protocols. |
| Information parcel: | A small, complete piece or packet of data. |
| Inheritance: | The incremental construction of a new definition in terms of existing definitions without disturbing the original definitions and their clients. |
| Instance: | Anything created from or corresponding to a definition. |
| Instance [of a given class]: | Any object instantiated according to the definition provided by the given class. |

| | |
|---|--|
| Instance variable: | Any attribute of an individual instance. |
| Interface: | Any specification of the boundary of something in terms of the possible interactions or properties that are visible across that boundary. |
| Method: | A discrete activity, action, or behavior that (1) implements a functional (i.e., sequential) or process (i.e., concurrent) abstraction, (2) is performed by, belongs to, and is encapsulated in an object or class, and (3) usually implements a service requested by a message. |
| Model(ling): | 1. Any abstraction that includes all essential capabilities, properties, or aspects of what is being modeled without any extraneous details. 2. Any cohesive set of requirements or design information. |
| Object: | Any real or abstract thing about which we store data and the operations to manipulate those data. |
| Object class: | See Class. |
| Object instance: | See instance and instance [of a class]. |
| Object interface: | Any listing of the operations and attributes that any object provides. This includes the signatures of the operations, and the types of the attributes. |
| Object state: | Any status, situation, condition, mode, or life-cycle phase of an object during which certain rules of overall behavior apply. |
| Object state transition network: (IDEF3) | Description of all allowable transitions an object may undergo throughout a particular process. |
| Open-edi | The electronic processing of business transactions among autonomous multiple organizations within and across sectors. |
| Open-edi reference model: | A model developed to provide standards required for the inter-working of organizations through interconnected information technology systems. The model is independent of specific information technology implementations, business content or conventions, business activities, and organizations. |
| Operation: | Any discrete activity, action, or behavior that is performed by an object or class. |
| Package: | Any program unit that allows the specification of groups of logically related entities. |
| Polymorphism: | The ability of a single name to refer to different things having different forms. |
| Primitive class: | A class that is provided by the environment, most typically by the programming language. |
| Process model (IDEF3): | A static architecture model of an application in terms of processes, the objects they contain, and the processors they are allocated to. |
| Public interface: | Describing an interface whose features are visible to both external clients and descendants. |
| Scenario: | Any single, specific, contiguous set of object interactions (e.g., instantiations, message sends, operation executions, exception raising and handling, and/or object destructions) that captures a single functional abstraction that cannot be encapsulated as an operation within a single object or class. |
| State transition arc: | A technique for representing the transition of an object from one state to another. |

REFERENCE GUIDE "THE NEXT GENERATION OF UN/EDIFACT"

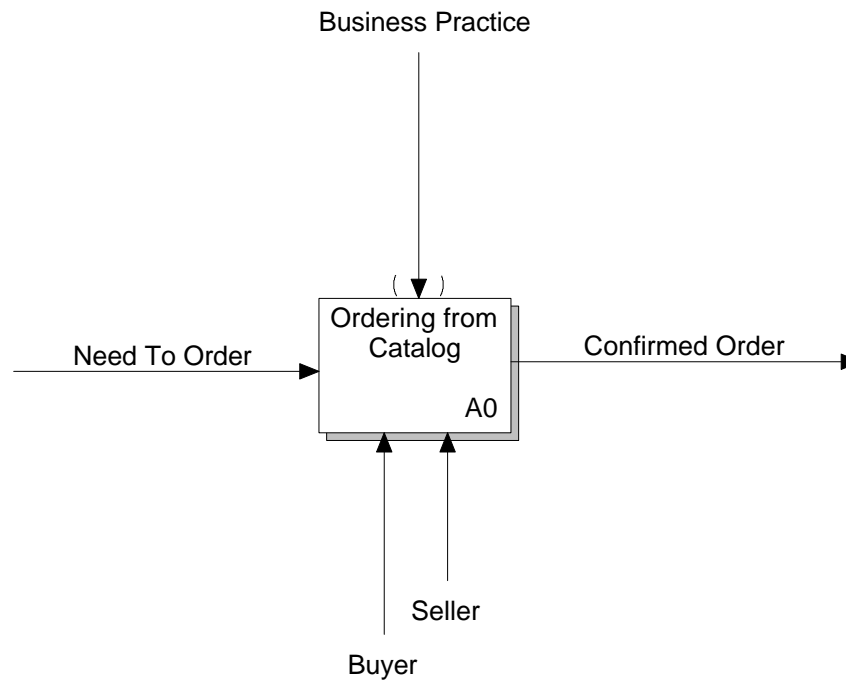
| | |
|--------------------------|--|
| Subclass: | any class that is derived, either directly or indirectly, from the given class via inheritance. |
| Superclass: | Any class from which the given class is derived, either directly or indirectly, via inheritance. |
| Unit of behavior: | A generic packet of information that represents activities, actions, processes, and operations of the real-world tied together with constraint links to reflect precedence, user-defined relations, or object flow. An elaboration document provides a full understanding of a process flow description for each Unit Of Behavior (UOB). |
| Variable: | Anything, the value(s) of which can change. |

6 ACRONYMS

| | |
|--------------------------|---|
| AC.1 | Research, strategic advice and implementation planning group |
| ASC X12 | Accredited standards committee X12 |
| CASE | Computer aided software engineering |
| CALS | Commerce at light speed |
| CEFACT | Centre for facilitation of procedures and practices in administration, commerce and transport |
| EBES | European board for EDI standards |
| EDI | Electronic data interchange |
| EDIFACT | Electronic data interchange for administration, commerce and transport (See UN/EDIFACT) |
| ESG | EDIFACT steering group |
| EWG | EDIFACT working group |
| ICAM | Integrated computer aided manufacturing |
| ICOM | Input, control, output, mechanism |
| ID | Identification |
| IDEF | Integrated definition language |
| IEC | International electrotechnical commission |
| IEEE | International electric and electronic engineering society |
| ISO | International organization for standardization |
| JRT | Joint rapporteur team |
| IEC JTC1/SC30/WG1 | IEC Joint technical committee 1/subcommittee 30/Working group 1 |
| OOM | Object oriented modeling |
| OOT | Object oriented technology |
| OSTN | Object state transition network |
| PFD | Process flow description |
| SITG | Strategic implementation task group |
| SME | Small and medium sized enterprise |
| UN/EDIFACT | United nations electronic data interchange for administration, commerce and transport |
| UOB | Unit of behavior |
| UNTDID | United nations trade data interchange directory |
| UNTDDED | United nations trade data elements directory |

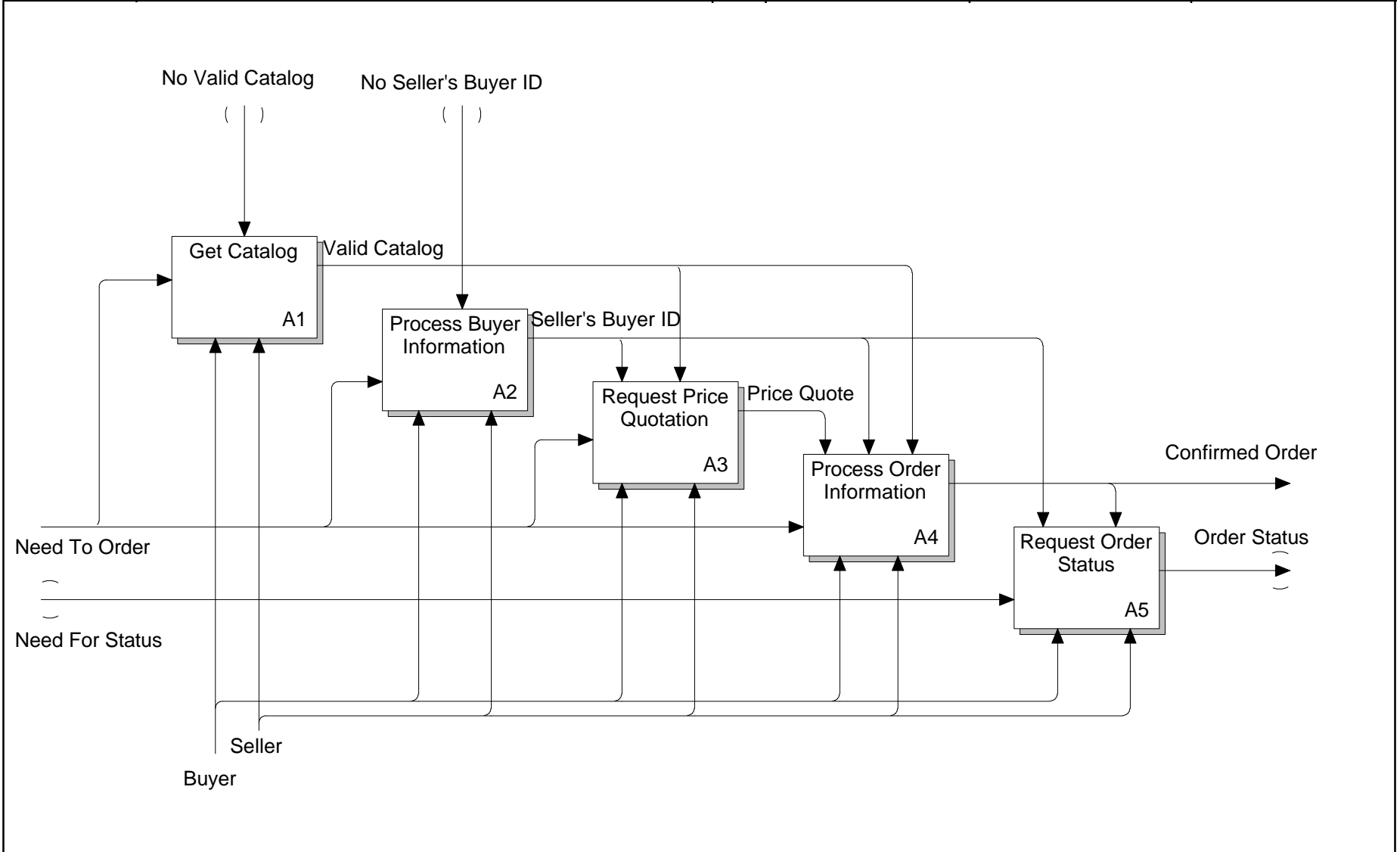
APPENDIX A – CATALOG ORDER FRAMEWORK

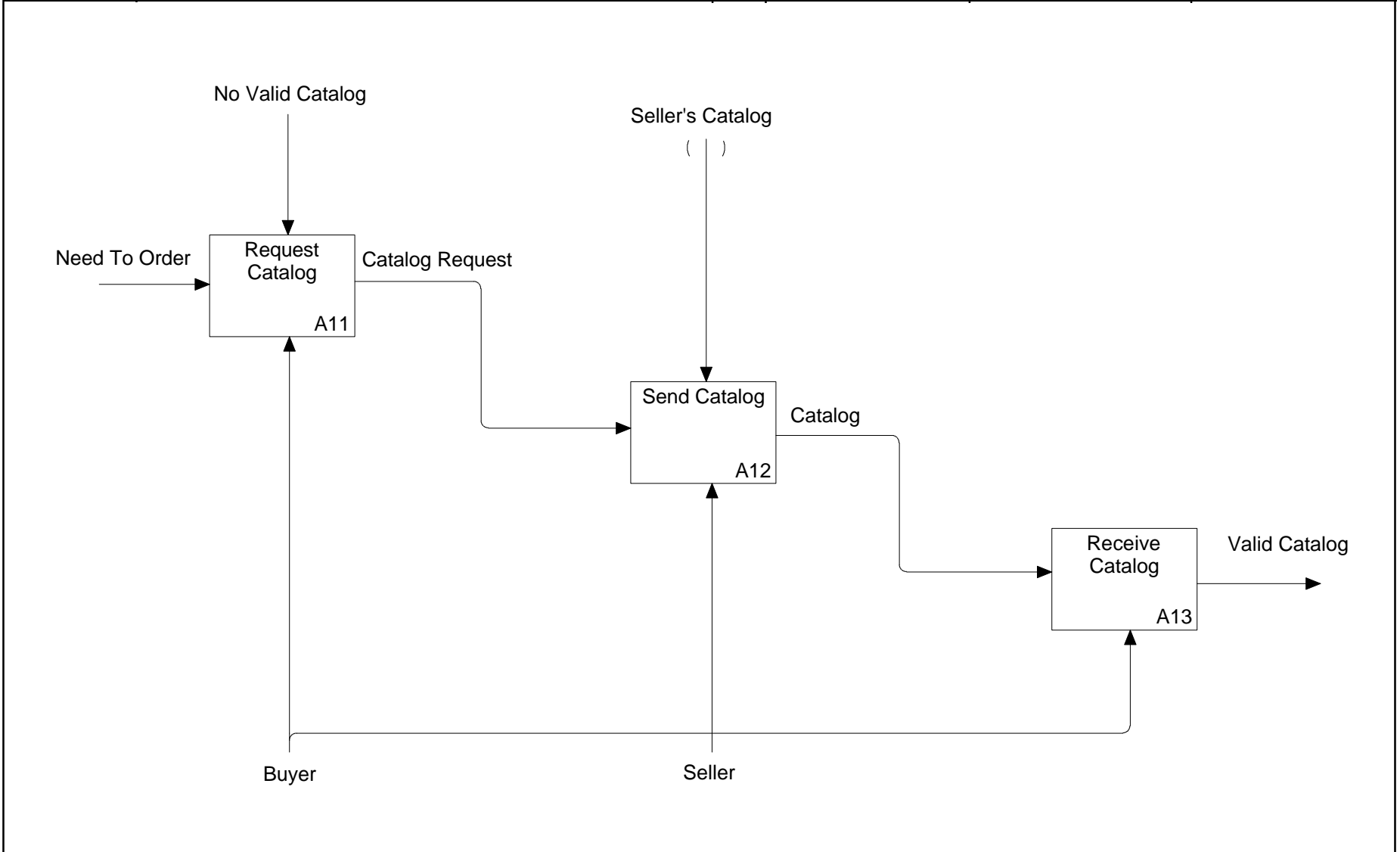
| | | | | | | |
|----------|---|----------------------|---|-------------|---------|-------------------------|
| Used at: | Author: <i>Klaus</i> | Date: <i>7/31/97</i> | | WORKING | READER: | Context: <i>NONE</i> |
| | Project: <i>Catalog Order Framework</i> | | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | | RECOMMENDED | | |
| | | | X | PUBLICATION | | |
| | | | | | DATE: | |



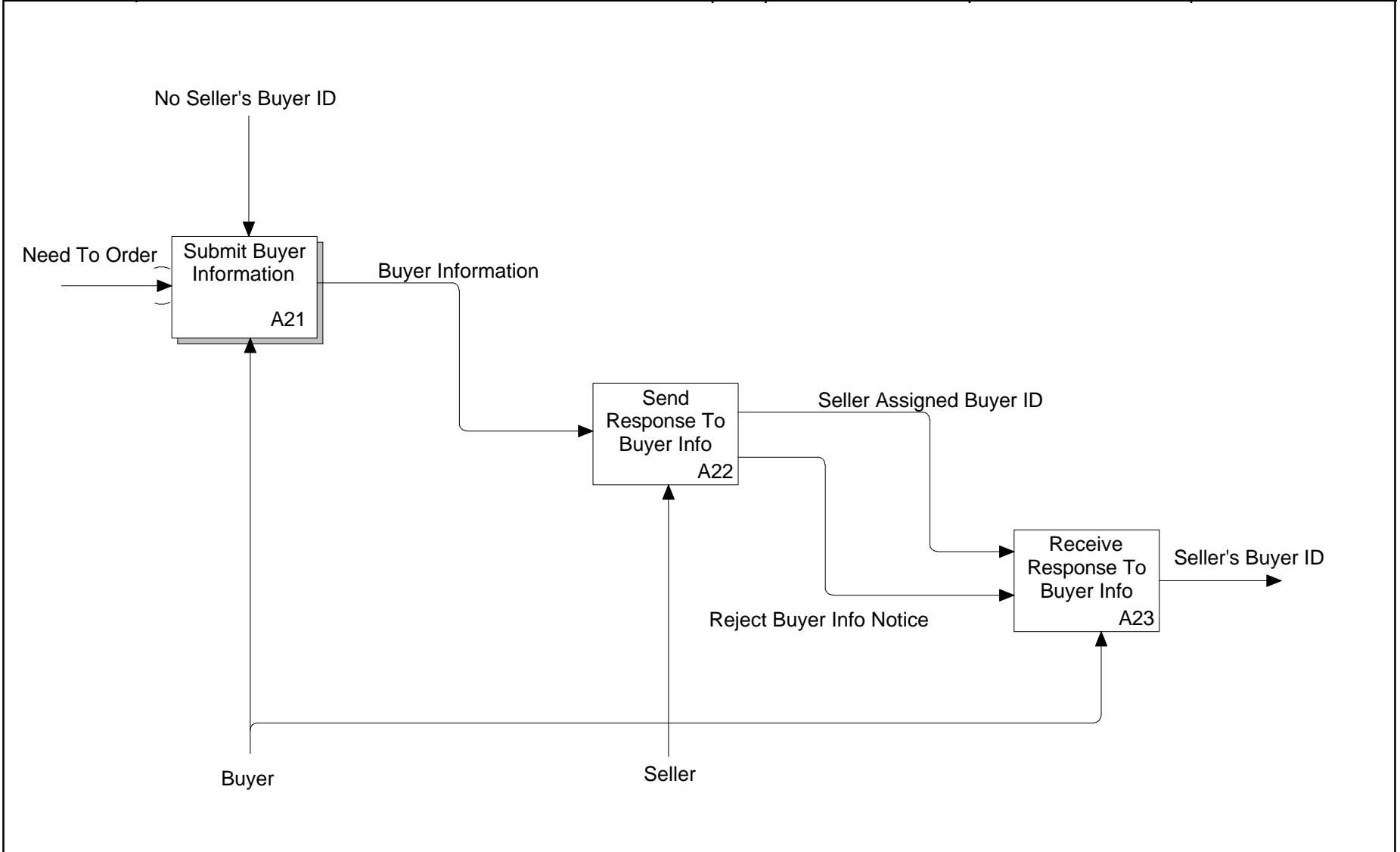
| | | | |
|------------------|---------------------------------------|---------|---------|
| Node: <i>A-0</i> | Title: <i>Catalog Order Framework</i> | Number: | Page: 1 |
|------------------|---------------------------------------|---------|---------|

| | | | | | |
|----------|-----------------------------|---------------|---------------|---------|------------------------|
| Used at: | Author: Klaus | Date: 7/31/97 | WORKING | READER: | Context: TOP A-0 |
| | Project: | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | RECOMMENDED | | |
| | | | X PUBLICATION | | |
| | | | DATE: | | |

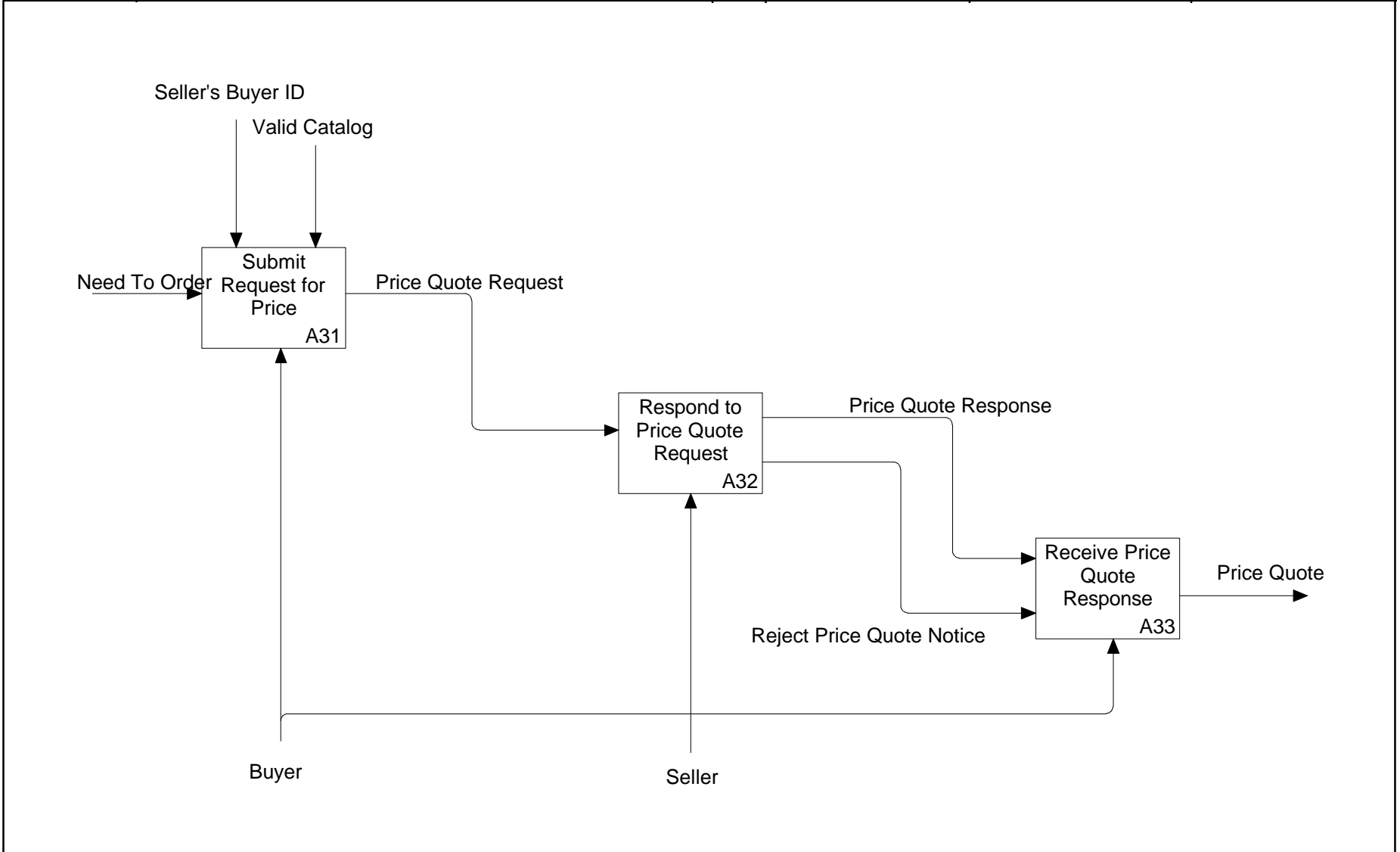




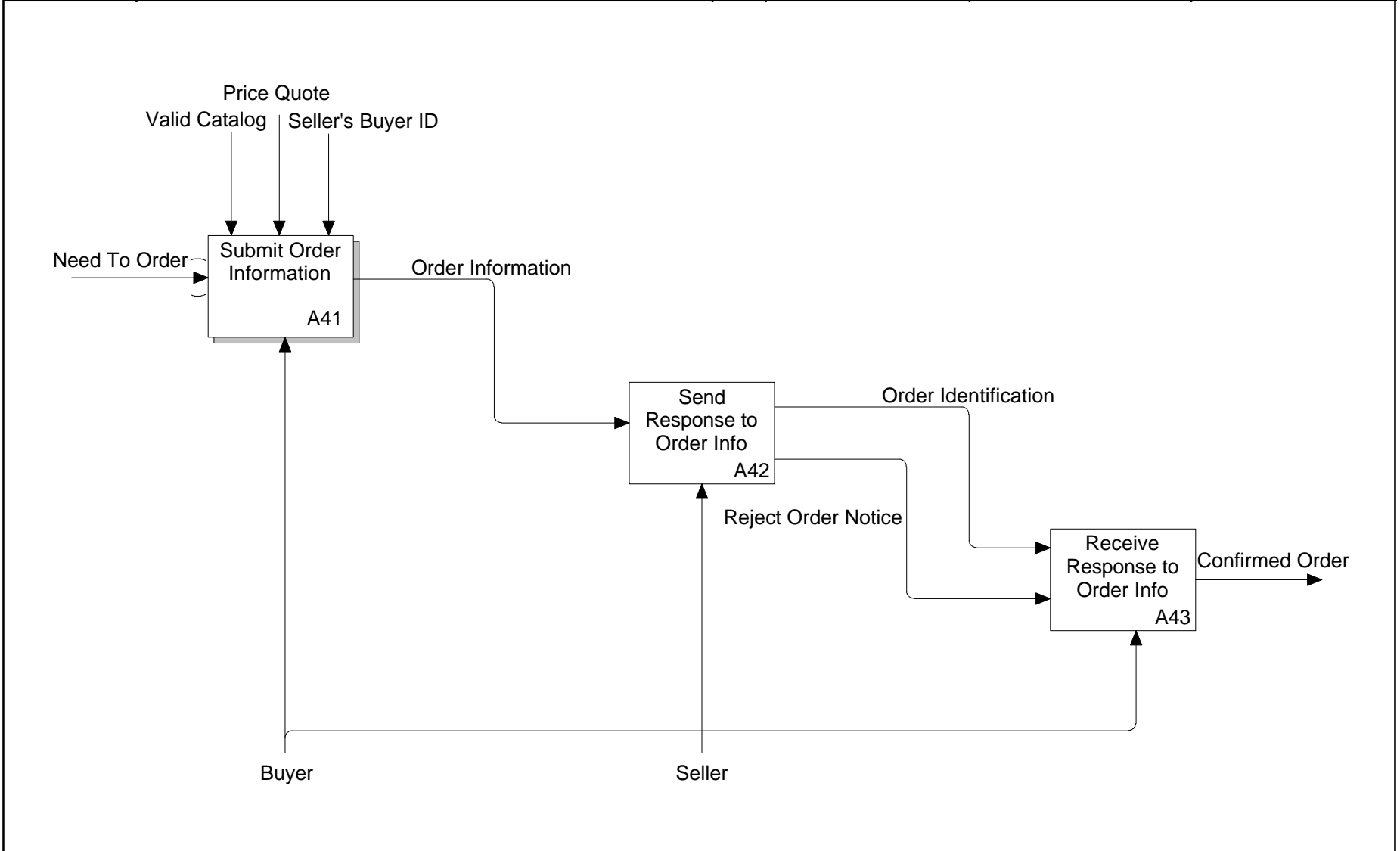
| | | | | | |
|----------|-----------------------------|----------------------|---------------|---------|---|
| Used at: | Author: <i>Klaus</i> | Date: <i>7/31/97</i> | WORKING | READER: | Context: <input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> A0 |
| | Project: | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | RECOMMENDED | DATE: | |
| | | | X PUBLICATION | | |

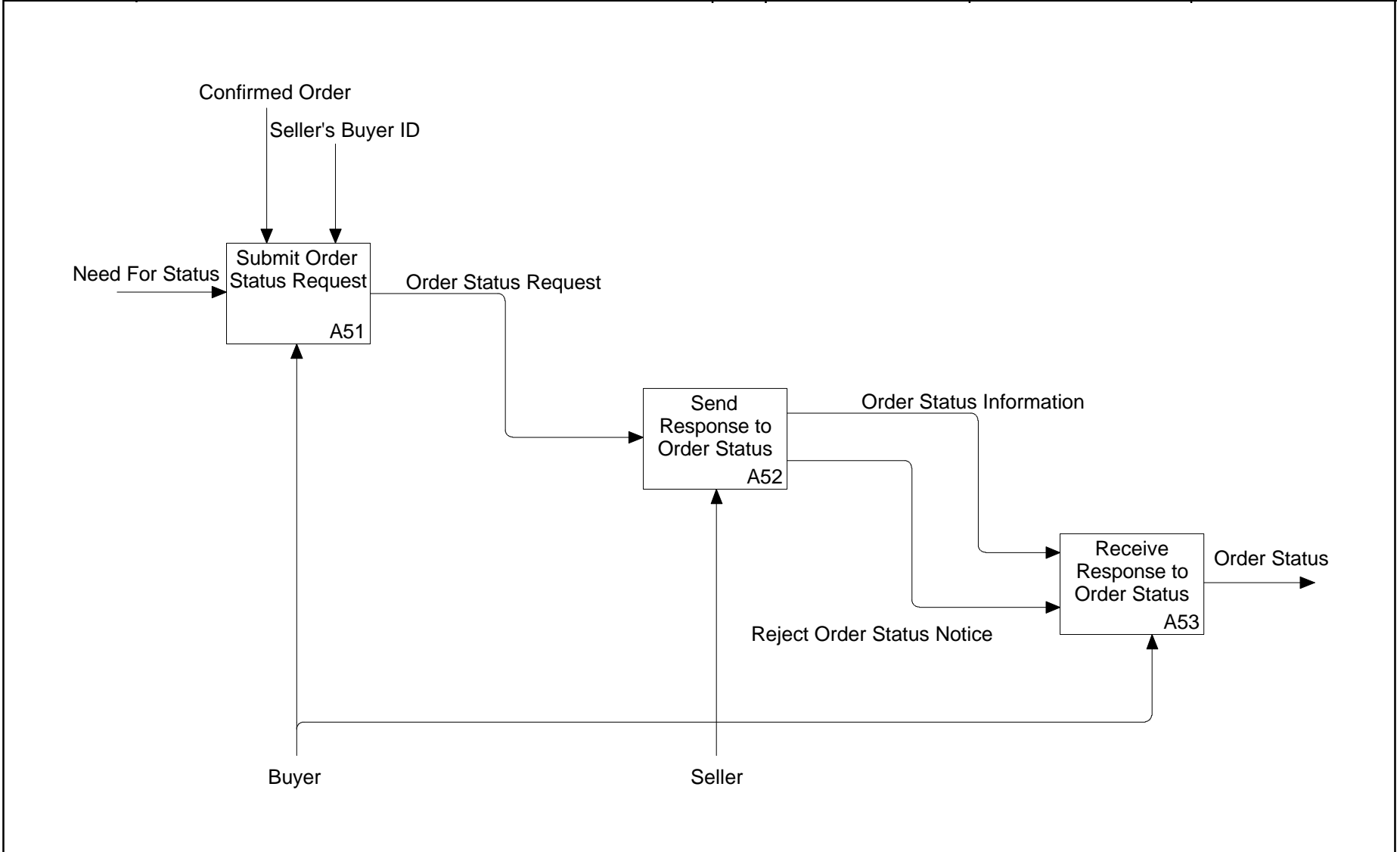


| | | | | | |
|----------|-----------------------------|---------------|---------------|---------|---|
| Used at: | Author: Klaus | Date: 7/31/97 | WORKING | READER: | Context: <input type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> A0 |
| | Project: | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | RECOMMENDED | | |
| | | | X PUBLICATION | | |
| | | | DATE: | | |

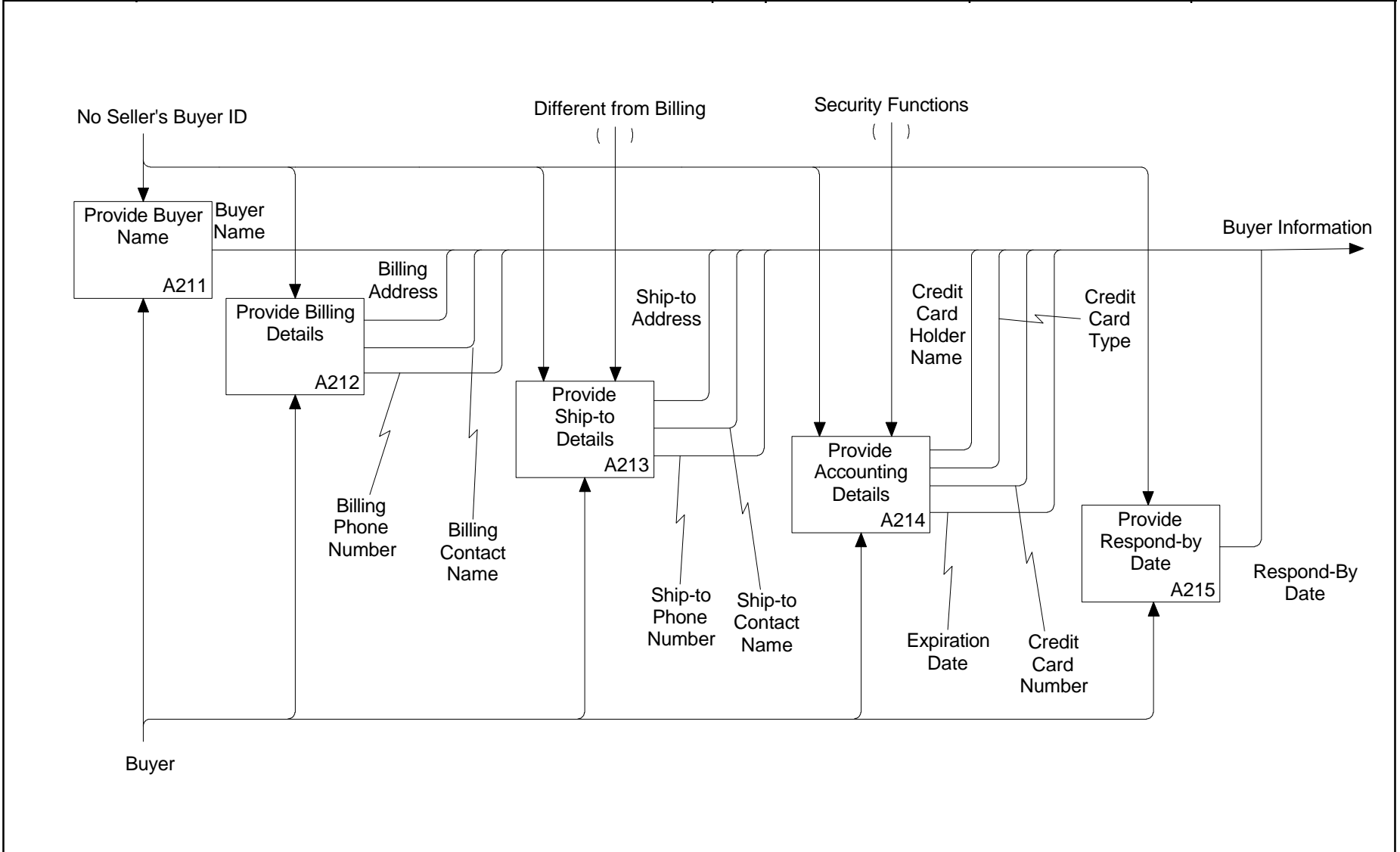


| | | | | | |
|----------|-----------------------------|----------------------|---------------|---------|---|
| Used at: | Author: <i>Klaus</i> | Date: <i>7/31/97</i> | WORKING | READER: | Context: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> A0 |
| | Project: | | DRAFT | DATE: | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | RECOMMENDED | | |
| | | | X PUBLICATION | | |

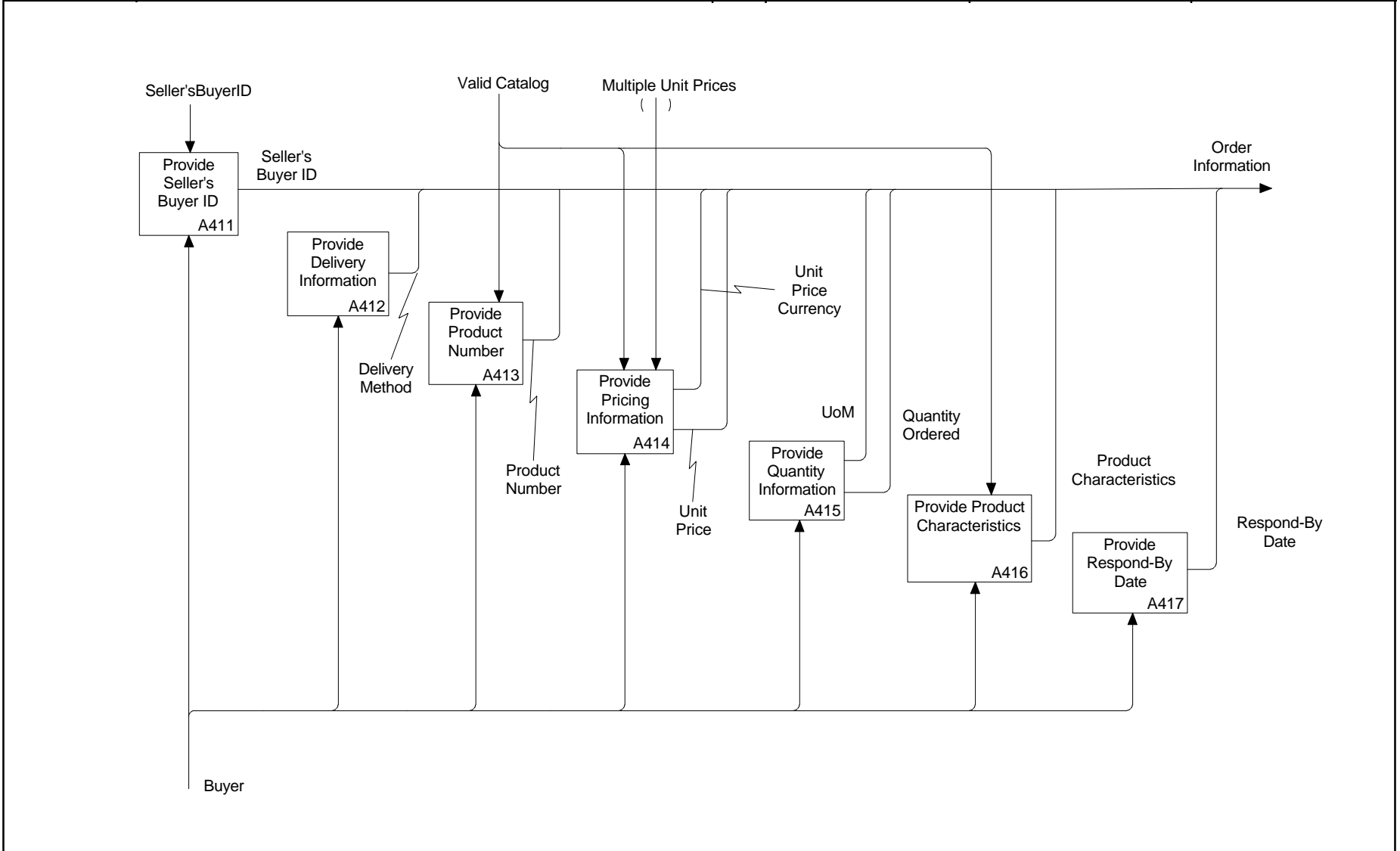




| | | | | | |
|----------|-----------------------------|---------------|---------------|---------|---|
| Used at: | Author: Klaus | Date: 7/31/97 | WORKING | READER: | Context: <input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> A2 |
| | Project: | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | RECOMMENDED | DATE: | |
| | | | X PUBLICATION | | |



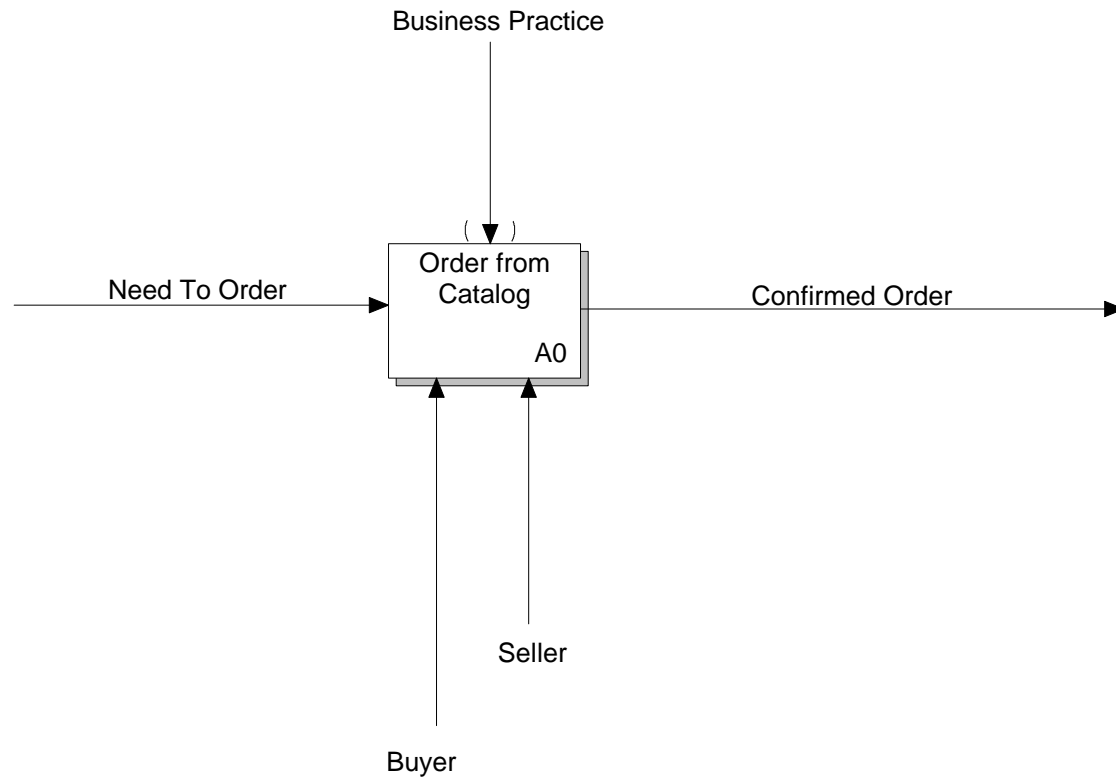
| | | | | | |
|----------|-----------------------------|---------------|---------------|---------|---|
| Used at: | Author: Klaus | Date: 7/31/97 | WORKING | READER: | Context: <input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> A4 |
| | Project: | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | RECOMMENDED | | |
| | | | X PUBLICATION | | |
| | | | DATE: | | |



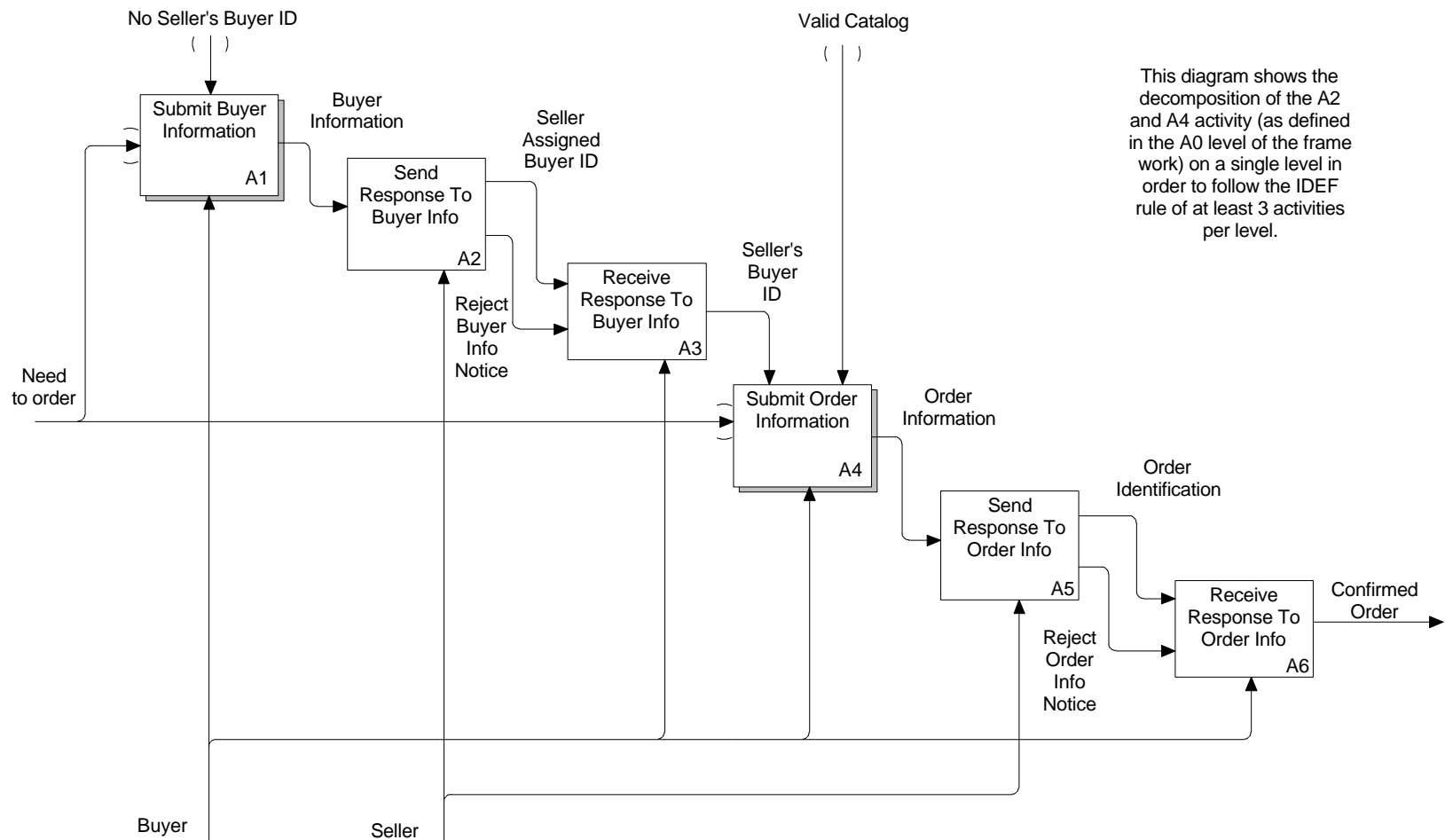
APPENDIX B – CATALOG ORDER SCENARIO EXAMPLE

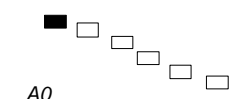
IDEF0 ACTIVITY MODEL (DIAGRAM PAGES 1-4)

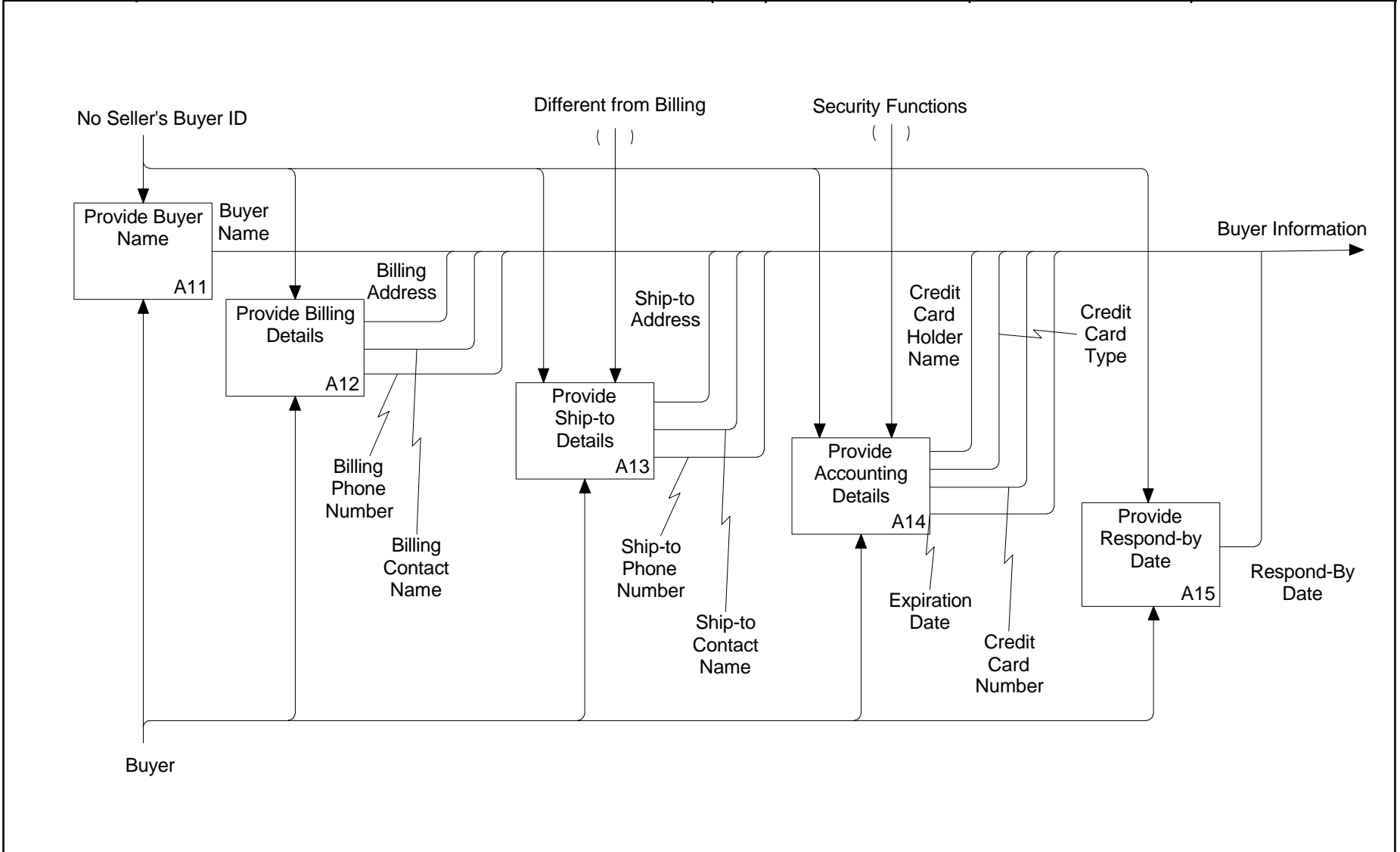
| | | | | | | |
|----------|-----------------------------|----------------------|---|-------------|---------|-------------------------|
| Used at: | Author: <i>Klaus</i> | Date: <i>7/30/97</i> | | WORKING | READER: | Context: <i>NONE</i> |
| | Project: | | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | | RECOMMENDED | | |
| | | | X | PUBLICATION | | |
| | | | | | DATE: | |



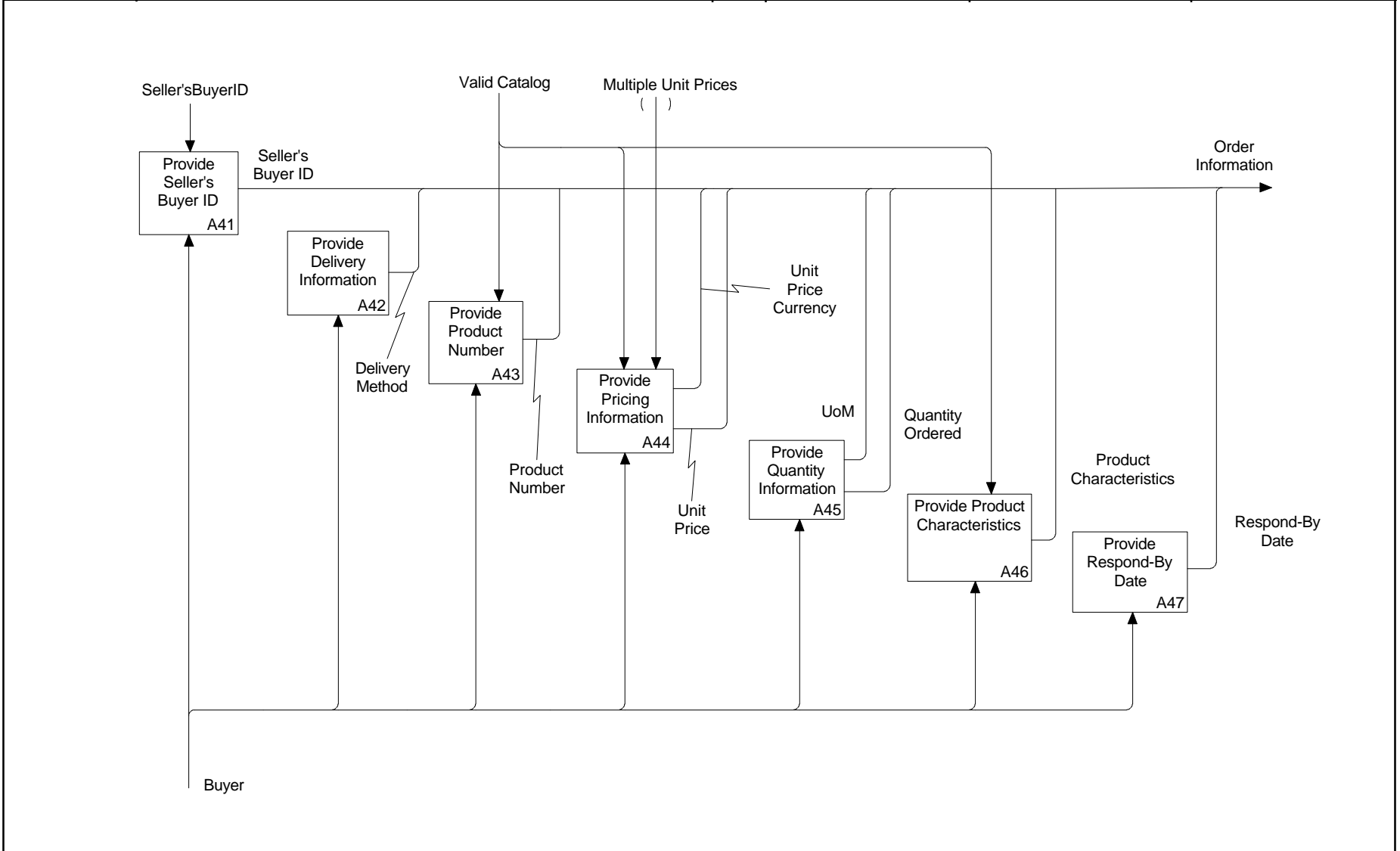
| | | | | | |
|----------|-----------------------------|---------------|---------------|---------|------------------------|
| Used at: | Author: Klaus | Date: 7/29/97 | WORKING | READER: | Context: TOP A-0 |
| | Project: | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | RECOMMENDED | | |
| | | | X PUBLICATION | | |
| | | | DATE: | | |



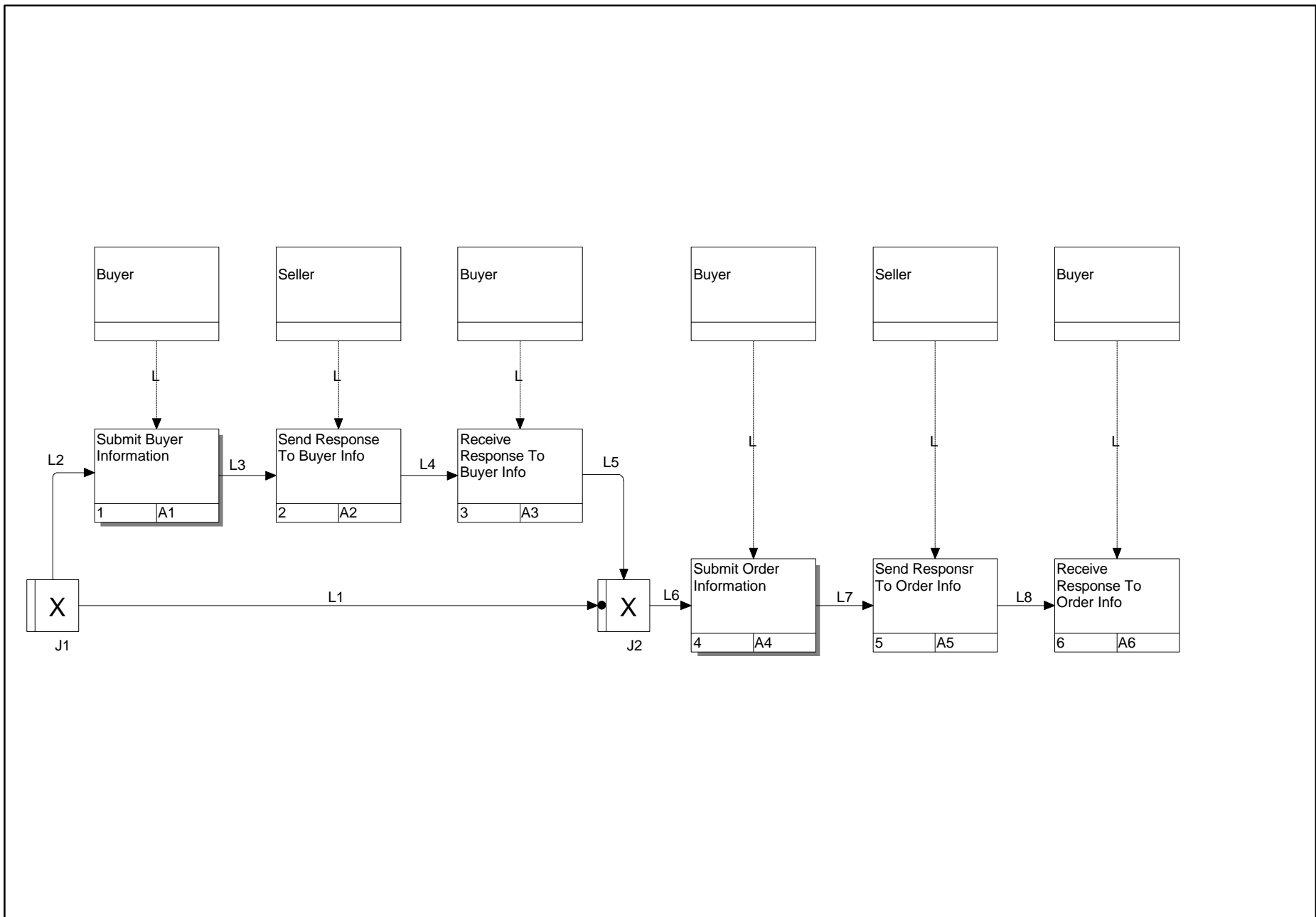
| | | | | | |
|----------|-----------------------------|---------------|---------------|---------|---|
| Used at: | Author: Klaus | Date: 7/29/97 | WORKING | READER: | Context:  |
| | Project: | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | RECOMMENDED | | |
| | | | X PUBLICATION | | |
| | | | | DATE: | |



| | | | | | |
|----------|-----------------------------|---------------|---------------|---------|--|
| Used at: | Author: Klaus | Date: 7/29/97 | WORKING | READER: | Context: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> A0 |
| | Project: | | DRAFT | | |
| | Notes: 1 2 3 4 5 6 7 8 9 10 | Rev: 2 | RECOMMENDED | | |
| | | | X PUBLICATION | | |
| | | | DATE: | | |

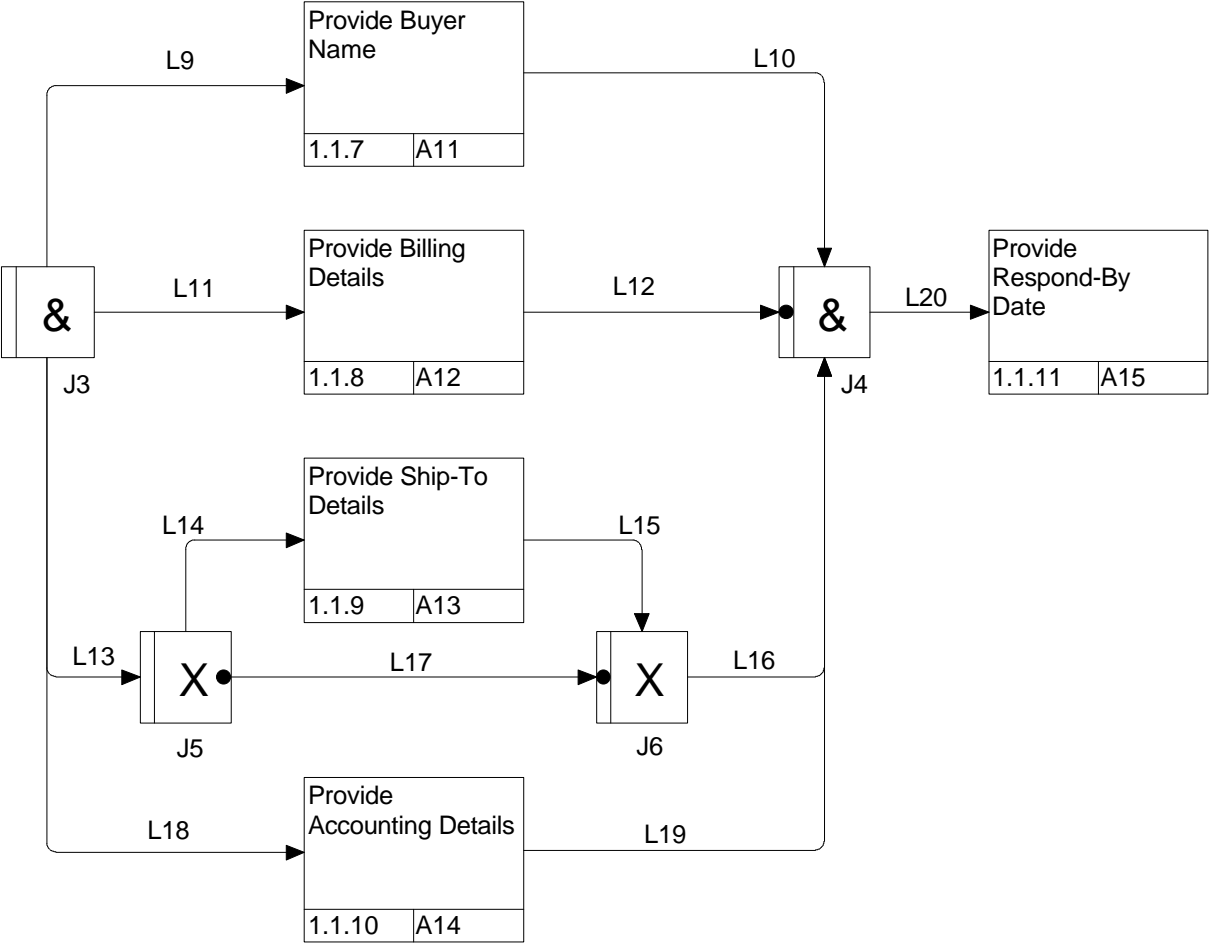


IDEF3 PROCESS FLOW MODEL (DIAGRAM PAGES 5-7)



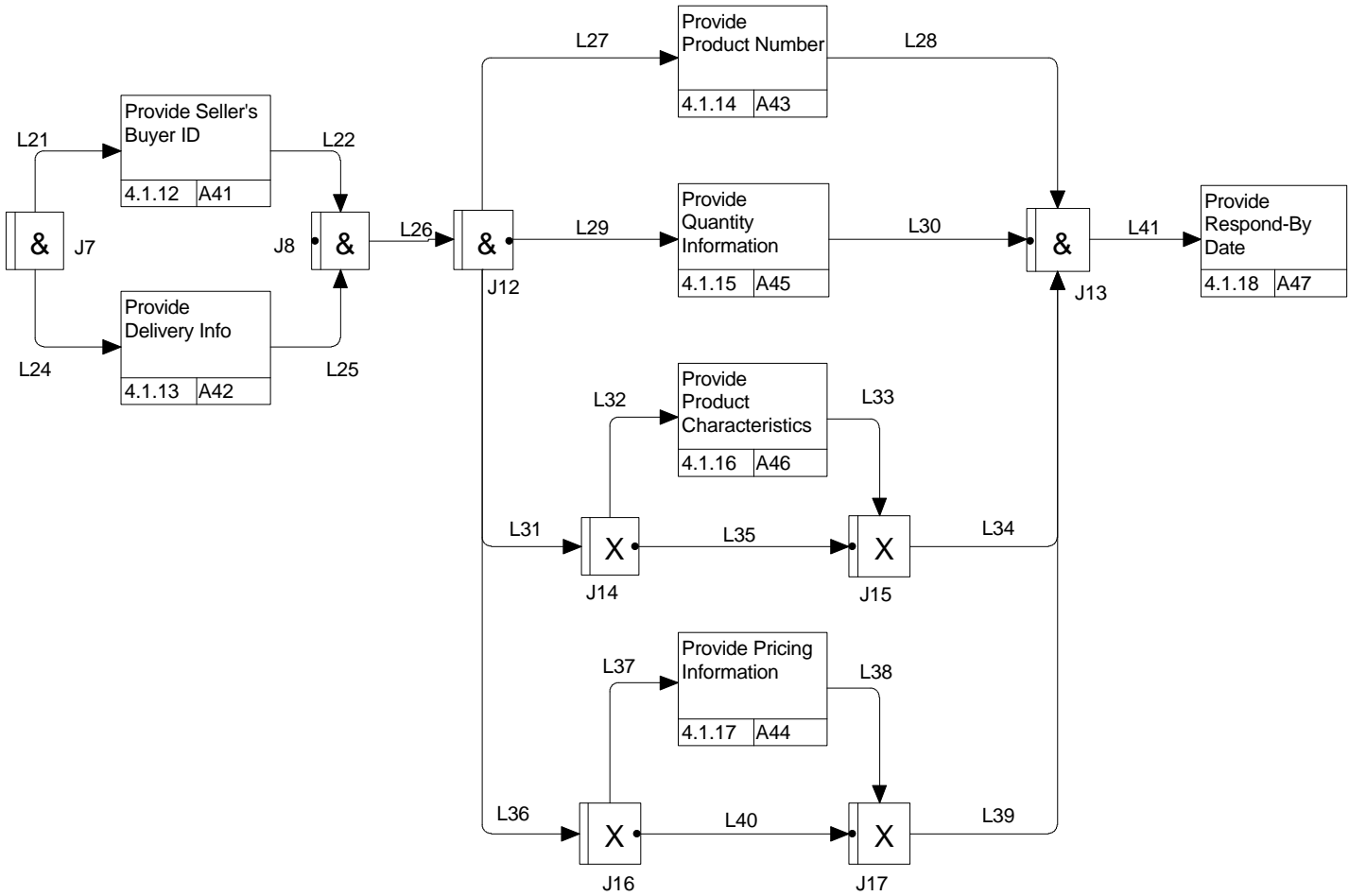
| | | | |
|--|---|--|---|
| SCENARIO: "Catalog Order Example" OBJECT: | DESCRIPTION NAME: Order from Catalog | NUMBER: <table border="1" style="float: right; margin-top: 10px;"> <tr> <td style="text-align: center;">5</td> </tr> </table> | 5 |
| 5 | | | |

| | | | | | |
|----------|-----------------------------|---------------|-------------|-----------|-------|
| USED AT: | ANALYST: | DATE: 7/29/97 | WORKING | REVIEWER: | VIEW: |
| | PROJECT: | | DRAFT | | |
| | NOTES: 1 2 3 4 5 6 7 8 9 10 | REV: 2 | RECOMMENDED | | |
| | | | X RELEASED | | |
| | | | DATE: | | |



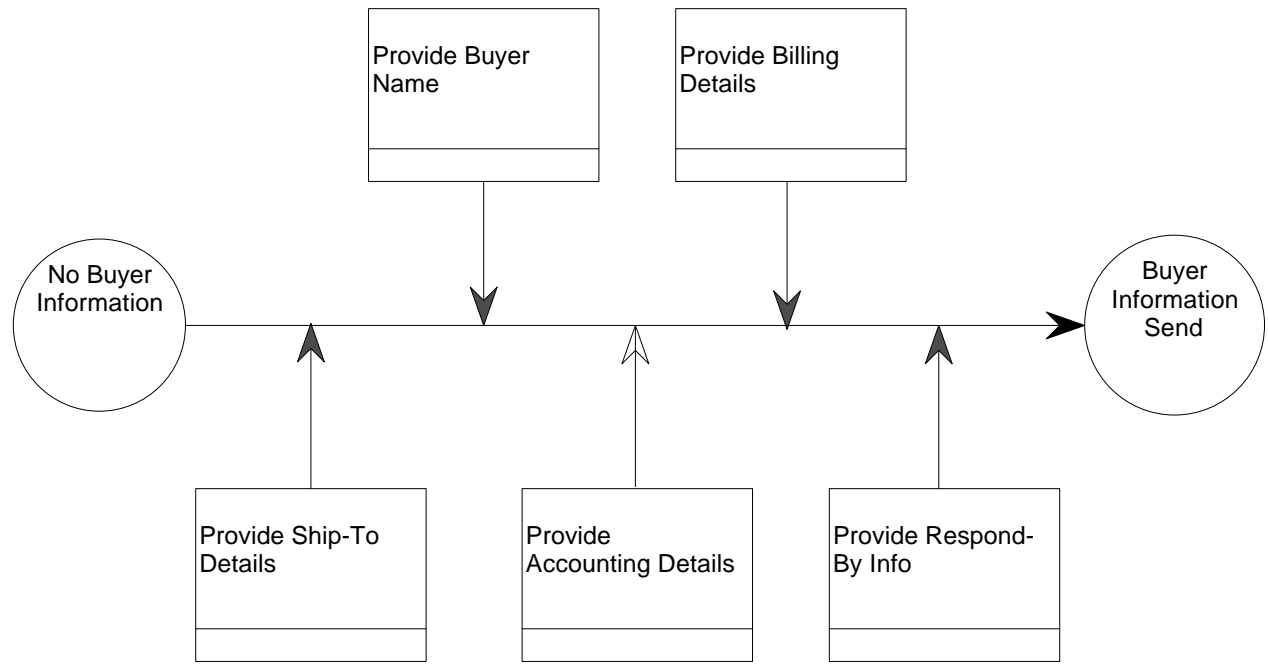
| | | | |
|-----------------------------------|--|---------|---|
| SCENARIO: "Catalog Order Example" | DESCRIPTION NAME: Submit Buyer Information | NUMBER: | 6 |
| OBJECT: | | | |

| | | | | | |
|----------|-----------------------------|---------------|-------------|-----------|-------|
| USED AT: | ANALYST: | DATE: 7/29/97 | WORKING | REVIEWER: | VIEW: |
| | PROJECT: | | DRAFT | | |
| | NOTES: 1 2 3 4 5 6 7 8 9 10 | REV: 2 | RECOMMENDED | | |
| | | | X RELEASED | | |
| | | | DATE: | | |



| | | | |
|-----------------------------------|---|---------|---|
| SCENARIO: "Catalog Order Example" | DESCRIPTION NAME: <i>Submit Order Information</i> | NUMBER: | 7 |
| OBJECT: | | | |

IDEF3 OBJECT STATE TRANSITION NETWORK (DIAGRAM PAGES 8-11)

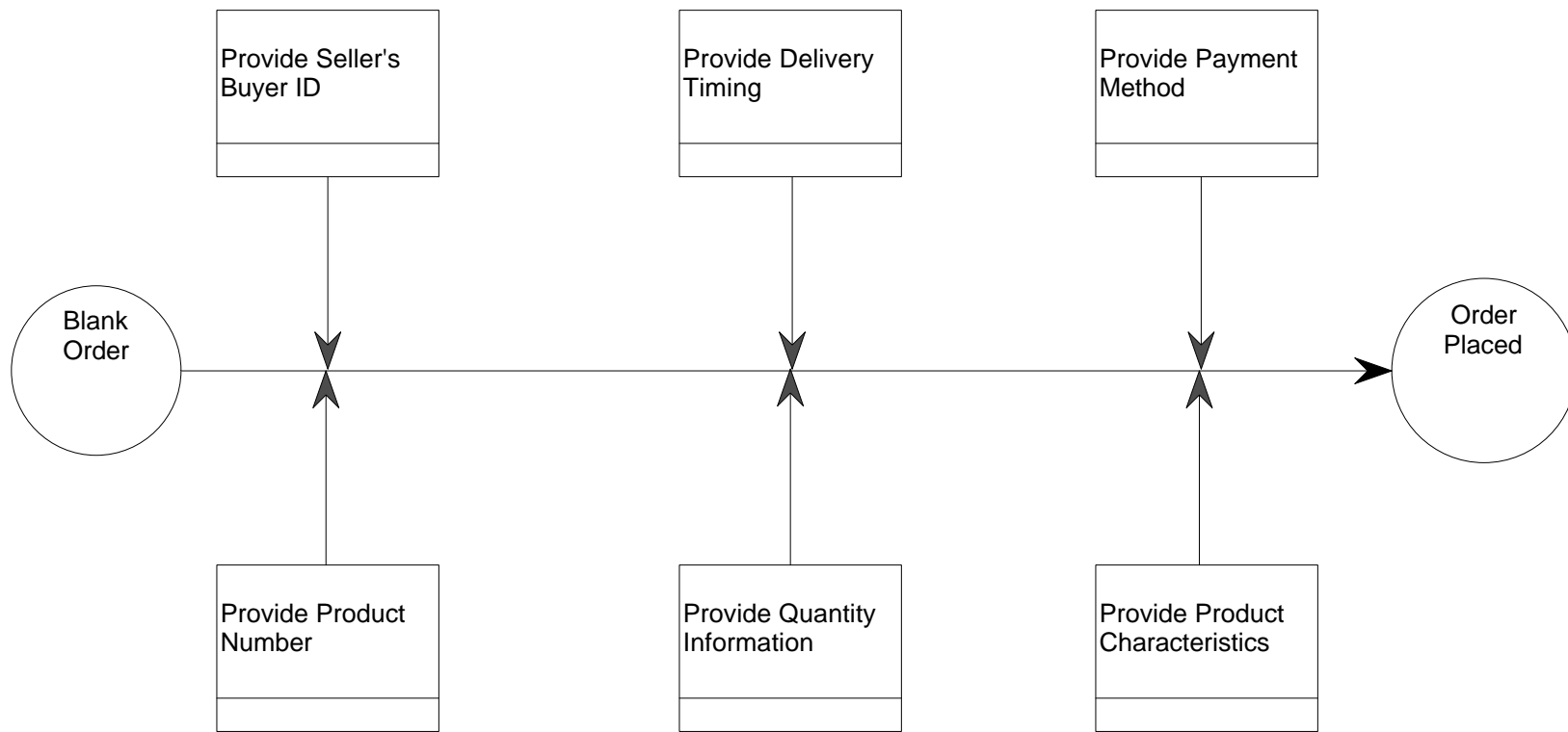


SCENARIO: "Catalog Order"
OBJECT: "Buyer Information"

DESCRIPTION NAME: *Submit Buyer Information*

NUMBER:

8

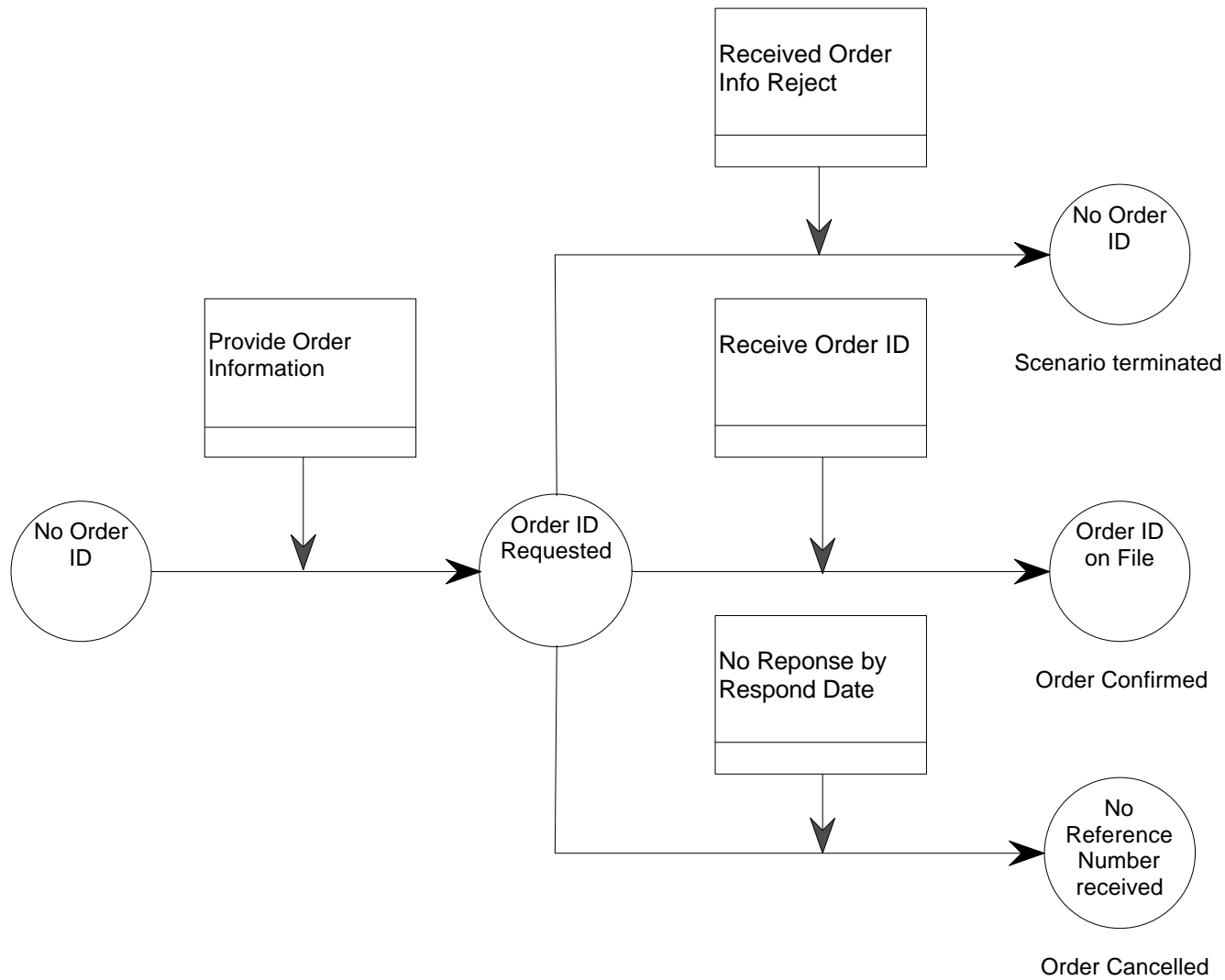


SCENARIO: "Catalog Order"
OBJECT: "Order Information"

DESCRIPTION NAME: *Submit Order Information*

NUMBER:

9



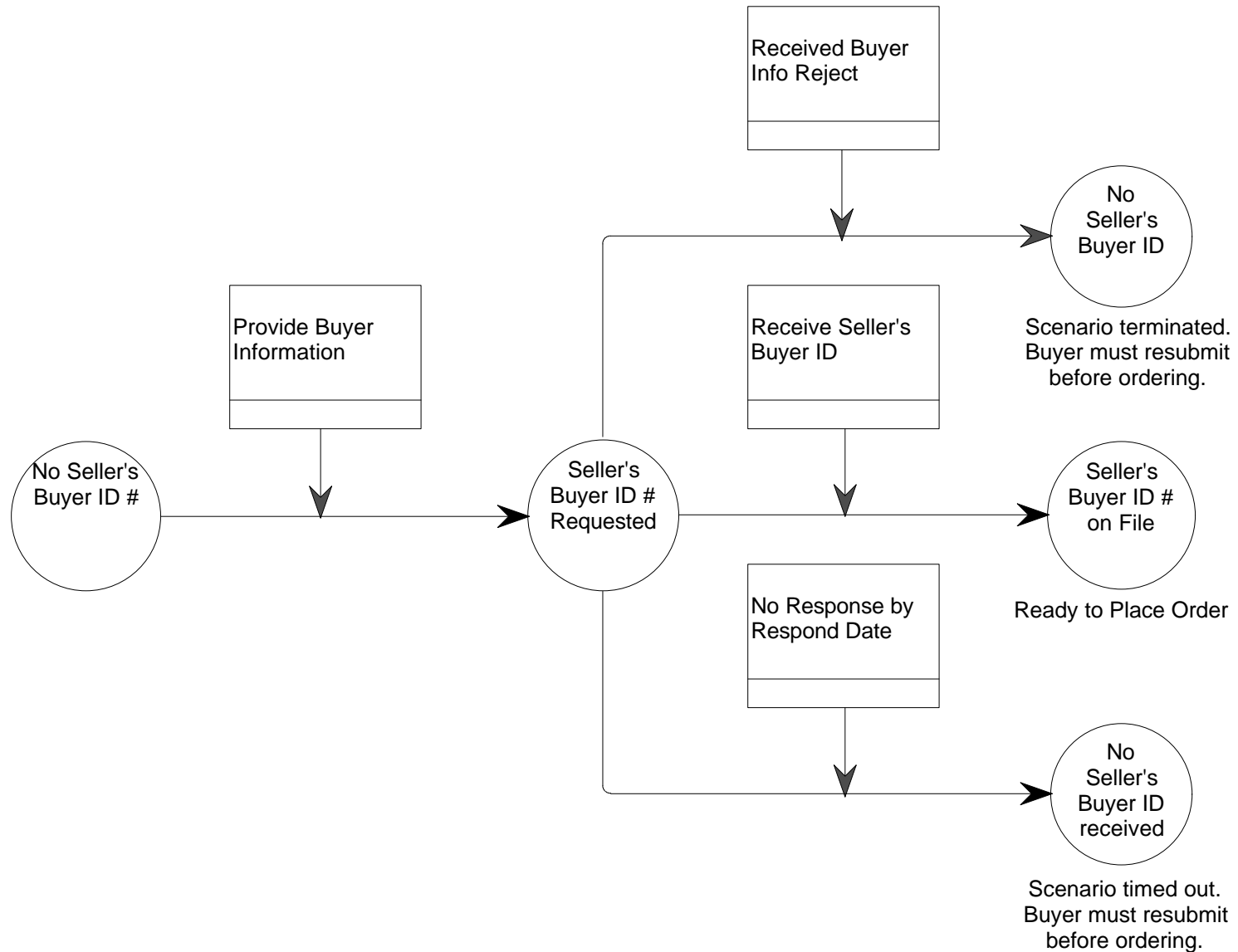
SCENARIO: "Catalog Order"

OBJECT: "Order Reference Number"

DESCRIPTION NAME: *Receive Order Reference Number*

NUMBER:

10



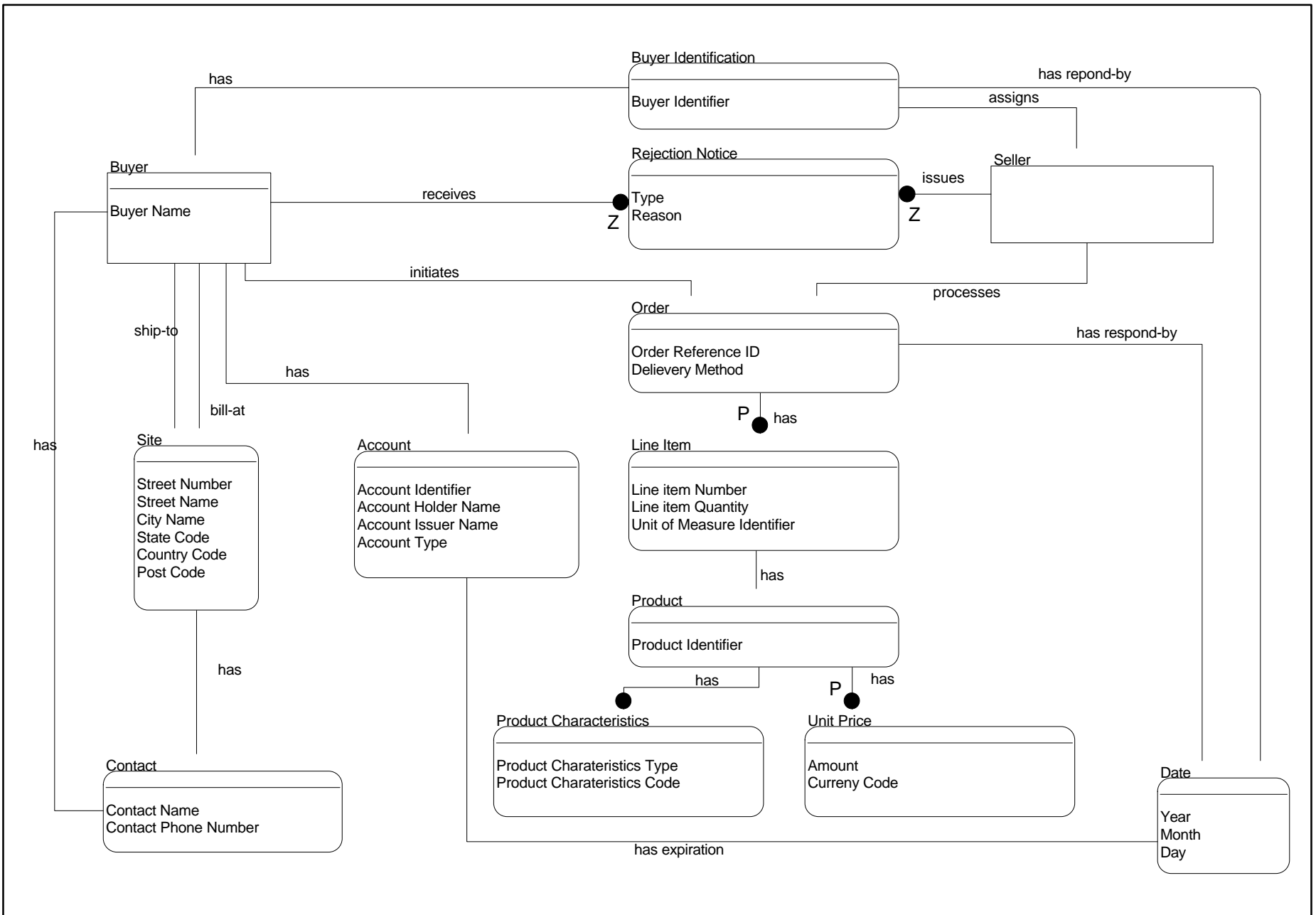
SCENARIO: "Catalog Order"

DESCRIPTION NAME: Receive Seller's Buyer ID

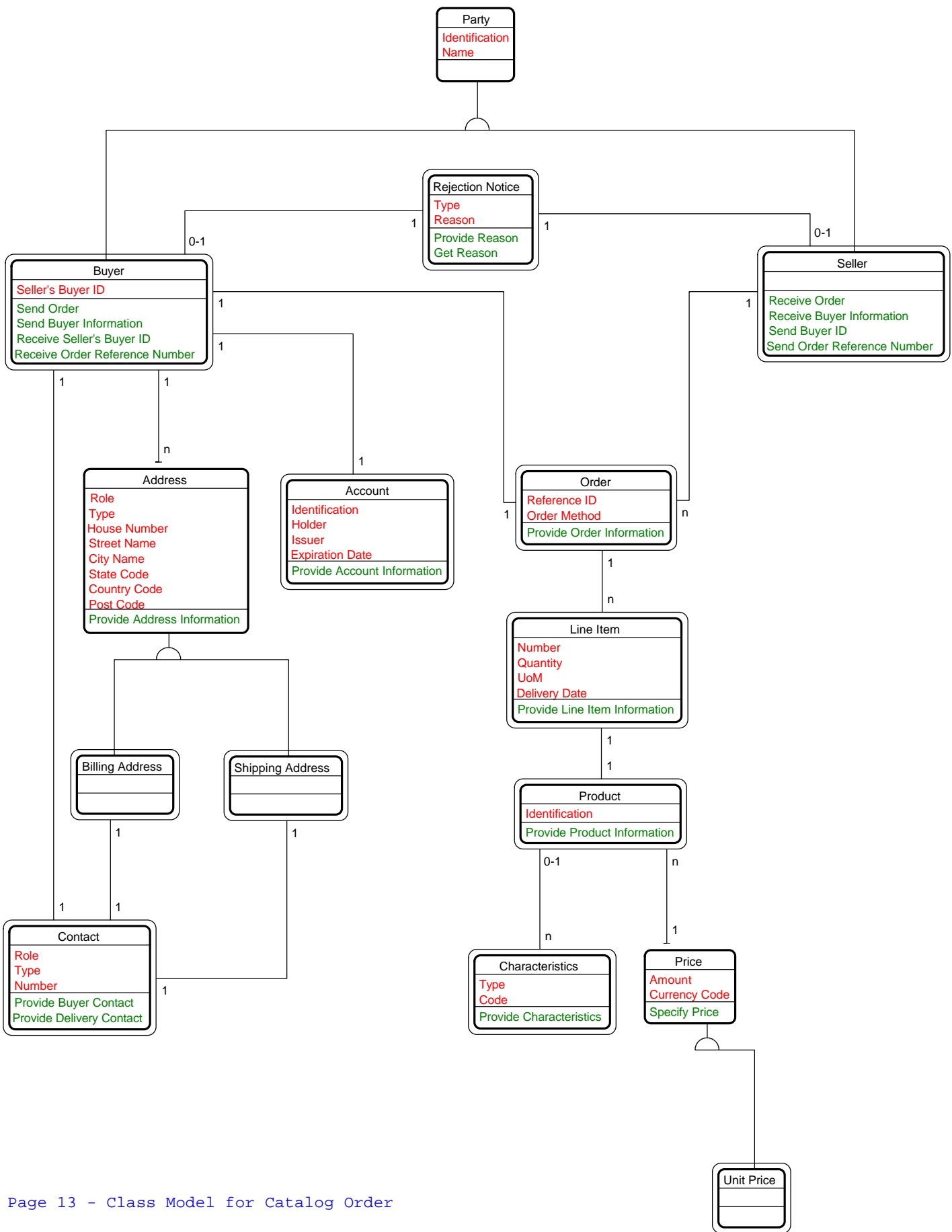
NUMBER:

OBJECT: "Seller's Buyer ID"

IDEF1X DATA MODEL (DIAGRAM PAGE 12)



OBJECT CLASS MODEL (DIAGRAM PAGE 13)



APPENDIX C – EDI CLASS LIBRARIES

PRIMITIVE CLASSES

| Class Name |
|------------------|
| Integer |
| Float (IEEE 754) |
| Character |
| Boolean |

FOUNDATION CLASSES

| Class Name |
|-------------------|
| String |
| Code List |
| Code Entry |
| Identifier List |
| Identifier |
| Name |
| Description |
| Date |
| Quantity |
| Unit of Measure |
| Sequential Number |

CORE BUSINESS CLASSES

| Class Name |
|------------------|
| Party |
| Address |
| Contact |
| Account |
| Rejection Notice |
| Order |
| Line Item |
| Product |
| Characteristics |
| Price |

COMMON BUSINESS CLASSES

| |
|------------------|
| Class Name |
| Buyer |
| Seller |
| Shipping Address |
| Billing Address |

APPENDIX D – AC.1’S MEMBERSHIP LIST

REFERENCE GUIDE "THE NEXT GENERATION OF UN/EDIFACT"

| | | |
|---------------------|--|--|
| David Dobbing | ddobbing@attmail.com +61 2 555 9914 +61 2 810 7860 fax | Principal Consultant Data Logistics Pty. Ltd. 505 Darling St. Balmain, Sydney NSW 2041 Australia |
| Harry Featherstone | hfeather@lmi.org +1 703 917 7210 | Research Fellow Logistics Management Institute 2000 Corporate Ridge McLean, VA 22102-7805 U.S.A. |
| Paul R. Levine | plevine@notes.cc.bellcore.com +1 732 699 3042 +1 732 336 2980 fax | Principal Product Manager Language Standards Department Bellcore Bell Communications Research RRC AC-838 444 Hoes Lane Piscataway, NJ 08854 U.S.A. |
| Kenji Itoh | jastpro@po.ijnet.or.jp +81 3 3437 6135 +81 3 3437 6136 fax | Executive Director, JASPRO UN/EDIFACT Rapporteur for Asia Daiichi Daimon Bldg. 2-10-1 Shiba Daimon Minato-ku, Tokyo 105, Japan |
| Gail Jackson | jackson_g@supplytech.com +1 313 998 4141 voice or fax | Senior Vice President, Technology Harbinger SupplyTech 1000 Campus Dr. Ann Arbor, MI 48104 U.S.A. |
| Jean E. Kubler | jean.kubler@unece.org +41 22 917 27 74 +41 22 917 00 37 fax | Economic Commission for Europe United Nations Palais des Nations CH-1211 Geneva 10 Switzerland |
| Klaus-Dieter Naujok | klaus@premenos.com +1 510 688 2791 +1 510 602 2133 fax | Director, Standards and Technology Premenos 1000 Burnett Ave., 2 nd Floor Concord, CA 94520 U.S.A. |
| Scott T. Nieman | scott.nieman@connects.com +1 612 947 9871 +1 612 946 0390 fax | NORSTAN Consulting Sr. Consultant 7101 Metro blvd. Minneapolis, MN 55439 U.S.A |
| Alain Thienot | 100546.3352@compuserve.com +33 1 41 91 5973 +33 1 42 91 6026 fax | AFNOR-EDIFRANCE Tour Europe 92049 Paris la Defense France |

REFERENCE GUIDE "THE NEXT GENERATION OF UN/EDIFACT"

| | | |
|----------------------|---|--|
| François Vuilleumier | fvuille@ibm.net +41 31 322 65 23 or +41 31 322 66 11 +41 31 322 78 72 fax | Conseiller Direction Générale des Douanes Monbijoustrasse 40 CH-3003 Berne Switzerland |
| James K Werner | james.k.werner@boeing.com +1 425 237 6274 +1 425 237 8510 fax | THE BOEING CO. Supplier Network Support PPO BOX 3707, GC-FH Seattle, WA 98124-2207 U.S.A. |
| Peter Wilson | peter.wilson@eca.org.uk +44 171 432 2505 | Principal Advisor UKCEDIS Support Team ECA Ramillies House 1-9 Hills Place London, UK |