A LONGITUDINAL STUDY OF THE DIFFUSION OF THE ISO/IEC INFORMATION RESOURCE DICTIONARY SYSTEM STANDARD (IRDS)

BERNADETTE MARY BYRNE

Doctor of Philosophy

THE UNIVERSITY OF ASTON IN BIRMINGHAM

September 2001

THE UNIVERSITY OF ASTON IN BIRMINGHAM

THESIS SUMMARY

A LONGITUDINAL STUDY OF THE DIFFUSION OF THE ISO/IEC INFORMATION RESOURCE DICTIONARY SYSTEM STANDARD (IRDS)

The IRDS standard is an international standard produced by the International Organisation for Standardisation (ISO). In this work the process for producing standards in formal standards organisations, for example the ISO, and in more informal bodies, for example the Object Management Group (OMG), is examined. This thesis examines previous models and classifications of standards. The previous models and classifications are then combined to produce a new classification. The IRDS standard is then placed in a class in the new model as a reference anticipatory standard. Anticipatory standards are standards which are developed ahead of the technology in order to attempt to guide the market. The diffusion of the IRDS is traced over a period of eleven years. The economic conditions which affect the diffusion of standards are examined, particularly the economic conditions which prevail in compatibility markets such as the IT and ICT markets. Additionally the consequences of the introduction of gateway or converter devices into a market where a standard has not yet been established is examined. The IRDS standard did not have an installed base and this hindered its diffusion. The thesis concludes that the IRDS standard was overtaken by new developments such as object oriented technologies and middleware. This was partly because of the slow development process of developing standards in traditional organisations which operate on a consensus basis and partly because the IRDS standard did not have an installed base. Also the rise and proliferation of middleware products resulted in exchange mechanisms becoming dominant rather than repository solutions. The research method used in this work is a longitudinal study of the development and diffusion of the ISO/IEC IRDS standard. The research is regarded as a single case study and follows the interpretative epistemological point of view.

Key Words

CASE Tools, Repository, SQL, Middleware, IRDS, Classification of Standards, Research Methods, Longitudinal Study

ACKNOWLEDGEMENTS

I would like to express my grateful thanks to the School of Computing and Information Technology (SCIT) at Wolverhampton University for their enlightened staff development policy which has allowed me to continue this research on a part-time basis for several years. As well as a reduced teaching time-table in the initial years of the research they also provided a partial sabbatical in the final year which helped considerably in the completion of this work.

Also grateful thanks to my supervisor Dr Paul Golder for offering supervision and guidance over a long period which has seen many changes in the world of computing and Information Technology.

Bernadette Byrne September 2001

TABLES

FIGURES

CHAPTER ONE: INTRODUCTION TO THE THESIS AND THE RESEARCH METHOD

The history of computing spans approximately 50 years from the invention of the computer to the present day. The use of computers and software has extended from discrete systems to larger integrated systems to the point we are at today with large centralised or distributed systems. This research work takes one standard in Information Technology - the International Organisation for Standardisation (ISO) and the International Electrotechnical Commission (IEC) IRDS Framework Standard and follows its progress over time in the standards arena. A critical analysis of current classifications of standards is undertaken and the IRDS standard is placed in a classification scheme. The work examines the diffusion of the IRDS standard from when it became a full standard in 1990.

As with the development of any industry standards are necessary. It was quickly realised with computing that standards could help the portability and compatibility of systems and the ease of exchanging information between systems. Standards are an essential part of society and are constantly evolving. We expect to be able to travel by train across Europe without having to change railways because of different railway gauges, however, we would, when travelling from the U.K. carry an electrical adapter as the electrical current is not yet standardised between the U.K and the rest of Europe. Deming [Deming 1982] cites an example of how in World War II British ammunition would not fit the empty rifles of Belgian allies as there were no standard fittings. David [David 1987] cites an example of a large fire in Baltimore in 1907 where fire engines from neighbouring communities could not help put out the fire because the screw couplings on their hoses would not fit the fire hydrants in Baltimore. In the early days of the personal computers floppy disks had to be formatted to use on certain machines, whereas today disks are sold pre-formatted and can be used on almost any personal computer as a standard format has developed. There are obvious benefits to standardisation: interchangeability, convenience, safety, interoperability, risk reduction and improved trade. Standards in computing are necessary for a variety of reasons. Computer programming languages need to be standardised in order to make the code portable to different systems, and in order to bring about widespread use of the language. Standards can bring about compatibility and interoperability between systems. Trade can be aided by international standards. In the early 1990's the computing industry was aiming for a global interoperable infrastructure for exchanging information between systems. In October 1997, three of the main international organisations for standardisation the ISO, the IEC and the International Telecommunication Union (ITU) sponsored a conference named "Building the Global Information Society for the 21st century – Applications and Business Opportunities – Coherent Standards and Regulations" [http://www.ispo.cec.be]. The aim of the conference was to facilitate the development of a Global Information Society/Global Information Infrastructure (GIS/GII) through coherent standards and regulations. In order to bring about this GIS/GII the momentum towards global standards needs to be encouraged.

This work examines the evolution and diffusion of the ISO/IEC Information Resource Dictionary System (IRDS) Framework Standard. This standard has its origins in data dictionary systems and the catalogues and dictionaries of Relational Database Management Systems (RDBMS). The purpose of the ISO IRDS standard as described in the standard document is

> "to provide a common basis for the development of Information Resource Dictionaries. An Information Resource Dictionary is a shareable repository for a definition of the information resources relevant to all or part of an enterprise"

and the scope of the standard is to: -

> "describe the framework for a number of international standards that specify a specialised information system, called an Information Resource Dictionary System (IRDS)." [ISO/IEC 10027]

This 'specialised information system' will be achieved by having a central repository and services which operate within the repository to enable the organisation to achieve total resource management. As well as the IRDS framework standard there is also a Services Interface Standard [ISO/IEC 10728], which became a full international standard in 1993. The services interface structure is defined by using Structured Query Language (SQL). The Services Interface Standard provides full data definition statements and the necessary constraints for the tables which would make up the schema for the repository.

The author has always had an interest in Relational Database Management Systems and the structured query language SQL and was interested in the IRDS standard as it appeared to be a natural progression for RDBMSs to move into the area of total resource management. The author found it interesting to see how the functionality of a repository could be achieved using SQL as the model, as was outlined in the Services Interface Standard. The author was also interested in Fourth Generation Languages (4GLs) for database systems and felt that the IRDS standard would be one way to bring about standardisation in this area.

Research Questions

This work takes a particular standard in Information Technology - the ISO/IEC IRDS Framework Standard - places it in a model for standards, and investigates the progress of the IRDS Standard over time.

The work will answer the following research questions:

1. What is the role of formal standards bodies in the current standards arena and what is the current process of producing standards within these bodies?

2. What current models of standards and classifications of standards are available for understanding the standards process?

3. What are the underlying forces, which drive the diffusion of standards?

4. Are the development of anticipatory framework standards in traditional standards organisations suitable for the computing industry, with particular reference to the progress of the ISO/IEC IRDS Framework Standard. (Anticipatory standards are standards developed ahead of the technology in order to attempt to guide the market.)

5. To what extent has the ISO/IEC IRDS Framework Standard been successful in providing a common basis for the development of Information Resource Dictionaries?

Research Method

This research consists of a longitudinal study of the development and diffusion of the ISO/IEC IRDS standard. IRDS became a full international standard in 1990 and the diffusion of the standard is traced from this point to the time of writing (summer 2001). The focus of the work is three fold: - firstly explanatory - to explain classifications of standards and place the IRDS standard in context, secondly exploratory - to explore the diffusion of the IRDS standard, and thirdly descriptive - to describe the IRDS standard's functionality and place IRDS development in the history of computerised systems. The research is regarded as a case study. Yin [Yin 1994] divides case studies into exploratory, explanatory and descriptive case studies, however, he also says that there are not clear dividing lines between the three. Mitchell [Mitchell 1983] defines a case study as

> "a detailed examination of an event (or series of events) which the analyst believes exhibits (or exhibit) the operation of some identified general theoretical principal".

The case study as a research method is a form of empirical enquiry where empirical is taken to mean, "based on experiment, observation or experience rather than on theory" [Chambers 97]. As this work is based on observation of the diffusion and development of the IRDS standard it will be regarded as a case study. Yin [1994] suggests that with case study investigation it is useful to introduce a previously developed theory that can be used as a template with which to compare the empirical results of the case study. The questions that could be posed at this stage are: - to what extent has the ISO/IEC IRDS Framework Standard been successful in providing a common basis for the development of Information Resource Dictionaries? And secondly to what extent is the development of framework standards appropriate for formal standards organisations?

In the past there has been some criticism of the case study as a research method. Mitchell [1983] attributes the eclipse of case study research to the improvement in quantitative and statistical techniques due to the rise in use and the improved capabilities of computers, which made it easier to process large sets of data. The main criticism levied at case study research has been that it is difficult to generalise from a case study. The word "generalise" is intended to mean to infer or formulate theories. This was regarded as a problem with case study research as the criticism was that in order to formulate theories the case study should be representative of typical or similar problems in the area being studied. However, in the social sciences as opposed to the natural sciences it is often difficult to decide on what is typical, as there may be many variables involved. Yin[1994] and Walsham [1996] believe that it is possible to generalise from a single case study or multiple case studies and produce theoretical propositions. In this thesis a detailed examination of the formation and diffusion of the ISO/IEC IRDS standard is provided and from the work general inferences are drawn. Although part of this work can be described as a case study the work does not use some of the traditional case study tools of, for example, interviews, action research, and participation observation. Yin [1994] also points out that the use of a single case study is often appropriate where there is little previous work in the area. In the standards setting arena there has been little empirical work on the diffusion (degree of success) of framework standards. The main studies carried out are Bonino's [1991] exploration of anticipatory information technology standards and Egeydi's [1996] study of standards processes and standards policies which also examined the relationship between standardisation and technology development using the OSI Reference Model as a case study.

Positivist and Interpretive Research Methods

Many researchers in Information Systems [Braa & Vigden 2000, Gable 1994, Lee 1991, Smith 1989] categorise research methods based on a positivist or interpretive point of view and believe that there should be a strong distinction between positivist and interpretive research methods. A positivist approach takes a point of view that research should be based on facts which are quantifiable, whereas an interpretive approach takes a point of view that research is based on observation and interpretation of observable aspects. The interpretive paradigm could be described as qualitative, and the positivist as quantitative, although this is a simplistic dichotomy as they are both philosophical views rather than clear divisions in methods. Evered & Louis [1981] quoted in Braa & Vidgen [2000] regard the positivist paradigm as where the researcher is an "observer of the laboratory" where any change in variables have to be tightly controlled in order to be able to generalise and agree the representativeness of the experiment. In the positivist stance representativeness means that the case studied can be regarded as typical among similar cases. Whereas the interpretive paradigm is concerned with "making a reading of a situation in order to gain understanding" [Braa & Vidgen 00 op cit page 253]. This work on the ISO/IEC IRDS standard is based on making a reading of a situation in order to gain understanding on the operation of standards bodies and the dissemination of a certain type of standard. There is no quantification of facts or statistical analysis in this work.

The case study itself as a research method appears to fall into either a positivist or interpretive method. Braa & Vidgen [2000] distinguish between a 'hard case study' (positivist) and a 'soft case study' (interpretive). Smith [1989], while advocating an interpretive approach, states that positivists regard qualitative research as useful for exploratory studies which would then be followed up with some quantification. He states that positivists feel that "quantification is necessary to establish the validity of any findings" [op cit p56] The case study would appear to fall into the interpretive approach as its results are not usually quantifiable in the sense that there may not be a collection of data and statistical analysis of facts. However some case studies may also include survey research methods thereby including qualitative and quantitative methods [Gable 1994] and thereby combining positivist and interpretive approaches. Walsham & Waema [1994] regard the case study as falling into the interpretive camp

> "the use of a single case study as a basis for drawing inferences about a particular area of study is related to an interpretive epistemological stance" [page 151].

Walsham [1995] states that the researcher should state their philosophical point of view explicitly (interpretive or positivist) when writing up their work. Lee[1991] takes the point of view that positivist and interpretive approaches are in opposition. However other sources believe that there is common ground between the two views. Yin [1994] has an implicit positivist stance and dismisses the interpretive and positivist debate in a few lines. He takes the stance that there is "strong and essential ground between the two" philosophies of interpretive and positivist approaches.

The interpretative and positivist approaches to research have been the subject of many debates in Information Systems research. As regards the case study it can be regarded as positivist if it is the exploratory stage of some later quantification method, interpretive if it is a case (or several cases) studied to provide understanding, or both if it uses qualitative and quantificative methods. However the author believes that it is not necessary to get caught up in what Smith [1989] terms a "methodological maze" of different methods and philosophies. This research uses a case study of the development and diffusion of the ISO/IEC IRDS standard to draw inferences about the standard setting arena and the development of certain types of standards in formal standards setting organisations. The inferences drawn from the work can be used as the basis for further research in the standards setting arena. The author regards this work on IRDS standards as falling into the interpretative stance and that we can generalise from this case study and thereby increase our understanding of the wider standards process.

Research Design

In order to provide an analytical framework a broad research design for this work is included below. In the design the author has described the descriptive elements of this work and the areas where a contribution is evident.

Descriptive	Contribution
Describe the current main formal standards organisations and how they produce standards. Describe the increase in informal standard making bodies.	A synthesis of the area to provide a better understanding of standards development in Information Technology.
How the ISO/IEC IRDS standard evolved from its origins in data dictionaries and relational catalogues. Why there was a need for the standard.	Document the historical development of the ISO/IEC IRDS standard
Describe the ISO/IEC IRDS standard and demonstrate how the ISO/IEC IRDS standard fits into current taxonomies and models of standards.	A better understanding of the ISO/IEC IRDS standard. Previous work on models and classifications of standards is brought together to produce a new overall model. ISO/IEC IRDS is placed in this model.
Detail the diffusion of the ISO/IEC IRDS standard. Also including descriptions of similar and competing repository standards which were being developed at the same time as ISO/IEC IRDS.	Document the progress of the ISO/IEC IRDS standard from 1991. General inferences can be drawn about the appropriateness of developing this type of standard in formal standards making bodies.

Table 1.1 The Research Design

Main Contributions of this research

A critical analysis of current taxonomies/classifications of standards has been carried out. From this analysis a new taxonomy/classification of standards is produced which brings together previous work in the area. The ISO/IEC IRDS standard is positioned in the new classification.

The work also contains a longitudinal study of the evolution and diffusion of a framework standard developed in a formal standards organisation. Generalising from the case study of ISO/IEC IRDS inferences are made about the suitability of creating certain types of standards in formal standards bodies. This will help to refocus future work in this field. Overall the work also provides a synthesis of the area of standards development to create a better understanding of standards in IT.

Thesis Organisation

Chapter 2 describes the main traditional standards organisations. The administrative process for introducing a standard in two of the main organisations (the IEC and the ISO) is described. This process is broadly similar to the administrative process in other organisations and uses a consensus approach to develop international standards. The membership structures of the main international standards organisations are described. Informal standards setting organisations such as consortia are described and particular reference is made to the Object Management Group (OMG), the World Wide Web Consortium (W3C) and the Internet Engineering Task Force (IETF). The measures which the ISO have taken to attempt to speed up the standardisation process are described.

Chapter 3 describes the evolution of the meta-data concept and the evolution of the ISO/IEC IRDS standard up to the creation of the standard in 1990. The chapter describes how the ISO/IEC IRDS standard had its roots in data dictionaries and the relational catalogues of Relational Database Management Systems and then evolved towards information resource management. The chapter introduces the ISO/IEC IRDS four level architecture and the functionality that should be contained within it. The functionality of an IRDS is described including extensibility, version control, configuration management and impact analysis. The chapter outlines the main areas of functionality in the standard and comments how some of these are similar to data dictionary functionality. The differences between an active and a passive repository are also outlined. The chapter also makes some comparisons between the ISO/IEC IRDS standard and the ANSI IRDS standard.

Chapter 4 introduces the ISO/IEC IRDS family of standards and specifically considers the ISO IRDS Services Interface Standard. The data structures for the Services Interface Standard are outlined. The use of case data interchange formats and the usefulness of an extensibility feature in an IRDS are introduced. The chapter also describes the other standards in the ISO/IEC IRDS family of standards.

Chapter 5 describes other main standards, which had a similar purpose to ISO/IEC IRDS, and which were developed in the late 1980's and early 1990's. These standards include the Portable Common Tool Environment (PCTE), AD/Cycle sponsored by IBM and the American National Standards Institute (ANSI) IRDS standard. This chapter also describes the rise of Computer Aided Software Engineering (CASE) tools which took place in the late 1980's and early 1990's and compares integrated CASE (I-CASE) tools to repositories. This chapter also introduces the CASE Data Integration Format Standard (CDIF) and the Common Object Request Broker Architecture (CORBA).

Chapter 6 takes a chronological view of models and taxonomies of standards, which have been developed so far. This chapter reviews these classifications of standards and current models of standards. The chapter begins with David's [1987] taxonomy of standards and continues with Cargill's [1989] classification of standards and Cargill's [1989] model of standards and concludes with Bonino & Spring's [1991 & 1999] work on anticipatory standards. A normalisation of the previous work on the classification of standards is carried out and a new model of standards classification is proposed. The ISO/IEC IRDS standard is then placed in the new classification.

Chapter 7 discusses the prevalent economic theories on the diffusion of standards in a compatibility market. The characteristics of a compatibility market are described and the chapter discusses why the law of diminishing returns is not relevant in an Information Technology compatibility market. Economic theories on the diffusion of standards are discussed with particular relevance to de facto standards, as this is the main area where economic theories have been applied. This chapter also examines previous research on the introduction of gateway and converter devices into a compatibility market.

Chapter 8 covers the diffusion of the ISO/IEC IRDS standard from when it became a full international standard in 1991. This chapter also considers the diffusion of competing standards; ANSI IRDS, AD Cycle and PCTE. The parallel rise of CASE tools and the overlap between integrated CASE tools, which required a central repository, and the ISO/IEC IRDS standards is also discussed. Generalising from the case study of the ISO/IEC IRDS standard conclusions are drawn about the suitability of creating these types of standards in formal standards bodies.

Chapter 9 summarises the contributions of this research to the field of standards development. The research questions outlined in chapter one are answered and conclusions are drawn about the suitability of creating anticipatory reference standards in formal Standards Development Organisations.

CHAPTER TWO: FORMAL STANDARDS ORGANISATIONS AND CONSORTIA

This chapter reviews the main formal standards bodies currently in operation and the general administrative processes for producing de jure standards. The chapter continues by describing informal standards bodies such as consortia and the types of standards they produce. The chapter also covers a review of work in the area of standards setting and standards organisations.

The main formal international standards bodies for producing de jure standards for computing are:
- The International Organisation for Standardisation (ISO) and the International Electrotechnical Commission (IEC). The term 'De jure' means 'force of law' however although the majority of standards produced by national and international standards bodies do not have the 'force of law' behind them – that is there is no legal requirement to use them – they are still referred to as 'de jure' standards. It will depend on the individual countries to decide whether they have mandatory or voluntary standards compliance for a particular de jure standard.

The International Organisation for Standardisation (ISO)

The ISO is a non-governmental organisation and was founded in 1947. The mission of ISO is to

> "promote the development of standardisation and related activities in the world with a view to facilitating the international exchange of goods and services, and to developing co-operation in the spheres of intellectual, scientific, technological and economic activity."(http://www.iso.ch).

ISO is made up of national standards bodies from approximately 130 countries. It has a hierarchical structure with a number of technical committees. Each technical committee has a number of subcommittees and each subcommittee will have a group of members from all over the world who will have regular face-to-face meetings. Within each subcommittee there are smaller working groups who undertake most of the standards writing work. ISO has many member bodies of which the British Standards Institute (BSI) is one as is the American National Standards Institute (ANSI). In the member bodies the organisations have their own groups which participate in the ISO Technical Committees and some of the members of these groups attend the regular subcommittee meetings.

ISO members are divided into three categories. A member body, a correspondent member, and a subscriber. A member body is the national organisation which is the most representative of standardisation in its country (for example the British Standards Institute (BSI) for the United Kingdom.). Therefore there can only be one member body per country. The member body from each country has the responsibility of informing other interested parties in their country of ISO activities and of representing their country's interests at meetings. The member bodies have full voting rights.

Member Body	A member body is a national organisation	Full Voting Rights
Correspondent Member	An organisation from a country which does not yet have a fully developed standards activity	Do not have voting rights
Subscriber	For countries with small economies.	Do not have voting rights.

Table 2.1 ISO Membership

A correspondent member is usually an organisation from a country which does not yet have a fully developed standards activity. The correspondent members do not take an active part in standards development and do not have voting rights. However, they can be kept fully informed about the work that interests them. The subscriber membership is for countries with small economies. These members pay a reduced membership fee.

The International Electrotechnical Commission (IEC)

The other main international standards body is the International Electrotechnical Commission (IEC). The IEC produces standards for all electrical, electronic and related technologies. The IEC was founded in 1906 and has more than 50 participating countries as members. The IEC works with other standardisation bodies such as the International Telecommunications Union (ITU) and ISO.

There are three types of membership within IEC. Full membership gives full participation and voting rights. Full members are the National Committees from each country. Each National Committee has equal voting rights. There is also an Associate Membership category for countries with limited resources. Associate members have observer status and can attend meetings but have no voting rights. There is also a pre-associate membership to encourage and provide support to countries who are in the process of forming a National Committee. In general the types of members are similar to the ISO membership categories. The IEC's mission is to:

"promote, through its members, international co-operation on all questions of electrotechnical standardisation and related matters, such as the assessment of conformity to standards, in the fields of electricity, electronics and related technologies". [http://www.iec.ch]

Full Membership	Full members are national bodies from each country	Each national committee has equal voting rights
Associate Membership	For countries with limited resources.	Associate members have observer status. They can attend meetings but have no voting rights.
Pre-Associate Membership	**For countries who are in the process of forming a National Committee.**	This membership is to encourage and support countries who would like to join IEC and form a National Committee.

Table 2.2 IEC Membership

Within IEC approximately 200 technical committees and subcommittees, and approximately 700 working groups carry out the standards work. The IEC has a similar hierarchical structure to ISO with the technical committees and the sub committees preparing the technical documents within their areas. The documents are then submitted to National Committees for voting to take place. A Technical Committee will be made up of National Committees from various countries. The committees will decide themselves when to hold meetings. Each country may have their own committees for dealing with and discussing the work of the international committees.

Joint Technical Committee on Information Technology (JTC1)

The Joint Technical Committee on Information Technology was formed in 1987 by the IEC and the ISO. The aim of the Committee, usually known as JTC1, is to avoid any overlap in standards development by the ISO and IEC by the committees working together on standards for Information Technology. ISO/IEC JTC1 includes 17 subcommittees. JTC1 members are the national standards bodies from countries in the world. At the time of writing there are 26 participating members and 37 observer members. There are also internal and external liaison members. JTC1 holds plenary meetings once every twelve months. The JTC1 subcommittees are grouped within 11 technical directions. For example Sub Committee 32 has the title Data Management and Interchange (Figure 2.1). This sub committee deals with, among others, the current SQL standard and the ISO/IEC IRDS standard which is the subject of the case study in this work. National standards organisations will then have their own groups which work with the international subcommittees. For example, the BSI IST/40 technical committee is the committee which deals with and co-ordinates the work of the United Kingdom with ISO/IEC JTC1 SC32 Data Management and Interchange. IST/40 has members who work with and attend the annual international SC32 meetings and then feedback results to IST/40. Some of the members of IST/40 will have voting rights within ISO/IEC JTC1 as representatives of the British Standards Institute.

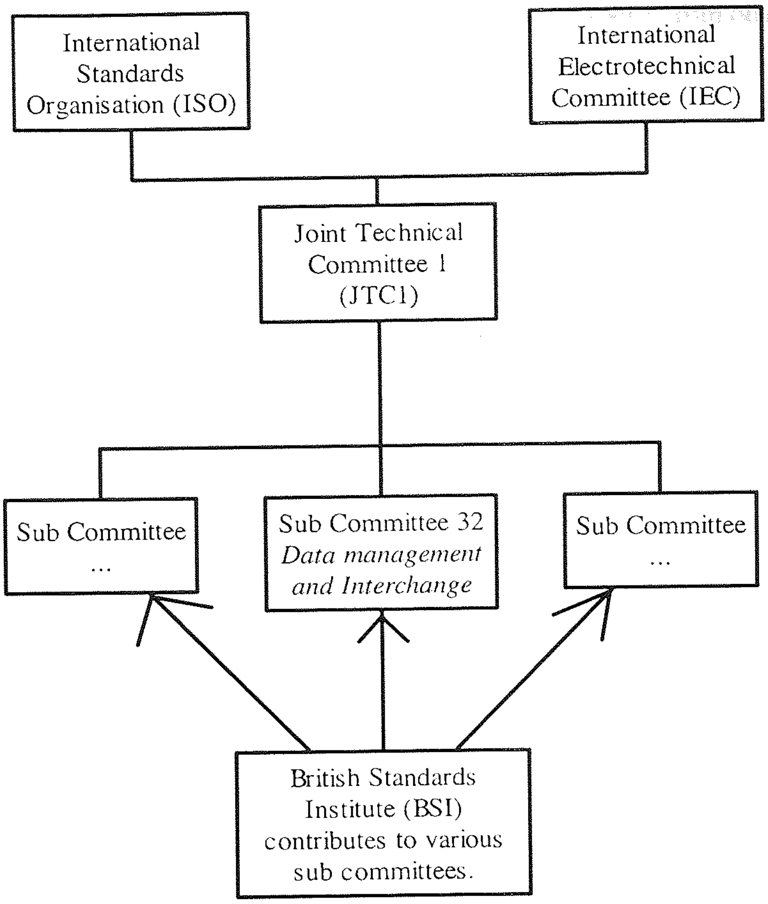

Figure 2.1 Structure of the ISO and IEC Standards Organisations.

The European Computer Manufacturers Association (ECMA)

The European Computer Manufacturers Association (ECMA) was founded in 1961 and has non-voting observers on many of the ISO technical committees. ECMA is a European group of companies concerned with the standardisation of information and communication systems. The results of the ECMA standardisation work is available free of charge. A compact disc containing current standards can be obtained free of charge from their web page. [http://www.ecma.ch]. ECMA proposes new standards to ISO and enhances existing standards. ECMA also develops standards in co-operation with other National, European and International organisations.

ECMA consists of ordinary, associate and Small and Medium Enterprise (SME) members. Ordinary and SME members are companies which develop and produce in Europe with the SME's having a smaller turnover than ordinary members have. An associate member is a company which has an interest in one or more of the Technical Committees but does not qualify as an ordinary member. At the technical level ECMA is open to experts from other organisations, for example non-profit making organisations and universities. Although these experts may not qualify for full membership they still exercise full voting rights on Technical Committees. The ECMA work is divided into Technical Committees and task groups. Any interested party can request an invitation to meetings. ECMA produces two types of publication: ECMA standards and ECMA Technical Reports. A majority of at least two-thirds of all the ordinary members is required for approval of a standard or technical report.

National Standards Bodies

Most countries will have their own formal standards making bodies, which will usually be members of ISO. The British Standards Institution (BSI) is the British National Standards body. BSI was founded in 1901 and is independent of the British government although it receives some funding. It is a non-profit making organisation and aims to be impartial in its dealings with the private and public sectors. BSI has no obligation to adopt ISO standards although it will try to align the work of BSI with the standards within ISO. Examples of other national standards bodies organisations are the Association Francaise de Normalisation (AFNOR) for France, and Deutsches Institut fur Normalische (DIN) for Germany, ANSI (the American National Standards Institute) is the official governmental standardisation body for the United States of America (USA).

Other Standards Bodies

The Comite European de Normalisation Electrotechnique (CENELEC) was founded in 1973 and is the European committee for electrotechnical standardisation (the European equivalent of IEC). It is the European standards organisation in its field. The European Committee for standardisation (CEN) has a division called the Information Society Standardisation System (CEN/ISSS) which deals with standards in ICT and IT. The ITU (International Telecommunications Union) is an agency for the United Nations. It has 160 member countries. The ITU also has an official liaison role in JTC1. The IEEE is a United States professional body (Institute of Electrical and Electronics Engineers) which produces standards, which are then fed into ANSI. ANSI works in a different way to BSI in that it has within it organisations which are commissioned to write standards.

As outlined above the global formal standards making organisations are made up of various bodies throughout the world some of which are linked to the government of the country (ANSI) and some which are independent (BSI). There are also other bodies like ECMA which serve a group of countries. The national bodies which make up the membership of the ISO and IEC are therefore diverse in nature.

The Process of Standardisation - The Preparation of Standards

The process a standard goes through before it becomes an international standard is similar in many of the standards organisations. Cargill [1989] puts forward a generic five-stage model which outlines the stages in the development of a standard. The five stages are: preconceptualization, conceptualisation, discussion, writing the standard and implementing the standard. Figure 2.2 shows the progress of a standard from the preliminary stage to becoming a full International Standard within ISO/IEC and these stages are broadly similar to Cargill's model.

Proposal Stage

A technical committee or subcommittee of a technical committee may propose a new work item for a new standard or a new part of an existing standard. The proposal is accepted if a majority of the participating members of the Technical Committee (TC) or Sub Committee (SC) votes in favour of the proposal; also, at least 5 members of the group must agree to participate actively in the work. (The members involved in the work may be representatives from different countries as the TC/SCs are made up of representatives from member countries of JTC1). If there is a majority vote a project leader is appointed.

Preparatory Stage

The TC/SC will set up a working group consisting of the project leader and technical experts nominated by the members who agreed to actively participate in the project and other members who may now decide to actively participate. The working group will prepare successive working drafts until they decide that the best solution has been arrived at. The working draft is then passed to the parent TC or SC for the committee stage. ISO/IEC expect a working draft to be arrived at within 6 months.

Committee Stage

The committee stage marks the beginning of the consensus building stage. The Committee Draft (CD) is distributed for comments or sometimes voting to the participating members of the TC or SC. Successive CD's may be considered until consensus is arrived at. Time limits are set for the returns of comments for each successive draft. Where there is difficulty in achieving consensus, if two thirds of the participating members vote to accept the CD it will go forward to the enquiry stage, although the ISO states that "every attempt shall be made to resolve negative votes" [ISO/IEC Directives Part 1] The ISO and IEC expect a committee draft within two years. When the process is complete and consensus is arrived at the document is finalised and submitted as a Draft International Standard (DIS).

Enquiry Stage

At this stage the Draft International Standard (DIS) is circulated to all ISO member bodies for voting and comments within 5 months. If two-thirds of the majority of the participating members

Figure 2.2 The Preparation of a Standard - Based on the ISO/IEC model

Proposal Stage

A National Committee originates a proposal which is sent to the appropriate IEC or ISO Technical or Sub Committee.

↓

The Committee votes on the proposal. If the vote is positive it is included in the work of the committee and becomes a work item with a project number. → *A simple majority vote*

↓

Preparatory Stage

Successive Working Drafts (WD) are prepared within the TC or the SC until a final WD is accepted.

↓

Committee Stage

WD is submitted to the National Committees as a Committee Draft (CD) for comment resulting in a Draft International Standard (DIS).

↓

Enquiry Stage

The DIS is submitted to participating members for a five-month voting period. → *CD is approved if a majority of two thirds of the votes cast by participating members is in favour <u>and if</u> the number of negative votes cast by all national committees does not exceed one quarter of all the votes cast.*

↓

A revised CD is sent to the central office within a period of four months for Final Draft International Standard (FDIS) approval.

↓

Approval Stage

The FDIS is circulated to the National Committees for a two month voting period. → *Vote must be explicit either positive, negative or abstention.*

↓

If the FDIS is approved it is published as a full International Standard. → *FDIS is approved if a majority of two thirds of the votes cast by participating members are positive <u>and if</u> the number of all negative votes cast by all national committees does not exceed one quarter of all votes cast.*

of the TC/SC are in favour and not more than a quarter of all the votes cast are negative then the DIS can be submitted as a Final Draft International Standard (FDIS). If this is not the case the DIS is returned to its originating TC or SC for revision and further comment. The ISO expects a FDIS within three years although there are processes for extending this time.

Approval Stage

The FDIS is circulated to all ISO member bodies for a final vote within two months. The vote at this stage is either a Yes/No vote or an abstention. The FDIS is approved as an international standard if a two-thirds majority of the participating members of the TC/SC are in favour and not more than one-quarter of the total number of votes cast by all national committees are negative. If the FDIS is not approved it is returned to the originating TC or SC for revision in the light of the negative comments which have been received. When the FDIS is approved it is published as a full international standard. The International Standards are reviewed every five years by the originating SC or TC. Every five years the participating members of the SC or TC vote to decide if the standard should be confirmed, revised or withdrawn.

Working Groups

Within ISO/IEC the Information Technology standards are devised by working groups of subcommittees of JTC1. The working group decides the contents of the standard or revisions to the standard and then the standard is passed to the parent committee for comments and voting. As an example the latest SQL standard [ISO/IEC 9075] is dealt with in WG.3 Database Languages. The standard is then passed to the parent committee which is JTC1/SC32 (subcommittee 32 has the title Data Management and Interchange.) WG.3 will have representation in various national standards bodies throughout the world. Members of the subcommittee will vote on the standard or the revisions. Once a Draft International Standard is agreed within the originating TC or SC the standard then passes to all ISO member bodies for voting. The ISO, IEC and BSI make draft standards available for public comment for 60 days. The general public can comment on the standards through an electronic comment form on the BSI web site which also announces which standards are available for comment. [http://www.bsi-global.com]

Consensus Standards

The standards organisations mentioned above all have a similar hierarchical structure with Technical Committees and Sub Committees. Their procedures for developing standards are similar with the technical work taking place in Technical Committees or working groups, discussion taking part in subcommittees and a system of voting. Standards developed in this way are called consensus standards. Although the dictionary definition of consensus is "agreement" Cargill[1989] points out that:

"Consensus does not mean majority agreement; it indicates that a commonality of perception and opinions has been achieved. To gain consensus, there must be time to resolve disputes, not by compromise ...but by redefining the problem so that it lends itself to a solution that will meet the needs of all participants. The process takes time and effort:" [op.cit. page 32]

The consensus process in formal Standards Development Organisations is an 'open' process. Interested parties can join standards groups (for a fee in some cases) and participate in the standard making process. The standard will be arrived at by agreement and some compromise. The ISO/IEC definition of consensus is:

"General agreement, characterised by the absence of sustained opposition to substantial issues by any important part of the concerned interests and by a process that involves seeking to take into account the views of all parties concerned and to reconcile any conflicting arguments". [ISO/IEC Directives 2001]

However, the main disadvantage and criticism of the de jure standard making process is that it can take a long time. This is partly because of the time taken to resolve disputes in order to arrive at consensus and partly because of the time taken to co-ordinate activities between different countries. The process can take 4 to 5 years from inception to the first draft and in some cases much longer. In the area of Information Technology this is a problem as the technology could be out-dated by the time the standard is published.

Consortia in the Standards Arena

Since the mid 1980's a number of consortia dealing with IT Standardisation have formed. As the IT industry grew standards became more complex, Open Systems became important as users were beginning to demand interoperability among products. Standards Development Organisations (SDOs) could not develop these standards quickly enough and new organisations (consortia) began to form. These new standards setting organisations were also referred to as "Grey Standards Bodies" [Egyedi 1996 & 2000] and the term "grey standardisation" became used for organisations which developed standards which did not emanate from the formal standards bodies. Weis and Cargill [1992] divide consortia into 3 types which had emerged by the end of the 1980's. Implementation consortia, Application consortia and Proof-of-Technology consortia.

Application Consortia

Application consortia are described as a group of vendors who aim to accomplish standardisation of a technology which is not yet agreed by a Standards Development Organisation (maybe due to some disagreement within the Standards Development Organisation or by a group that wants to avoid the formal lengthy standardisation process and get a product on the market more quickly). The main example in 1992 of this type of consortia was the Open Software Foundation (OSF). The

OSF consisted of International Business Machines (IBM), Hewlett-Packard (HP) and Digital Equipment Corporation (DEC). This group of three worked to develop their own open operating system called OSF/1 in order to try to halt the increasing market of AT&T's proprietary UNIX operating system. OSF also developed a Graphical User Interface (GUI) called MOTIF again to try to compete with the growing trend towards Sun Microsystems' Open Look GUI. MOTIF was actually placed in an IEEE standards committee, as also was the Open Look GUI from AT&T. This type of consortia can form to prevent a rival from having exclusive success with a competing product.

Implementation Consortia

This group is described as a consortia which takes a completed standard and tries to make it usable. The example given by Weis and Cargill [1992] is the Structured Query Language (SQL) Access Group (SAG). This was originally a group of SQL users who found difficulty in using the SQL standard because of its ambiguities.

Proof-of-Technology Consortia

The third group of consortia is described as Proof-of-Technology Consortia. The purpose of these groups is to try to begin the consensus process during the emergence of new technology. The Object Management Group (OMG) is an example of this type of consortia. The idea behind these groups is for competing vendors to try to work together on emerging standards for new technology and thereby avoid having large financial outlay on competing products.

The OMG Consortia: An example of a proof-of-technology consortia heading

The OMG was founded in April 1989 starting with initially 11 companies which included Data General, Hewlett-Packard, Unisys and Sun Microsystems. By October the OMG was established as a not-for-profit organisation. It was formed in order to create "a component-based software marketplace by hastening the introduction of standardised object software." (http://www.omg.org). Today the OMG claims to be the largest software consortium in the world with a membership of approximately 800 members. It is divided into three sections: the Platform Technology Committee (PTC), the Domain Technology Committee (DTC) and the Architecture Board. It operates a consensus building approach and has different levels of membership. Membership fees are based on the amount of influence which the member requires, they may have voting rights on the working groups (called task forces in OMG), the technical committees themselves, or they may have the rights to submit technology at the technical committee level. A typical time frame for the approval of a standard is 18 months.

The OMG is an example of a consortia which operates on a consensus basis however the membership fees can be high for full voting rights. Members have to pay fees depending on the amount of influence they want to have. Full voting rights in both committees and task forces and special interest groups cost $65,500 in the year 2000 for a company with a large turnover to $8,750 for a company with a small turnover. The OMG charges for copies of their standards and the OMG will only approve standards where there is already an implementation.

> "A standard with no implementation is not worth the paper on which it's printed; therefore, OMG will never promulgate a standard with no existing implementation." [Soley 1995 page 611]

OMG has had continued success with its Common Object Request Broker Architecture (CORBA) and the OMG claim that CORBA is: "the de facto standard for ensuring open, distributed computing" [http://sisyphus.omg.org/news/about/myths.htm]. OMG is an example of a "proof-of-technology" consortia which has had considerable success in producing standards. One of the reasons for its success is that it attempted to establish standards before any major vendors had proprietary solutions. In this case the industry were willing to take on the OMG standards. The OMG now has links with JTC1 through a new fast tracking scheme introduced by the ISO/IEC.

The IETF: An example of a proof-of-technology consortia

The Internet Engineering Task Force (IETF) is an example of a recent consortia which would fall into the Proof-of-Technology group above. IETF started in 1986 with 15 attendees at the first meeting. It is made up of a "loosely self-organised group of people who make technical and other contributions to the engineering and evolution of the Internet and its technologies". [http://www.ietf.cnri.reston.va.us)] There are three annual face-to-face meetings but IETF stress that these meetings are not conferences and that IETF is not a traditional standards organisation. Unlike the OMG there is no membership fee and membership is open to any interested party. The nature of the Internet functions: transfer of files, exchange of electronic mail and accessing remote computers requires a set of networking protocol standards and the family of TCP/IP protocols was developed for use in the Internet. The continuing evolution of these protocols is managed by the IETF. Like the formal SDOs most of the work is done in working groups, however unlike formal bodies the documents produced are made publicly available on the Internet and most of the discussion and final decisions are made by email. There is no formal membership of a working group but someone who is on the mailing list for a working group is deemed to be a member. A working group disbands after it has finished its work. There is also a "Request For Comments" (RFCs) document series which reports official standards and statements of procedures. Again the working drafts, minutes of meetings and RFCs are all available free of charge on the Internet.

In recent years the IETF membership has grown larger and it now holds meetings outside the US and has become international. It also appears to be evolving a similar hierarchical structure as is present in the formal standards bodies. Lehr [1995] outlines three main reasons why he felt that IETF has been successful: -

- Firstly because IETF has a short life cycle for standards. For example any standard which has become a draft is removed from the list if it is not approved as an Internet standard within 6 months, also, any item being considered as a standard has to have at least two independent implementations in operation to progress to a draft.

- Secondly, users have been directly involved in the IETF from the beginning and were concerned with standardisation and interoperability from the outset.

- Thirdly, electronic communications on voting on standards and the electronic discussion of items have made document transmission quicker and more effective.

These reasons are also supported by Rutkowski [1995]. Also if the standards are freely available on the Internet it helps in the dissemination of the standard. However Lehr [1995] also comments that it is the nature of the technology, software communications, which has also helped in making the standards successful and he points out that because the IETF's procedures have worked in the Internet does not make them a role model for standardisation in other areas.

The World Wide Web Consortium (W3C): An example of a proof-of-technology consortia

The World Wide Web Consortium (W3C) was founded in 1994 by Tim Bernes-Lee. Although originally based at the Massachusetts Institute of Technology in the United States W3C is now an international consortium with over 400 members worldwide. The mission of W3C was to "lead the World Wide Web to its full potential by developing common protocols that promote its evolution and ensure its interoperability." (http://www.w3c.org). In the six years since its foundation the W3C has developed over 20 technical specifications for the Web. (W3C uses the words 'technical specification' or 'recommendation' rather than the word standard.) Membership of W3C is open to all types of organisations. Its membership scheme is tailored for organisations rather than individuals but it does have an affiliate membership which individuals can join. In the year 2000 the annual membership fee, which in the first instance must be for three years, is $US50, 000 for a full member and $US 5,000 for an affiliate member. Not-for-profit organisations, for example government agencies or educational establishments may join as affiliate members. Both types of members have the same benefits. On its web page the W3C states that it is a non-profit organisation and that its activities are vendor neutral. It is funded partly by its members and also with some government funding. W3C states that consensus in its operations is one of its most important principles. If unanimity cannot be achieved decisions are reached by "considering the ideas and viewpoints of all participants, whether W3C members, invited experts or the general public" (http://www.w3.org). However, Rada [2000] points out that the W3C does have a director who has the final say in any vote. The general public can in fact participate in discussions on public W3C mailing lists, or by trialing some of the software and offering feedback. Also all of the W3C technical reports and software are available free of charge to the general public. Although the W3C consortia has been active for only six years it has been successful in developing standards for the Web. Its combination of a vendor neutral organisation, funding from interested parties, a consensus approach and specifications and software available free of charge has worked well in the present climate. Also, similar to the IETF and the OMG, W3C standards are being developed in line with the technology and before there are any major proprietary solutions from single vendors. Similar to the OMG, W3C has now developed links with ISO/JTC1 and there is an ISO-HTML standard which is a subset of W3C HTML 4.0. In 1998 a draft agreement was produced between JTC1 and W3C in which ISO/IEC agree to adopt W3C standards as ISO/IEC standards as quickly as possible. Under the agreement ISO would still sell the standards documents and W3C would still make them available free of charge from its website. Rada [2000] points out that there is an anomaly with the agreement as the consensus processes used in both organisations are different as W3C has a director who has the final word in any vote, also, the ISO/IEC could be seen as "rubber-stamping" other organisations standards and then copyrighting and selling them.

The Success of Consortia

Three of the most successful consortia have been the OMG, IETF and W3C. In all three consortia vendors have come together, bypassing the International Standards Organisations, to produce standards ahead of the market in order to reduce their own risks. Cargill writing in 1992 thought that these types of consortia actually can augment the standards process by producing de facto standards which can later become formal international standards. This is now happening with some OMG and W3C specifications. Updegrove [1995] also supports this view and adds that diversity in the standards setting arena is useful. However one of the main objections to consortia is that they may not use a true consensus process in developing the standards. [Rada 1995] [Rada & Berg 1995], [Rada 2000]. If the fees to join a consortia are high, as they are with the OMG and W3C, it means that only a limited number of companies will join. The Members of the consortia may have a vested interest in promoting their own proprietary solutions.

Three of the most successful consortia have come from the proof-of-technology group. Cargill & Weis [1992] put forward a theory, based on group theory and economic theories, as to why consortia developed. They come to the conclusion that the consortia were "a vendor solution to the problem of long delays and ambiguous compromises in the formal standards process."[op cit page 563.] Writing in 1995 Cargill argues that the Open Systems movement also led to the growth in consortia in the late 1980's. Users needed standardised solutions for Open Systems and many of the consortia focused on providing these solutions. Updegrove [1995] compares consortia to trade associations where members of a trade would get together to promote their own interests to the general public. He states that some consortia were formed to try to prevent the adoption of a proprietary standard by a competitor (for example the OSF) and others to attempt to facilitate the adoption of a certain technology (for example the OMG).

	Membership Fees Charged	Type of Membership	Standards distributed free	Consensus
OMG	Yes - high	Individuals and Organisations	No (but only nominal amount charged)	Yes - but some members have higher levels of voting rights.
IETF	No	Individuals and Organisations	Yes	Yes
W3C	Yes - high	Organisations mainly, (individuals through the affiliate member scheme but tailored for organisations)	Yes	Yes (but a Director has final say in votes.)
ISO/IEC/JTC1	Yes - low	Countries only	No	Yes

Table 2.3 Summary of the Procedures of the main Consortia compared to ISO/IEC

Table 2.3 shows the differences between the three example consortia the OMG, the IETF, W3C and the ISO/IEC JTC1. The IETF is the only organisation which does not charge membership fees. The main difference in membership between the four organisations is that the ISO/IEC membership is restricted to countries only, whereas the consortia membership is mainly made up of organisations and individuals. The four organisations operate a consensus policy, however the W3C has a director who has the final say in voting, whereas within the OMG some members have higher levels of voting rights if they have paid for the higher privileges. Both the JTC1 and the OMG charge for standards, however the OMG only charges a nominal amount.

Charging for Standards

The BSI and the ISO make some of their revenue from selling standards documents. To have to pay for the standard seems an obvious barrier to the dissemination of the standard. However on the other hand ECMA distributes its standards free of charge. In the consortia domain the OMG charges for its standards documents (although this is only a nominal amount to cover paper costs) whereas the IETF makes its standards freely available on the Internet. Arguments as to whether standards should be free or sold are put forward in [Rada & Berg 1995]. Rada & Berg advocate that SDO's should increase their membership fee to defray the cost of copies of standards and that all drafts of standards should be available free of charge (possibly on the Internet) and only the final document should be charged for. This has the advantage of increasing the discussion on the draft standards which is unlikely to happen if the drafts are charged for. They also put forward the argument that SDO members are usually volunteers who develop the standards which are then copyrighted by the SDO's and the members do not receive any royalty payments. Rada [2000] says explicitly that all standards documents should be free:

> "The traditional standards development organisations continue to prevent the free flow of standards documents on the Internet. These documents should be free"

He makes the point that one of the reasons why the TCP/IP protocol has been so successful may have been because it was freely available on the Internet and therefore used easily by academic researchers, whereas the equivalent ISO standard had to be purchased in hard copy. However an advantage of distributing in hard copy is that the standard specification cannot be easily changed whereas an electronic copy could be easily added to or amended without the changes being obvious. As mentioned earlier the ISO, IEC and the BSI make draft standards available for public comment for 60 days. However the draft standard itself is not available electronically but a hard copy of the draft has to be purchased from the BSI. A comment form can then be filled in electronically on the BSI web site.

Intellectual Property Rights

Intellectual Property Rights (IPR) are protected through copyrights and patents. Organisations and individuals need to be encouraged to produce new products or new ideas and intellectual property rights are one way of encouraging them. Individuals and organisations will not be motivated to produce new products and ideas if there is no gain from doing so. Therefore their rights need to be protected if their products or ideas are placed in the public domain. This is problematical in the standards arena. In traditional SDO's it is member representatives from each country who, within working groups, produce standards. If one vendor is responsible for a property and brings that property to a committee for standardisation the intellectual property rights belong to the vendor and an agreement has to be reached as to how these rights will be enforced. (Most SDO's will have an IPR policy). However if the standards produced are the result of combined activities from different organisations the situation is not so straightforward. The IPR problem has to be addressed by all SDO's both the formal organisations and the more recent consortia.

Cargill's Five Stage Model

Cargill produced a Five Stage Model of the standardisation process [Cargill 1989]. Writing in 1989 he was concerned that the formal standards organisations had not moved on from a "two-stage" model, that is a model in which a de facto standard is taken and formalised in a standards organisation. He advocated that the formal standards organisations should move towards creating reference standards

> "reference models likely will become the premier planning devices for the industry, acting as change agents to influence the evolution of IT." [op cit p52]

Figure 2.3 Cargill's Five Stage Model

He was also concerned that there was a division between the needs of the providers of standards and the needs of the users of standards and that an attempt must be made to draw these two together. The first two stages in the five-stage model above in Figure 2.3 - the reference model and the industry standards - were relevant to the providers, the last two stages, the Systems Profiles and the Application Implementation, were relevant to the users. The middle stage, split by the dotted line in Figure 2.3, was relevant to both users and providers. Cargill's five-stage model was a model of the process of creating standards and was separate from his classification of different types of standards which is covered in Chapter 6. He was concerned that the providers of standards were too far removed from the needs of the users of standards. The providers were concerned with a global model which covered all the possibilities for an implementation and in fact served as a reference model for providers, whereas the users were more concerned with an implementation which gave an immediate solution to a particular problem. The model moves from the global reference model to the more specific systems profiles - which are in fact subsets of the global model. The original model is reproduced above in Figure 2.3. Cargill stated that within the Five Stage Model the stages could occur in any order but that eventually the market will develop into the five stage model. The functional profile stage was intended to be the stage which brought together the providers and the users by describing a set of functions from the industry standard and at the same time a set of functions required by a user group.

ISO's Attempts to Speed up the Standardisation Process.

ISO has had a long standing Fast Track Procedure which allows organisations like ECMA or the national standards bodies from member countries to submit a standard into a final JTC1 ballot in order for the standard to become an international standard. This allows standards which have been developed in the national standards bodies by or in other standards bodies, for example ECMA, to be submitted for approval as an international standard at the approval stage of the process outlined in Figure 2.2. The standard would be submitted as a final draft international standard (FDIS) or in some cases (depending on the category of organisation submitting the standard) at the enquiry stage (Draft International Standard (DIS)). This could happen where a standard developed in one country is also being used in, and becoming popular in, other countries, in which case there is obviously no need for it to go through the complete standards process.

More recently ISO has recognised that there are problems in the standards process especially in technical areas of rapid change like Information Technology. In November 1993 the ISO held an extraordinary session in order to propose new structures for the organisation which would help to achieve technical standards more quickly. A suggested target for ISO was the development of standards in "three years start-to-finish"[Smith 1994]. A resolution was passed which endorsed this goal by the end of 1996.

In 1994, as one of the new structures to speed up the standards process, JTC1 introduced the Publicly Available Specification (PAS) procedure. With the PAS procedure standards developed elsewhere, perhaps in consortia, can be submitted and accepted as ISO standards without going through the usual initial ISO processes. An organisation which owns a Publicly Available Specification/standard which could be transposed into an international standard can apply to JTC1 to be a PAS submitter. JTC1 regarded the introduction of the PAS system as a "landmark decision" [ISO/IEC JTC1 N5746] as it opened the doors to other organisations which are not linked to national or international standards organisations to put forward their own specifications as standards. (This differs from the Fast Track procedure which was intended for national standards organisations and bodies like ECMA. ISO/IEC JTC1 had consortia in mind as the type of organisations they would expect to submit a PAS). The organisation submitting a PAS initially has to be accepted as a PAS submitter. JTC1 has a list of criteria which the organisation should be able to demonstrate in order to become a PAS submitter [ISO/IEC JTC1 N5746]. Given prominence in the criteria is the ability to be able to demonstrate the consensus processes used in the development of the specification and to comply with ISO/IEC patent policy with regard to the specification.

Sun Microsystems and the PAS scheme

In 1997 Sun Microsystems applied to JTC1 to become a PAS submitter in order to standardise the programming language Java by submitting it directly to JTC1 under the PAS scheme. This was a lengthy process as it was unusual for a PAS submitter to be a profit-making organisation. Previously PAS status had been granted to consortia like the OMG. The reaction to Sun becoming a PAS submitter was mixed with, for example, the Danish National Standards Body objecting on the grounds that Sun did not adhere to consensus and openness. [Rada 1998], also, Microsoft as a member of JTC1 Technical Advisory Group (TAG) objected to Sun Microsystems becoming a PAS submitter on the grounds that firstly, it was inappropriate for a single company to be a PAS submitter, secondly that Sun did not appear willing to assign the necessary intellectual property rights in Java to ISO and thirdly that Sun did not appear willing to allow others to contribute to the on going development of Java. Microsoft's comments were typical of the comments also received from other national bodies who were members of the JTC1 Technical Advisory Group [Spring & Weiss 1997]. However, after voting rounds, eventually Sun Microsystems was accepted as a PAS submitter and started the process of standardising Java. However Sun Microsystems later objected to JTC1 changing some of the PAS regulations, which Sun Microsystems did not agree with. Sun Microsystems then took the JAVA out of the PAS submission and applied to ECMA where, after going through a standardisation process at ECMA, it could be fast tracked to become an ISO standard. Further details on the Sun PAS submission can be found in [Mähönen 2000], [Spring & Weiss 1997], [Cargill 1997,] [O'Gara 1997],[Rada 1998]. Mähönen [op cit] points out that an ISO/IEC JTC1 working group could probably have produced a draft standard in two years whereas an attempt to standardise Java through the PAS scheme had already taken over two years. In December 1999 Sun Microsystems then withdrew the Java standard from ECMA. The general view from ECMA was that Sun Microsystems did not want to give up control of Java and ECMA also heavily criticised Sun Microsystems for wasting a considerable amount of ECMA's resources. [Rohde & Niccolai 2000] Although Sun Microsystems wanted an open international standard for Java in order to protect the source code from violations which would hinder its interoperability, it appears that they were not prepared to give up their intellectual property rights on the language. In March 2000 Sun proposed its own development process called Java Community Process (JCP) which it states is an open process which will aim to ensure the compatibility of Java and its continued development. Therefore after abortive attempts to formalise Java as a de jure standard they have now decided to control its standardisation themselves. Sun Microsystems' problems with standardising Java highlight some of the difficulties and complexities of the standardisation process. It is not an easy process to take an existing de facto standard which is sponsored by a for-profit organisation and formalise it as a de jure standard.

Chapter Review

The formal standards organisations develop standards using a consensus approach however, because of the discussions and compromises that have to be adopted in a consensus approach where the members are contributing from countries all over the world, the formal standards development process can be lengthy. Although the ISO/IEC now circulates documents to its members electronically the face-to-face meeting of members is still the main way of resolving disagreements. From the mid 1980's other groups of consortia have formed to produce important IT standards more quickly than the formal bodies. The rapid pace of change in the computing industry often cannot wait for the lengthy consensus approach of the formal standards bodies. Recognising this problem the formal standards bodies have introduced measures to speed up the process of producing standards, the most radical being the JTC1 PAS scheme. With the acceptance of Sun Microsystems as a PAS submitter for Java it appeared that the PAS scheme had opened the way for non-consensus standards from for-profit organisations becoming international de jure standards. However, Sun could not agree on the PAS conditions. Sun's status as a PAS submitter to JTC1 lapsed in November 1999 and Sun has not renewed it. A further attempt by Sun Microsystems to standardise Java through ECMA was also aborted. At the time of writing, September 2001, there is still no international standard for Java. At the moment the PAS scheme remains useful for consortia such as the OMG to fast track their standards.

Since 1989 many authors have suggested that the formal SDO's should try to improve their processes to attempt to speed up the development of international standards for Information Technology [Cargill 1989],[Rada 1995] [Moad 1990] [Rankine 1995], [Gibson 1995], [Rada 2000],[Spring et al 1995]. The PAS fast tracking system was introduced as a structure to speed up the standards development process. It appears to be useful for 'proof of technology' consortia to submit their de facto standards to the formal bodies for standardisation, as the OMG has done, but the only 'for profit' organisation so far to use the PAS system has withdrawn from the process. The ISO/IEC have encouraged links with consortia through the PAS scheme in an attempt to speed up standards development. JTC1 now has an agreement with W3C to encourage the speedy adoption of W3C standards as international standards. This is a case where ISO/IEC JTC1 is "rubber-stamping" standards produced in a consortia. Therefore the role of the ISO and IEC with regard to Information Technology standards is changing as they are now accepting more and more standards which have been produced by consortia. The ISO/IEC IRDS standard, which is the subject of the case study in this research, is a de jure standard developed in the ISO/IEC JTC1 subcommittee 32 'Data Management and Interchange'. The next chapter examines the evolution of the meta data concept which was the foundation of the ISO/IEC IRDS standard.

CHAPTER THREE: THE ISO/IEC IRDS STANDARD

This chapter begins with a description of the software crisis of the 1960's and describes how in the 1960's there arose a need for tools and techniques which could help to facilitate and increase the speed of development of software. The chapter describes the development of the meta-data concept which was the foundation of the early data dictionaries. The early data dictionaries mirrored a parallel development in relational databases with the development of relational database catalogues which eventually led to the concept of a repository. The chapter describes the architecture and the functionality of the ISO/IEC IRDS standard and makes comparisons between the ISO/IEC IRDS standard and the ANSI IRDS standard.

The Software Crisis

In the 1960's a crisis in the production of software arose because of the increased demand for larger and better software systems which could not be met by the suppliers of software. The prices of computers were falling whilst their power was increasing. This resulted in more people being able to afford computers and those computers were capable of producing more sophisticated applications. Problems arose with the building of large software systems. Systems were often late, far exceeded the original budget and did not perform as had been expected. It soon became obvious that techniques used to design and build smaller systems could not be scaled up and used for larger systems. A universally accepted method for Software Development at the time was the "Waterfall Model". This was originally intended to be used where the implementation part of the software lifecycle was completed in third generation language code. There were also problems in managing this coding side of the life cycle. F Brooks, [1975] describes the problems of working in and managing teams of programmers. Brooks felt strongly that if software were late, adding extra manpower to the project would make it even later. He put forward suggestions for different types of programming teams, which would be effective. The overall opinion being that a small team, suitably arranged and managed is always better than a large team where excessive amounts of time have to be spent on communication between the team members. Even though the implementation and coding side of the traditional Software Life Cycle is only one part of the cycle, there were, in the late 60's and 70's, considerable problems in managing programming teams in large software projects. There were also problems at the time with users being presented with completed systems which they were not entirely happy with. Communication between technical Systems Analysis staff and users was often limited. Users were often presented with a system which they had had little say in the development of, and, when the system arrived it was often out-of-date as the users requirements had changed.

In the meantime demand for software was growing. A backlog of applications waiting to be developed was growing with many companies saying they had waiting lists of several years for systems. In this environment tools which could claim to facilitate or automate any part of the software development life cycle, would be very welcome. Data Dictionaries were one way forward to try to control the development and management of software. This chapter traces the evolution of the Information Resource Dictionary System Standard starting with the first meta-data concepts which eventually lead to the development of repositories. An IRDS, repositories and data dictionaries are all concerned with meta data, which is generally described as 'data about data'.

Development of the Meta Data Concept

In the first half of the 1960s COBOL copy statements were used to provide data definitions that could be referenced by different applications [Gillenson & Frost 1993]. These data definitions were one of the first types of meta data, as they were 'data about data' which could be used by a variety of different applications. Also program and subroutine libraries could be considered an early type of Information Resource Dictionary [Parker 1992a] as they encouraged re-use and the standard naming of some elements of data. As a follow on from these, data dictionaries eventually developed. In the same way as a spoken language requires a dictionary to confirm the spelling and meaning of words - otherwise a proliferation of homonyms and synonyms would give rise to a variety of mis-understandings, an organisation will require a dictionary for its data. An organisation can use a dictionary as an authoritative list to consult for its data. All documents would use the same name to refer to the same piece of data, the dictionary can be consulted to check for naming conventions and other information stored in the data dictionary can be reused. The first data dictionaries were sets of files and tools which helped in Information Systems Development. These dictionaries were mainly passive and used as documentation tools for Information Systems personnel. Wertz [1989] and Durell [1985] give comprehensive coverage of the requirements, functional aspects and political and personal implications of using a data dictionary in the software development lifecycle. Three of the popular data dictionaries in the late 1980s were Cullinet Integrated Data Dictionary, IBM DB/DC Data Dictionary, and Manager Software Product's Datamanager [Wertz 1989]. However these dictionaries could not share any meta data between them and it was difficult for a database administrator to decide on which would be the best data dictionary to choose as there were no current evaluation criteria for dictionaries/repositories. [Sankar 1991]

Leong-Hong and Plagman [1982] demonstrated how a data dictionary could be used in the different phases of the software development life cycle (SDLC). The different uses of a dictionary were for documentation support, design aid, metadata generation and change control. They use the term data dictionary/directory system and use the following definition.

> "A Data Dictionary/Directory System is a system that is designed to support comprehensively the logical centralisation of data about data (metadata)." [op cit p16.]

They define metadata in two different categories. Firstly what the data is or what it means – the dictionary. And secondly where the data can be found and how it can be accessed – the directory. At this time the dictionary was considered mainly for use in the SDLC and as a tool for system planning and not for wider information resource management.

Aston University

Illustration removed for copyright restrictions

Figure 3.1 Use of a Data Dictionary/Directory System during the System Development Life Cycle. 1 = Documentation Support, 2 = Design Aid, 3 = metadata generation, 4 = change control. (Taken from Leong-Hong & Plagman [1982] p 27.)

The Relational Catalogue

Gillenson and Frost writing in 1993 traced the evolution of the meta-data concept as it emerged in the first data dictionaries. They divide the evolution of the data dictionary into the stages of passive data dictionary, active data dictionary, relational catalogue, hybrid relational data dictionary, IRDS, repository and finally the Object Oriented Database Management System (OODBMS). The passive data dictionary was a dictionary which stored the meta data separately from the data and there was no link between the two. With a passive data dictionary any conceptual changes made to the data had to be added manually to the data dictionary, whereas the active data dictionary had a run time link to the underlying data and conceptual changes made to the data were automatically made in the meta data store.

Relational databases began to gain in popularity in the early 1980s and the relational catalogue was an integral part of relational database management systems. One of Codd's [1982] original rules for relational databases was that they should have a central catalogue or data dictionary which stored the meta data of the underlying database and that this dictionary/catalogue should be self describing. By self-describing it was meant that the dictionary/catalogue could be queried using the usual query language of the DBMS. For example a relational database which consists of sets of tables populated with rows would also have dictionary tables which store information about the tables themselves - for example who owns the table, the names of the columns and the names of the constraints in the tables. The user should be able to query the catalogue/dictionary tables in the same way as they query the implementational data they have access to - usually using the standard query language SQL. For example a relational DBMS user who needs to list the names of all the tables s/he has created would be able to issue the statement: "Select table_name from user_tables". This is a meta-data query which is extracting a list of tables created by that particular user from the central dictionary and SQL is used for the query. The ISO standard for SQL [ISO 9075] has a core set of dictionary meta-data tables. Using these tables a user can also list the columns in tables, the constraints on columns, the queries which make up views he/she has created and carry out a variety of other metadata queries.

Therefore there were two types of dictionary development, the programming language dictionary which had its origins in COBOL copy statements, followed by the RDBMS dictionary/catalogues. Comparisons can be made between the two as although they developed from different sources they were both concerned with storing meta data which could be centralised and accessed by a variety of users. However the data dictionaries which developed from programming environments usually were designed to store information about the entire information systems operation whereas the relational catalogue was intended to store operational data about the DBMS environment. Some texts refer to the RDBMS dictionary as a catalogue in order to distinguish it from the programming environment data dictionary. However other texts use the terms interchangeably.

Repositories

As the popularity and stability of relational DBMSs continued to grow, the information systems community began to consider extending the RDBMS catalogue to include higher level resource management information as well as operational data about the DBMS. [Navathe & Kerschberg 1986]. Some producers of RDBMS' began to extend their own relational catalogues to include information other than that pertaining to the database and Gillenson & Frost [1993] refer to these as the hybrid relational catalogues.

By the 1980's the proliferation of different computing capabilities had resulted in a large amount of duplicated redundant and inconsistent data in many organisations and the concept of having a repository as a central store of all information pertaining to the organisation was appealing. The new repositories were an expanded form of a data dictionary or relational catalogue that could also include information relating to the planning, analysis, design, construction, and maintenance of systems as well as application level data and meta data. It was also intended that the repository would include other information such as the rules and policies that govern the organisation. [Rosen & Fontaines 1991]. Repositories were an extension of the data dictionary function and were also intended to include the facility for total resource management for the organisation. Whereas the relational catalogue was limited to information and meta data about the database implementation, the repository was intended to cover total resource management as well as the usual dictionary functions. Also, data dictionary schemes were usually fixed and a prerequisite for a repository was to have the facility to extend schemes. This concept was called extensibility. The dictionaries should be able to be extended to add further customised meta entities and meta attributes for the organisations.

In the early 1980's work began within ANSI and The National Bureau of Standards (now the National Institute of Standards and Technology (NIST) on a standard for an Information Resource Dictionary System. (IRDS). In 1988 the ANSI standard X3.38 IRDS was published. [ANSI X3-138-1988] This was later submitted to ISO for consideration as an international standard. The aim of the ANSI standard was to develop standards for dictionary software. The National Bureau of Standards in America concentrated on developing a Federal Information Processing Standard (FIPS) IRDS that could be used for Government-wide programs. [Goldfine & Konig 1988]. At the same time work was started in an ISO sub-committee on an ISO/IEC IRDS.

The IRDS Standard

This section specifically describes the ISO/IEC IRDS standard [ISO/IEC 10027]. The ISO/IEC IRDS family of standards are standards for repositories. The standard was intended to be a framework for a family of international standards, which specify an IRDS. The ISO IRDS Framework standard became an international standard in 1990. It was based on an expanded type of relational catalogue with extra facilities for total information resource management. The purpose of an IRDS as defined in the ISO standard is: -

> "to provide a common basis for the development of Information Resource Dictionaries. An Information Resource Dictionary is a shareable repository for a definition of the information resources relevant to all or part of an enterprise". [ISO/IEC 10027 p3]

IRDS is an acronym for Information Resource Dictionary System. They are in fact repository standards, although the 'R' in IRDS stands for resource and not repository. The standard states that the information resources relevant to the organisation may include computerised and possibly non-computerised processes and information on the organisation of human and physical resources, which can make use of the information. ISO worked on developing a family of IRDS International Standards. There are two main standards in the family. The IRDS Framework standard (ISO/IEC 10027) which became a full standard in 1990 and the IRDS Services Interface standard (ISO/IEC 10728) which became a full standard in 1993.

The ISO/IEC IRDS Framework standard is intended to be a general and broad document. It was intended to be a framework into which other International Standards would slot in order to specify an Information Resource Dictionary System. Its scope was also to define the IRDS interfaces, which are prescribed by other standards in the family. The standard defines the term 'interface' to mean "a set of services made available by a processor". A processor is defined as "an abstract conceptualisation of an executable piece of code". Therefore the standard is a general description of the services which should be provided by an IRDS. The IRDS functionality is described as four data levels. The four levels are called:

a) IRD Definition Schema Level
b) IRD Definition Level
c) IRD Level
d) Application Level

The 'level pairs' referred to in the standard and in Figure 3.1. overleaf are pairs consisting of adjacent data levels. The upper level of the pair will always contain the types for the instances on the lower level. Therefore data on the lower level of any level pair (the instances) is consistent with its schema (the types) at the higher level. There are therefore three level pairs from the four levels illustrated in Figure 3.1. These are; the IRD Definition Level Pair, the IRD Level Pair and the Application Level Pair. It is only the first two level pairs which are the subject of the standard and there is no functionality prescribed for the application level pair (which is the data and meta data usually covered by the traditional relational catalogue). Although the standard points out that there will be other International Standards (for example database languages or programming languages) which do provide functionality which would normally be associated with the application level pair.

The ANSI IRDS standard and the ISO IRDS standard both produced diagrams which illustrated the architecture of a repository. Both ANSI and the IRDS models were similar; showing the same four levels and level pairs. However ANSI used the Entity Relationship Model [Chen 1976] to show a more detailed model of how the structures in the top-level pair linked together whereas ISO used SQL as the data model. ANSI chose the entity-relationship model to define the data model, as the standard was not intended to be implementation specific. In this way the standard could be implemented on a network, hierarchical or relational DBMS. At the time the relational model of data management was still in the early stages in the market place and hierarchical and network model based DBMSs were still heavily used. However, the ISO/IRDS standard has its roots firmly in DBMS relational catalogue evolution and the ISO IRDS standard continued to define the data model in SQL - although the standard claimed that though it was defined in SQL this was not meant to be implementation specific. The IRDS architecture of both ANSI and ISO were similar.

Both the ANSI IRDS and the ISO IRDS have the same levels including broadly the same concepts as can be seen from Figures 3.1 and 3.2.

Rosen [1991] describes an IRDS as an "expanded data dictionary that includes the support of metadata for all System Life Cycle phases of any system in an organisation" [page 6]. This is achieved by using schema layers as in Figure 3.1 and 3.2 . The levels use the concept of types and instances. Using examples from the ISO IRDS, the IRD Definition Schema Level describes the types of object that will make up the first layer. This first layer is static and will contain objects such as meta entity types and meta relationship. Instances of the object types and the associations (relationships or links) between them are then contained in the layer underneath. The layer in turn describes the types of object which may be stored in the level underneath. Each level is therefore

Figure 3.1 - Levels and the IRDS taken from ISO/IRDS standard.

IRD PROCESSES

IRD "DATA" LAYERS

Figure 3.2 Overview of the IRDS Architecture (taken from the ANSI IRDS standard).

linked with another level and these are referred to as "level pairs". The top level prescribes the types of objects used to define dictionaries. It prescribes the types of objects, which have instances or occurrences on the level below. At the IRD level would be specific groupings of entities, relationships and attributes which together model a specific information system. There may then be several IRDSs in the same repository. At the application level real world data is stored with their types described and defined in the level above. Figure 3.3 shows an example from the ISO standard of each object type and an instantiation of the object type at the level below. The IRD Definition Schema Level has just two objects which make up dictionaries, the object types and the associations between them. An instantiation of this in the IRD Definition Level might be 'Table contains columns.' Figure 3.4 shows a similar example taken from the ANSI IRDS standard.

Figure 3.4 Examples of Data and Data Types at Levels (taken from the UK working draft for the IRDS framework revision.) [BSI Project 1.21.6.8 p 30]

Information Resource Management

The ISO IRDS standard document describes a repository as having all the basic facilities of a DBMS as well as other facilities necessary for the Information Resource Management requirements of a repository. The framework standard states that the data in an Information Resource Dictionary is in many ways similar to that in an application database and divides the IRDS facilities into two groups; general database management facilities and facilities specific to information resource management. The extra facilities - to cater for Information Resource Management - are those which can be dealt with mainly at the IRD Definition Level - the meta meta data level. The list of facilities includes such topics as version control and information system lifecycle management. Table 3.1 lists those facilities described in the standard as general database management facilities and table 3.2 lists the extra facilities specific to information resource management. The ISO framework standard states that: "A large part of the task of information resource management is concerned with controlling the development, introduction and use of information processing systems". The facilities outlined for information resource management are intended to help with

Aston University

Content has been removed due to copyright restrictions

Figure 3.5 – Overview of the IRDS Architecture

this task. The IRDS will provide services at the two top-level pairs of the architecture. This will be the IRD Definition Level Pair and the IRD Level Pair.

The application level pair is not considered. Each service will always relate to a level pair (type and instance) and never to a single level. The IRDS Services standard provides more specific detail on the services needed by a processor in order to be able to access and manipulate the IRDS data.

IRDS GENERAL DATABASE MANAGEMENT FACILITIES

Enforcement of Constraints	It should be possible to specify constraints relating to domain integrity, cardinality constraints and constraints on the values associated with objects. Constraint integrity should be enforced but with the option of switching off constraints under certain circumstances.
Access Control	Access should be limited to suitably authorised users. There may also be a provision to limit access to individual objects or attributes within an IRD or an IRD Definition.
Audit Trail	Auditing is regarded as a process of checking that changes made to data have been made correctly and by an appropriate user. It should be possible to audit these changes.
Limits and Defaults	It should be possible to specify limits and defaults for specified attributes. These limits and defaults can then be applied when objects are created or modified.
Database Integrity	Integrity of the data should be supported in a single or multi-user environment (if a multi-user environment is supported). This should include the provision to recover from failure, either program or system failure, by using rollback techniques.
Query and Reporting Facilities	Should be able to query the IRD or IRD definition using general-purpose query and reporting facilities.
Remote Data Access	Users do not necessarily have to be at the same real system as the IRDS. Therefore a distributed IRDS is possible.

Table 3.1 IRDS GENERAL DATABASE MANAGEMENT FACILITIES (summarised from the ISO IRDS Framework)

Table 3.1 above shows the facilities listed in the standard for general database management. These facilities are what would usually be expected in a DBMS with an active catalogue. Table 3.2 shows the extra facilities required for total information resource management.

IRDS FACILITIES SPECIFIC TO INFORMATION RESOURCE MANAGEMENT	
Naming	A facility for the unique naming of objects and for imposing constraints on the naming of objects in order to control homonyms and synonyms.
Status of Dictionary Content	The user should be able to distinguish between data which is no longer used (archived), data which is unstable and data which is stable.
Information System Life Cycle Management	The IRDS should be able to support the idea of phases in a Life Cycle. The life cycle of an object can be split into a number of phases. This is useful for controlling the development and use of information systems.
Version Control	Users should be able to create and maintain versions of objects, or groups of objects and be able to use different versions of an object if required.
Partitioning	For convenience the IRD may be divided into partitions. A partition is a logical subdivision of an IRD or an IRD Definition. The standard states that an object can be in only one partition, but other objects representing different versions of the object may exist in other partitions.
Copy Creation	Users should be able to make copies of an object or group of objects in order to avoid duplication of work.
Impact Analysis	An IRDS should be able to answer queries concerned with impact analysis. Particularly the concepts of dependency of one object on another. If one object is modified you should be able to analyse which other objects are dependent on it and which may be affected.
Subsetting	It should be possible to define subsets of an IRD or an IRD Definition. This is to control user access and provide each user with their own context. Each subset provides update access to one partition only and optional read-only access to other partitions.

Table 3.2 IRDS FACILITIES SPECIFIC TO INFORMATION RESOURCE MANAGEMENT (summarised from the ISO IRDS Framework)

IRDS Naming – The Control of Homonyms and Synonyms

There are obvious advantages in having a central store of the metadata. First of all it can be used as a communication tool for users or programmers to consult - to see what data is in the organisation and how it is named. File and table descriptions can be re-used and in fact should be re-used to ensure that standard naming conventions are used in the organisation. If the repository is going to be a reference source for the Information Systems of an organisation there must be definite naming conventions and strict control of synonyms and homonyms. This is something that can be controlled and documented through the repository.

Version Control, Configuration Management and Impact Analysis

If the aim of the repository were to provide total information resource management including the management of the development of systems it would be necessary to include versioning and configuration management as part of the functionality. Many version control applications would operate using a "check-in" and a "check-out" facility. A user may "check-out" an object and after modifying it check it back in to the repository thus creating a new version of the object. While the object is checked out other readers may only have read-only access to it. Configuration management refers to a logical view of a set of versions. A configuration could be regarded as a group of specific versions of software development objects which are related or which are brought together for a specific reason. The ability to be able to assess the impact of changing one object on another is called impact analysis. The repository software therefore needs to have the capability of assessing dependencies between objects in order to carry out impact analysis.

Information System Life Cycle Management

The life cycle of a system will be broken down into phases. It should be possible to store information on the development of each phase. Combined with the version control facilities, life cycle management would enable different groups of developers to work on different phases and different versions of the system under development. This information could be produced in a form that also provides the documentation for the system.

Active versus Passive Repositories

Once the metadata is collected and centralised controls have to be introduced to ensure that it is kept up-to-date. Every time a change is made, for example, in the underlying database, (for example an extra field is added to a table) this change has to be reflected in the repository/dictionary. This is a metadata issue, adding an extra field to a table changes the conceptual level and this change has to be made in the repository. The possibilities for additions and changes are either a passive or an active repository.

Passive Repository

If the repository is passive there will be no dynamic or run-time link between the repository and the RDBMS catalogue, or, in terms of levels there will be no dynamic link between the IRD level and the application. If the repository is passive it has to be continually reconciled with the underlying database as it does not have a run time or dynamic link to the DBMS. If the repository is working with a relational DBMS, the change will be automatically made in the relational catalogue, and then the change will then also need to be recorded in the repository.

The repository was also intended to document the development of application systems and to control synonyms and homonyms. Therefore strict controls have to be kept on implementing changes. A decision could be made to only allow changes from the conceptual level first, followed by the change at the implementational level, or, changes may be allowed at the implementational level first and followed through with the change at the conceptual level. Nevertheless, whichever is chosen strict change control has to be exercised. A passive repository may suffer from a lack of user commitment to update it unless there are strict procedures to record any changes. With the above example if an extra column is added to a table in the underlying database and if the repository is passive the fact that there is an extra column, its data type and any indexes or constraints on the column has to be manually added to the repository to ensure that it is consistent with the lower application level.

Active Repository

With an active repository a change made in the table structure is automatically carried through to the repository level or conversely a change at the conceptual level would be carried through to the lower levels. The main disadvantage of an active repository is that it will carry with it a larger processing overhead if it is to be on-line all of the time and may slow down performance considerably.

A repository can be described as active if it is used in the run-time environment. Most RDBMS catalogues or dictionaries today are active and when queries are executed will use the catalogue to check on authorisation to the tables, whether the tables exist and whether the required columns exist etc. Also the fact that RDBMSs are non-navigational and queries are in the first instance executed at run time meant that the RDBMS dictionaries had to be part of the run-time environment in order to carry out query processing. The DBMS would use the dictionary or catalogue to determine which columns had indexes, which columns were in which tables, whether the user had authorisation to those tables and columns. As well as being active the DBMS catalogue is also dynamic. When new tables are created they are automatically added to the catalogue along with the column names in the table and any other constraints. In the same way a repository could be described as active if it can be used in a run-time environment.

In the early 1970s there were several passive data dictionaries on the market some of which were dictionaries for specific DBMSs. Today most large commercial RDBMSs will have active system catalogues. An active dictionary involves considerable processor activity and may cause bottlenecks in processing especially in a multi-user environment. However a passive dictionary means that there have to be strict manual procedures to control change and ensure that changes are made to the dictionary and not forgotten about.

Extensibility

It has been recognised that a repository should be able to be extended (customised) by the user. Previous dictionary systems had tended to be inflexible and users could not add their own extensions. Both the ANSI and the ISO IRDS outlined the necessity for a repository to have the capability of extending the object types in the meta model. This would usually mean extending the model at the equivalent of the IRD Definition Level. For example a new object type could be added, or new attributes added to existing object types. Rosen [1991] states that it is essential to have the ability to extend schemas because the volatility of organisations means that change is constant and it would not be practical to have fixed schemas. Also traditional data dictionaries usually supported the system design and operational side of the organisation, however one of the uses of a repository would be to record information on earlier stages of a system lifecycle, for example, requirements analysis, or to store information on business rules. If a company wanted to add some of its functions to the repository and document how the functions react with other instances of entity types, the company could add a new entity-type called functions at the IRD Definition level and then add instances of the entity type at the IRD level. They may also need to add a new relationship type to link the functions to other entity type instances. Without schema extensibility the company would not be able to extend the repository to meet their requirements. Extending the schema would also allow for using different methodologies, which may have differing entity-types and relationship-types. The schema could be extended to accommodate these different methodologies.

Theoretically, a further use for the extensibility feature was to enable the repositories to be extended in order for Computer Aided Software Engineering (CASE) tools to map on to the repository. A variety of CASE tools may by used by an organisation to model and implement application systems at different stages of the software development lifecycle. When a new CASE tool is used the IRD schema can be extended in order for the new tool to map on to the repository thereby creating an integrated environment for application development. In 1990 NIST completed a project which involved transferring data from a CASE tool (KnowledgeWare's Information Engineering Workbench (IEW)) to an IRDS using the IRDS Export/Import File Format Standard which was then under development [Rosen 1991]. The process involved creating an IRD which had a schema which the CASE tool could map to, a program was created which accepted a set of IEW files as input and generated as output an IRDS export/import file format, the export/import file was then entered into the IRD through use of the IRDS import command. This project was successful and demonstrated how having a standard format for import to, and export from, an IRDS decreases the need for proprietary exchange methods being developed by vendors for individual tools exchanging information between them.

Repository Management

However the repository is still only a tool and mainly a communication tool. The successful use of a repository requires a good data administration team and an organisational approach. Stenton [1983] stresses that dictionaries were not a universal panacea and that the "success of a data administration function depends almost entirely upon the quality of the data dictionary and the aptitude of the person in charge of the data administration function". "It is the person who solves and prevents the problems not the dictionary".

The importance of adequate control and organised introduction of the data dictionary by the correct personnel is also emphasised by Wertz[1989] and Durrell[1985]. This is still as true today for repositories, datawarehouses and web meta data. Success depends not just on the quality of the tool but the aptitude of the person controlling the tool.

Conformance and Conformance Testing

On the publication of the IRDS standard the ISO subcommittee did not issue any instructions for conformance to the IRDS standard or conformance testing. The ISO/IEC IRDS standard concluded with the phrase:

> "Conformance requirements are not stated in this International Standard. Rather they are stated in each of the other Standards in the IRDS family of International Standards"[ISO/IEC 10027].

Therefore there was not expected to be any conformance testing after the publication of the IRDS Framework standard. The Services Interface standard which was published three years later contained more specific details about conformance. In 1986 Dolk & Kirsch [1987] implemented a free-standing IRDS model using the Oracle database management system. At the time the ISO IRDS had not been published, however, they implemented a repository with basic functionality using a set of tables which were similar to the tables described in the later IRDS Services Interface standard. The purpose of their work was to show how a repository could be implemented as a layer above the DBMS. The work concluded that it was possible to build repository functionality using SQL data definition language to define meta-meta data tables which would run with an underlying DBMS. Research was undertaken in 1992 [Casanova et al 1992] on the implementation of a standard IRDS repository. In this research an IRDS repository was implemented based on the ANSI IRDS model and the services interface to this IRDS was based on the ISO services interface (as at the time the ANSI services interface was not developed). One of the objectives of this research was to gather preliminary knowledge about the feasibility of software engineering environments based on IRDS repositories. The research made a distinction between the IPSE (Integrated Project Support Environment) of which PCTE was an example and the Information Engineering approach to integrating tools through a repository environment. They supported the idea that the IPSE approach was popular in software engineering environments and the repository approach in environments which dealt with database management systems. The research produced some positive results on the performance of a standard IRDS (which was linked to a well known database management system) in a research environment and was not intended to be a commercial product.

Chapter Review

A repository is a natural progression from the DBMS active catalogue. A repository system is a logically centralised store of data about data (metadata) and includes facilities for total information resource management within the organisation. A repository can be described as a specialised database of information which describes the characteristics of the data used to design, monitor, document, protect, and control the information resources of the organisation. As well as having a meta data role the dictionary is expanded to include information about the planning, analysis, design and maintenance of systems and the dictionary should be able to be extended, if necessary, by the user. A repository can therefore be described as a meta-metadatabase. The organisation will have to consider and plan exactly what it will need in the repository and recognise that the repository is a tool which requires suitably qualified personnel to administer it. After the publication of the ANSI and the ISO IRDS standards the word 'repository' came to be used for the type of extended data dictionary which was described in the ISO/IEC IRDS standard.

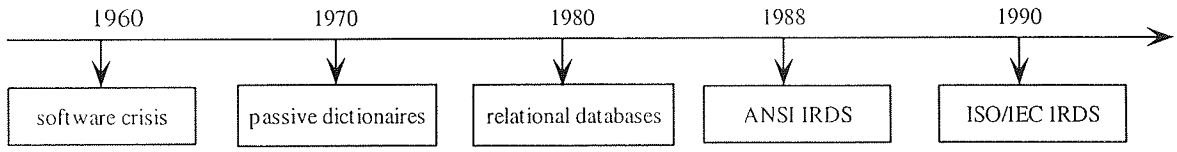

Figure 3.6 Timeline from 1960-1990

This chapter has outlined the main sections of the IRDS Framework Standard in order to provide an overview of the content of the standard and the functionality described for the standard. The IRDS framework standard was the guiding standard and the Services Interface standard which followed, covered implementational detail by outlining a set of SQL tables which could provide the basis for a repository. The Services Interface standard is the subject of the next chapter. Within the ISO/IEC sub-committee work was ongoing to complete content modules for the IRDS standard and an export/import facility were also under development. There were two versions of an IRDS standard; the ISO/IEC IRDS standard and the ANSI IRDS standard. They were both similar in architecture and functionality and it was intended that the ANSI standard would be presented to ISO and that the two standards would eventually converge.

CHAPTER FOUR: THE IRDS SERVICES INTERFACE STANDARD

The IRDS Framework Standard became a full international standard in 1990. From 1990 onwards work continued in the relevant ISO subcommittee, and in the committees of national bodies, on the IRDS family of standards. The next standard in the planned family of standards was the IRDS Services Interface Standard which became a full standard in 1993.

The purpose of the Services Interface Standard is to describe the services on the IRD level and the IRD Definition Level in the IRDS standard and to provide all the services needed by any processor wishing to access IRDS data. The IRDS Framework standard, which was the subject of the previous chapter, specifies the context within which the Services Interface standard is defined. The services interface describes the facilities provided at the two top-level pairs - The IRD definition level and the IRD level (as described in Figure 3.2). The facilities are supported by data which is recorded in 27 IRD Definition tables. The first part of the standard outlines the 27 tables and their constraints. The tables are described formally using standard SQL. The tables are similar to the standard SQL dictionary tables but with additions to cope with the functionality of the extra Information Resource Management requirements. The standard states that it should also be possible to extend the dictionary at the IRD Definition Level. The second part of the standard describes the services necessary. The format of the services is described using the ISO Pascal programming language.

Data Structures for the Services Interface

In the standard the data structures are described using a set of tables. The tables are divided into four sections; the internal tables, the environment tables, IRD specific tables and common tables. The full list of tables appear below in table 4.1.

Internal Tables

The internal tables are mainly intended for object management and version control. The internal tables will exist for each IRD Definition and IRD. Table 1, IRD_Object, contains a row for each set of rows in table 3, IRD_Object_Version, which represents different versions of the same object. Table 2 contains a row for each working set defined within an IRD Definition. A working set is explained as a particular area a user may be developing, for example a screen layout, a database schema or a business model. Working sets are sections which a user will work with and

Internal Tables:	
1	IRD_OBJECT
2	IRD_WORKING_SET
3	IRD_OBJECT_VERSION
Environment Tables:	
4	IRDS_USER
5	IMP_LIMITS
IRD_specific tables:	
6	IRD_SCHEMA_GROUP
7	IRD_SCHEMA
8	IRD_SCHEMA_REFERENCE
9	IRD_DATA_TYPE_DESCRIPTOR
10	IRD_DOMAIN
11	IRD_TABLE
12	IRD_VIEW
13	IRD_COLUMN
14	IRD_VIEW_TABLE_USAGE
15	IRD_VIEW_COLUMN_USAGE
16	IRD_TABLE_CONSTRAINT
17	IRD_KEY_COLUMN_USAGE
18	IRD_REF_CONSTRAINT
19	IRD_CHECK_CONSTRAINT
20	IRD_CHECK_TABLE_USAGE
21	IRD_CHECK_COLUMN_USAGE
22	IRD_ASSERTION
Common Tables:	
23	IRD_MODULE
24	IRD_CONTENT_STATUS
25	INSTALLATION_DEFAULT
26	IRD_WORKING_SET_PRIVILEGE
27	IRD_REFERENCE_PATH

Table 4.1 The Services Interface Tables there are facilities described for sharing objects between working sets using reference paths. Facilities are also described for version control of these working sets and for coping with objects referenced between different versions of working sets. The set of objects which make up, for example, a business model, needs to be managed as a whole. The sets of objects which the user needs to handle as a whole unit for the purposes of change management or version control are called working sets. The tables which deal with version control are the IRD_Working_Set (table 2), the IRD_Working_Set -_Privilege (table 26) and the IRD_Reference_Path. (table 27). A working set contains only one version of each of the objects it contains. Working sets can be based on other working sets and references can be made from one working set to another.

Environment Tables

The environment tables exist once in each IRD Definition. There will be a row in table 4, IRDS_USER, for each user who has privileges to access any IRD Definition. Table 5, IMP_LIMITS, specifies implementation limits on the maximum lengths of text values, integers and IRDS names. This table is defined by the implementers and used only for control purposes and is not directly linked to any other table.

IRD Specific Tables

The IRD Specific Tables, tables 6-22, exist only in the IRD Definition or a specific IRD. They are based on part of the SQL Definition Schema (ISO 9075), but with some extra columns in the tables which are necessary to support IRDS facilities such as naming.

Common Tables

The tables IRD Module, IRD Content Status and Installation Default are to support defaults and extra facilities. Tables 26 IRD_WORKING_SET_PRIVILEGE and 27 IRD_REFERENCE_PATH are concerned with version control and configuration management.

Structure Diagrams

The standard also includes structure diagrams to illustrate parts of the IRDS. (The structure diagrams are similar to Entity-Relationship Diagrams). Figure 4.1 below shows a structure diagram which illustrates the links between the tables which deal with version control issues.

In Figure 4.1 an IRD table will be made up of many IRD objects. (A row in a table is referred to as an object). Each object can have many different versions, referred to in the IRD object version table. A working set is then made up of many object versions. As it may not always be practical

Aston University

Content has been removed for copyright reasons

Figure 4.1 Tables illustrating version control links taken from IRDS Services Interface Standard

to create a new working set for every object level change the standard states the granularity of version control can be under the control of the IRDS user. A working set may be based on another working set. The new working set will initially be a copy of the working set it is based on. This is shown by the recursive relationship on the working set entity in figure 4.1. The working set privilege entity stores information on the privileges granted to specific IRDS uses to access specific working sets.

The reference path facility is to allow object versions in one working set to reference object versions in other working sets. Reference paths are only allowed from a working set to other working sets which are not within its version path. This is useful for when different development teams may need to see work in other working sets. The IRD Content Status refers to whether the working set is Uncontrolled, controlled or archived. Changes can only be made to working sets which have a status of Uncontrolled. The structure diagrams map to relational tables using a generic Entity-Relationship to relational mapping algorithm [Elmasri & Navathe 2000]. When mapped to tables, referential integrity and entity integrity are supported, although some of the tables result in long composite keys.

Figure 4.2. shows an example of user-defined tables which are sub-tables of the IRD Object Version table. The rectangles within the main object version rectangle represent sub-types or sub-tables of the main table object version. The user has recorded as object types the objects, System, Program, File, and Record Type and the referential constraints between the objects. There are also links between the sub-tables, for example a one to many relationship between program and program statement, and a many to many relationship between program and file. The sub-tables within the object_version table are the tables which will have been added at the IRD Definition Level. At this level the user could extend the IRD by adding their own customised objects. The top part of figure 4.2 deals with version control, working sets and IRD tables at the IRD Definition Level, whereas the tables within the Object_Version table represent tables at the IRD level.

Table Definitions

The structure diagrams can be mapped directly to tables and each of the 27 tables is defined with SQL data definition language with referential constraints and domain constraints. Also a verbal description of each table is included in the standard. The first table in the list, IRD_Object, is reproduced in table 4.2.

Figure 4.2 Version Control and User Defined Tables taken from the ISO IRDS Services Interface Standard

(The top part of the diagram deals with version control facilities and the object version table in the lower part of the diagram contains many sub-type tables within it.)

Table 4.2 Definition of the IRD_Object Table taken from the ISO 10728

A domain that contains all valid IRDS Keys has previously been defined with a create domain statement. Audit attributes are also added to each IRD Definition or IRD table. These include, the name of the user adding the object version, and date and time when the object was added and the information added. Object versions include the name of the user modifying the object version and the data and time when the object version was last modified.

Therefore the Services Interface provides a complete data model for Information Resource Management using SQL as the defining language. The IRD specific tables (tables numbers 6-22) are similar to the standard SQL*92 tables for an information schema. This is not surprising as part of the functionality of a repository was to provide general DBMS facilities as well as the extra facilities required for Information Resource Management. However, the standard states that "it makes no assumptions about an implementation environment, and assumes so specific run-time or compile-time interfaces." So although the model is described in a data model which uses SQL the use of SQL is not a necessary part of the implementation.

Services Provided

After defining the data structures in SQL the standard then describes the services which should be provided. These are divided into three groups; Operational Services, Level Independent Services and Level Specific Services.

Operational Services (used by the client to initiate and terminate processing and transactions)	**Level Independent Services (for manipulating data in the IRD or in the IRD Definition)**	**IRD Definition Level Specific Services**
Create an IRD Definition	Set Context Service (sets a context for future requests within the session)	Create IRD Service
Drop an IRD Definition	Add Object Service (adds an object version within the current working set context)	Drop IRD Service
Open IRDS Service	Open Cursor Service	Reactivate IRD Service
Prepare Service (prepare the current session to commit)	Retrieve Object Service	Deactivate IRD Service
Commit Service	Modify Object Service	Validate IRD Schema Group Service
Rollback (rollbacks any updates made as part of the current transaction)	Delete Object Service	
Close IRDS Service	Close Cursor Service	
Get Diagnostic Service	Create Working Set Service	
	Drop Working Set Service	
	Modify Content Status Service	

Table 4.3 Summary of the services for each category

For each service specified information is also supplied on the service's function, format, input, output, any general rules and any operations on the Abstract Data Structures (the tables). The format of each service is specified using the programming language ISO Pascal. The full information for the "Create IRD Definition Service" is provided in figure 4.4 below.

Aston University

Content has been removed due to copyright restrictions

Table 4.4 Example of the full description of a services interface service taken from ISO IRDS Services Interface Standard.

Each service in the standard is described using the format outlined above in table 4.4.

Conformance to The Services Interface Standard

The services interface standard stated that in order for a product to claim conformance it must firstly conform to the semantics of the data structures and rules which were outlined in the standard. This refers to the structure of the tables and the entity integrity, referential integrity and domain integrity rules which the tables supported. Secondly it must conform to the IRDS service formats and descriptions which are summarised in table 4.4 above. However the standard also continues to say that although the data structures are expressed in SQL there is no obligation for a product to use SQL and likewise although the service formats and descriptions are in the programming language Pascal a product which is claiming conformance need not support Pascal as long as it supports another ISO standard programming language. As far as the author is aware there were no plans for conformance testing.

The IRDS Family of Standards

From 1990 onwards work continued in the ISO subcommittee and in the committees of national bodies on the IRDS family of standards. The Services Interface standard was published in 1993 and by 1996 the following were published or in the process of being developed:

Published by 1996

ISO/IEC 10027:1990 IRDS Framework

ISO/IEC 10728:1993 IRDS Services Interface

ISO/IEC 10728:1993 Amendment 1 1996 IRDS Services Interface C Language Binding

ISO/IEC 10728:1993 Amendment 2 1996 IRDS Services Interface Ada Language Binding

Draft International Standards or Draft Amendments in 1996

ISO/IEC 10728:1993 Draft Amendment 3 1996 IRDS Services Interface CORBA IDL Binding

ISO/IEC 13238-3 Export/Import Facility

Committee Drafts or proposed draft amendments in 1996

IRDS Content Module: Naming & Thesaurus

Guidelines for the Design of IRDS Content Modules

RPC IDL Binding . Proposed Draft Amendment to IRDS Services Interface.

Working Drafts in 1996:

IRDS Framework Revision. This was a revised version of the original IRDS standard.

IRDS Services Interface Extensions

IRDS Content Module for Data Flow Diagrams.

A content model is a model of information requirements in a particular subject area and the first step towards defining an IRDS Content Module which will be a set of definitions prepared for use with an IRDS. Firstly a model of the subject area is necessary and this is known as a Content Model. The IRDS Content Module is a set of definitions based on the content model which is prepared for use with an IRDS. Some of the definitions can be stored in the IRD Definition Level or at the IRD Level. In 1996 there were committee drafts for a standard set of guidelines for developing IRDS Content Modules and a committee draft for a standard Naming and Thesaurus Content Module. Also in 1996 there was a working draft for a Content Module for data flow diagrams. It was envisaged that in some subject areas there would be a need for a family of Content Modules [CD 13645.2]. It would be expected that content modules would be developed in a wide range of areas for example: business modelling, manual systems modelling (forms, procedures), computer systems modelling and generic facilities (for example naming and thesaurus facilities and security control). A language binding illustrated a mapping of the abstract data structures and services to a certain programming language. Amendment 1 and 2 to the Services Interface standard demonstrated the abstract data structures in C and Ada respectively. The initial Services Interface Standard used the programming language Pascal as an example of binding the data structures to a programming language.

Chapter Review

The Services Interface Standard was published three years after the IRDS standard. It provided implementational detail in the form of definite data structures and it outlined the services which an IRDS should support. The data model used was SQL and there was an overlap between the tables defined in the SQL standard for database administration and the tables defined in the services interface standard. Work continued in the relevant sub-committee of JTC1 on various content modules for the IRDS family of standards. There were some conformance statements in the Services Interface Standard but no available conformance testing procedures.

CHAPTER FIVE: BACKGROUND TO THE IRDS STANDARD

In order to place the ISO/IEC standard in context this chapter describes and discusses other standards and products which were emerging in the late 1980s and early 1990s. The chapter begins with the rise of Fourth Generation Languages and then the subsequent development of Integrated and Component CASE tools. The chapter also examines two other repository products which entered the arena in the early 1990's. The Application Development Cycle (AD/Cycle) sponsored by IBM and the Portable Common Tool Environment (PCTE).

Fourth Generation Languages

During the late 1970's and early 1980's Fourth Generation Languages (4GLs) were given much attention and hype in the computing press and by computer vendors. They were seen as one of the solutions to the problems of software development mentioned in the previous chapter. Theoretically with 4GLs users could develop their own systems. It was claimed that it was less demanding to develop systems with 4GL's than with third generation languages (3GL's) because of the non-procedural nature of most 4GL's. This also implied that end-users could develop their own systems and thereby help to reduce the backlog of applications waiting to be developed. By completing part of the implementation side of the Software Development Life Cycle with 4GLs, application backlogs - which many organisations claimed to have - would be drastically reduced. The term 4GL was based on previous generations of computer languages. The first generation was machine code, which is based on binary digits. For a programmer to work in machine code will be very tedious, time consuming and difficult and the programmer will also need to have an intimate knowledge of the hardware. Second generation was Assembler which used mnemonics, which are then translated into, usually, one machine code instruction. So instead of having to write in binary the programmer can write words like ADD, MOVE, etc. Although easier to use than machine code the assembler language is still difficult and tedious to use. With third generation languages, for example Cobol, Pascal and 'C' one statement can be translated into several machine code instructions. Third generation languages, similar to assembler, use natural English words and are easier to use than previous generations as less knowledge of the hardware is required. They use compilers, which translate the program into machine code. However these third generation languages still require the programmer to have undergone considerable training and therefore they require trained computer professionals to use them. It was claimed that Fourth Generation Languages theoretically do not require traditional programming skills, and, as a follow on from this that they do not need computer professionals to develop systems[Martin 1986] . Fourth generation languages were largely non-procedural in nature whereas first, second and third generation computer languages are all procedural in nature. That is, the programmer has to program from the point of view of how something has to be done to achieve a result. Whereas the 4GL programmer can tell the system what he wants and let the system deal with now to get the result. To explain the need for 4GLs Martin [1986] uses the analogy of the American Telephone Industry before the First World War where the telephone industry was growing furiously but each telephone call had to be switched manually. However, as the user group increased and the distance of the calls increased several manual switches were required to switch one call. It was calculated that if the telephone industry were to carry on growing at the same rate it would require the entire work force of the USA to switch the calls. The telephone industry's solution to this was the invention of an automatic switching device. To transfer the analogy to the computing industry the solution was the invention of 4GLs. Tools were needed which could automatically generate third generation language code.

When the first 4GLs were developed many in the computing press made claims about the productivity increases that 4GLs would bring to systems development. The fact that they were even called 4GLs gave the impression that they would supersede or replace the older generation of procedural third generation languages. Vendors began to describe software, which exhibited some non-procedural features, as 4GL software. In fact the term 4GL was a misnomer as 4GLs were not really languages in the same sense as previous generations. Other terms which came to be used for 4GLs were Application Generators, Fourth Generation Tools, Fourth Generation Systems and Fourth Generation Environments. Fourth Generation System or Environment was often used to describe a system based round a relational database. [Holloway 1990] The system would often incorporate a non-procedural query language, reportwriter, and screen generator and each of these would be called a 4GL tool. One definition of a 4GL is that it is a tool which could be used by end-users to develop their own systems quickly. However, more often, 4GL tools proved useful tools for use by programmers in conjunction with 3GLs. Also 4GLs, because of their non-procedural nature and ability to quickly generate screens, have proved useful tools for prototyping. Often, prototypes can be put together on the desktop, with the user actively and immediately involved.

Fourth generation languages, although successful in reducing the application backlog, did not achieve the success which was originally hoped for [Baer 1992, Glass 1991]. One of the reasons given for the limited success of 4GLs was that there was no 'standard' 4GL but a proliferation of proprietary systems which were not portable to other systems. Also, although mainly non-procedural they could often have a steep learning curve. The basics of an application could often be produced quickly but anything complex needed considerable training or even 'reverting' to a 3GL to code in any complex issues. Also, using a 4GL did not give the same job satisfaction as 3GL coding to many programmers and the use of 4GLS did bring with it another set of problems. If end-users were going to develop their own systems this introduced the problem of ensuring that adequate documentation on the system was available in order to maintain it and also that the quality assurance guidelines of the company would be followed. Although the evaluation of quality assurance and the production of documentation would have procedures to follow within a data processing department these would be more difficult to enforce company wide. However, 4GLS did prove useful to speed up programmer productivity and have proved excellent tools for prototyping. Fourth Generation languages now form the basis of many database and DBMS programs and are used extensively to create a forms based interface to a database and to produce reporting facilities.

By the late 1980s the term 4GL was also used to refer to a wide variety of products. Some 4GLs were attached to an underlying database for example the Oracle RDBMS used SQL*Forms, the Ingres DBMS used Ingres Windows 4GL. In a database environment 4GLs would use the meta data from the relational catalogue to help produce the information on the forms and reports. Some microcomputer database packages called themselves 4GLS (incorporating the database with the 4GL tools) for example Paradox and Progress. Some 4GLs were DBMS independent for example Uniface, and Focus. Therefore it was also difficult to define exactly what was meant by a 4GL. However, one aspect all the 4GL products had in common was that they dealt mainly with the implementational phase of the Software Life Cycle.

Computer Aided Software Engineering (CASE)

Computer Aided Software Engineering (CASE) tools began to gain in popularity in the early 1990's. By 1993 CASE tools were also being called Computer Aided Systems Engineering tools. The emphasis was changing from automating the coding of software to automating the whole of the systems/software lifecycle. A well-known research company came to the conclusion that "CASE will eventually make fourth generation languages redundant"[Kavanagh 1991]. Failing to see that 4GLs were in fact the forerunners to CASE tools as they were the first tools to automate part of the software development life cycle.

CASE tools could automate parts of, or in some cases all of, the Software Development Lifecycle - not just the implementation phase as was the case with 4GLS. Definitions of CASE tools were; Integrated CASE tools (I-CASE) and Component CASE tools (C-CASE). I-CASE Tools provide automation for the whole of, or a major part of, the SDLC and are usually the product of one vendor. I-CASE tools will usually have a central repository where all information on the phases of development and versions of applications will be held. Component CASE tools should work with other CASE tools in an integrated environment. They will help in a particular stage of the SDLC. The term "Upper CASE" tools was used for tools which deal with Project Management and Design and Analysis phases of the software development lifecycle. The term "Lower CASE" for tools which deal with coding and implementation. Also these were sometimes referred to as front-end tools (Upper CASE) and back end (Lower CASE) tools. Although the definitions for types of Case tools were very fluid – the author has also seen the term I-CASE used to refer to Component CASE tools working in an integrated environment [Rosen 1991]. Using the above definitions a 4GL could be described as lower CASE, component CASE or as a back end CASE tool.

The I-CASE tools supported a methodology which was often vendor specific. For example the Oracle*CASE tool used a methodology called CASE*Method which was broadly similar to Yourdon and SSADM. [Barker 1990] Information Engineering Facility (IEF) from Texas Instruments used the Information Engineering Methodology.

For an I-CASE tool to aid in automating the stages of the software development lifecycle it follows that information on the various stages has to be documented. The Oracle CASE tool is an example of an I-CASE tool. It consists of a tool called CASE*Dictionary which is a repository which records information about the stages of the SDLC. The tool has a suite of diagrams which aid in the analysis and design of the system and any information entered on diagrams is automatically entered into the CASE*Dictionary. The Oracle*CASE tool had a suite of diagrammers which included an Entity Relationship Diagrammer, a Data Flow Diagrammer, a Function Hierarchy Diagrammer and a Matrix Diagrammer. The Matrix Diagrammer allowed information from two of the other diagramming tools to be put on a matrix for further analysis. More specific information can then be entered through CASE*Dictionary and forms, modules and programs can be automatically generated with the CASE*Generators. In this way the CASE*Generators incorporated the 4GL technology into the larger I-CASE tool. Data Definition Language statements can also be automatically generated from the mapping of the Entity Relationship diagrams to tables. The necessary tables, views and indexes can then be generated in the underlying Oracle DBMS. The core of the Oracle CASE product is the CASE*Dictionary which is a repository for all information pertaining to the system's (or indeed several systems) under development or completed systems with documentation.

Repositories and CASE tools.

By the early 1990's it was thought that a user should be able to, theoretically, select a variety of C-CASE tools from different vendors to help in various stages of the SDLC, for example a diagramming tool, a 4GL screen generator etc. Many conferences included the topic of the interoperability of heterogeneous tools. However, in order for these tools to work in an integrated environment standards will be necessary for a central dictionary/repository and for the services and facilities of the repository.

Large Integrated CASE (I-CASE) tools were often referred to as tools which covered all, or a major part of, the software development life cycle and were linked to one vendor (for example IEF from Texas Instruments and Oracle*CASE form Oracle Corporation). Component case tools were usually referred as a collection of tools from different vendors and these tools would help in particular stages of the SDLC. The I-CASE tools had their own central repositories - the

Encyclopaedia in IEF and the Oracle*CASE Dictionary in Oracle*CASE. If component CASE tools need to work in an integrated environment it would be necessary to access a common repository of information in that environment. The situation is similar for component-based software engineering. Components, whether developed by CASE tools or not, cannot operate on their own but will need to link into a central repository in order to exchange information between other components, to have their meta data stored in the repository for reuse, to store information on different versions of components, to control unique naming of objects and to be able to answer impact analysis questions. [Byrne & Golder 1996] The alternative, of not having a repository, will be that there are either different data exchange methods between proprietary tools or one common exchange format which all tools will use.

Towards the end of the 1980s and the early 1990s there was an increase in the popularity of CASE tools. At least two companies had brought out their own proprietary relationally based integrated CASE tools with repositories. Texas Instruments shipped IEF (Information Engineering Facility) based on the Information Engineering Methodology. (By the mid 1990s the client server version of IEF was called Composer) Oracle brought out the Oracle CASE tool based on a methodology called CASE*Method which was broadly similar to SSADM [Barker 1990]. (By the mid 1990s the client server version was called Designer 2000.) Both IEF and the Oracle CASE tools were based on relational technology. IEF was compatible with the IBM DBMS DB2 and other DBMSs products as will as its own proprietary product. The ORACLE CASE tool also had facilities for use with other DBMS as well as the Oracle DBMS. The repositories of both of these tools were passive repositories. Any changes made in the underlying DBMS which affected the conceptual model would also have to be reconciled with the repository. The Oracle CASE tool provided facilities to reconcile the DBMS with the Repository (which it called CASE*Dictionary) and vice versa, as did IEF. However these changes were not dynamic as the repository did not have an active run time link with the underlying DBMS. The DBMS still had an active system catalogue.

The rise of standards for repositories and the rise in CASE tools were further steps in the direction of trying to improve how we develop software. The aim of a repository is to provide a shareable container which can define the information resources of an enterprise. Shareable implies that services that use the container must adhere to certain standards. An enterprise's information resources could, obviously, include the documentation of the development of software systems and the subsequent use of those systems. This implies that CASE tools will have to interface to repositories - or in the case of integrated case tools model their central dictionary on a repository standard.

This parallel rise in CASE tools and the rise in standards for repositories do have an overlap[Byrne & Shah 1994, Olle 1992]. Integrated CASE tools need a repository, component case tools could map into a repository to share information with main systems. Integrated CASE tools and repositories also both had in common the support for, and documentation of all stages of the software development lifecycle as well as the possibility of documenting business and enterprise rules. It was recognised that the development of repository standards was a critical step towards a new industry of data integration CASE tools [Rosen 1991] and that repositories would benefit from more research by the database community [Bernstein & Dayal 1994].

AD/Cycle

In September 1989 IBM announced the development of Application Development Cycle (AD/Cycle). AD/Cycle was set to become the application development platform for the whole range of IBM equipment from mainframe to PC. It would be based on CASE technology and form an I-CASE tool with a common repository for systems development across a range of IBM platforms. Other major manufacturers already had CASE tools on the market or near completion (Oracle*CASE and IEF for example) and at the time IBM had fallen behind in the CASE market. The aim of the AD/Cycle repository was to provide a central store which all tools could plug into in order that information would be available across the development life cycle and afterwards.

AD/Cycle was an extension to IBM's System Application Architecture (SAA). SAA was launched in 1987 with the aim of achieving application portability across IBM's varying architectures and operating systems [Richards 1990]. IBM made arrangements with several third party vendors to use some of their CASE tools within AD/Cycle and AD/Cycle was to provide the tools for the development of systems on the SAA machines. IBM bought in, or had partnerships with, firms which produced upper and lower CASE tools and reverse engineering tools. One of the main partnerships was with Knowledgeware who produced Information Engineering Workbench (IEW). Knowledgeware's CASE tools worked with a common repository and it was assumed that IBM had bought in Knowledgeware's repository to use in AD/Cycle. These various component CASE tools would plug into the repository creating a life-cycle integrated CASE tool based on IBM's relational DBMS DB/2. Therefore the repository would also have all of the main features of a DBMS. AD/Cycle was also to include some expert system technology and object oriented features. The repository had a pre-defined Entity Relationship model with extensibility features.

The repository of AD/Cycle was to be the first deliverable. As this was a huge project and IBM were committing vast amounts of money and resources to AD/Cycle it was assumed by many that AD/Cycle would become the industry de facto standard for repositories [Richards 1990, Gillenson & Frost 1993] and that component CASE tool manufacturers would have to conform to AD/Cycle standards and be compatible with the AD/Cycle repository.

Portable Common Tool Environment (PCTE)

PCTE is an acronym for Portable Common Tool Environment and is described in the PCTE literature as a "standard for a public tool interface for an open repository" [Jowett & Wakeman

1993]. Similar to the ISO/IEC IRDS standard, PCTE describes a model for (framework) and operations to manage (services interface) for what is described as "an open CASE repository" [Jowett & Wakeman 1993 page 167]. Both IRDS and PCTE are repository standards however they are based on different paradigms and have different models. ISO IRDS has its roots firmly in database standards and uses SQL as its data model whereas PCTE has been influenced by programming language standards. The PCTE repository is a database that can model the structure of data and store the data used in the software development process. The repository provides the same range of services as the IRDS repository. The basic information storage element is referred to as an object, and instances of the objects are linked to an object type definition which is stored in the meta database. However PCTE is not based on object oriented technology although there were plans to add object oriented extensions. It is also an 'Open Repository' in that it is in the public domain, is not tied to any proprietary suppliers and does not require the payment of any license fees or royalties. Although PCTE was described as a public tool interface, the interface was to the data integration and services of a repository. Therefore it was not merely a data interchange tool but aimed for the full services of a repository (as described in the IRDS facilities in Chapter 3).

PCTE started as a project in 1984 and resulted from a European Strategic Programme on Research and Development in Information Technology (ESPRIT) project. ESPRIT projects were designed to carry out research and development in subject areas which could later develop viable commercial products by the companies involved in the project. The project included such organisations as Bull, Olivetti, GEC Software, Siemens and Nixdorf. By 1986 the PCTE Interface Management Board was formed (PIMG) with the aim of, bringing in new members to the project, increasing PCTE's usage and helping to develop PCTE as an international standard. PCTE became an ECMA standard in 1990 [ECMA 149] and in June 1993 it was decided to submit PCTE to ISO/IEC using the fast track submission route. The second edition of the ECMA standard consisted of four volumes. Two volumes covered the abstract specification which was a language-independent specification of the facilities. A further two volumes covered language specific bindings that would map abstract operations to a programming language. A language binding for C became an ECMA standard in June 1991 [ECMA 158] and a language binding for ADA was published in December 1991 [ECMA 162]. A PCTE working group was created in sub committee 22 of JTC1. PCTE , as part of the ESPRIT project did have some installed products, although these were mainly experimental and were intended to help in the development of the PCTE standard.

PCTE was regarded as a competing standard to IRDS [Bird 1994] and there were some publications comparing PCTE and IRDS [Oliver 1990] [Olle 1992] as well as work within ISO/IEC JTC1 on improving compatibility between the two standards. PCTE was the preferred standard for software engineering environments. The ISO JTC1 view was that there could be a rapprochement between the two standards with the possibility of an underlying repository having a

PCTE interface and an IRDS services interface (using SQL) [Gradwell 1993]. The PCTE publications were extensive. In 1993 there were four volumes consisting of the specification and implementational details and there were several pilot projects.

The CASE Data Interchange Format Standard (CDIF)

The CASE Data Interchange Format Standard (CDIF) was released in 1991 from the Electronic Industries Association (EIA). The purpose of the standard was to facilitate the movement of information between different CASE tools. The CDIF standard aimed to provide a single interchange standard between tools supporting the standard. Without a standard interchange format vendors would have to construct their own proprietary standards for exchanging data between other CASE tools. This would lead to a proliferation of different exchange formats. In Figure 5.1 if the ellipses are taken to represent vendors and the arrows are exchange formats, ten exchange mechanisms are required if vendors want to exchange their data with another vendor. However in Figure 5.2 if vendors are interchanging data using CDIF each vendor only has to exchange their data into a CDIF format which can then be used by any other vendor who has a CDIF interchange format.

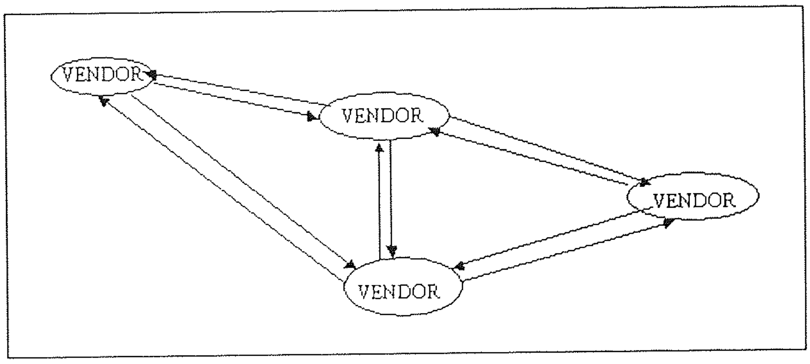

Figure 5.1 The CASE interchange Problem – vendors create their own exchange mechanisms

However the CDIF solution was relevant to exchanging data between tools and was not concerned with tools actually sharing information and accessing the same information. The role of sharing information between tools would need the services of a repository. The CDIF standards were intended to harmonise with the work of ISO and ANSI IRDS as well as PCTE. CDIF would provide an export – import facility for use with these standards. [Parker 1992b] [Thompson 1992], [Imber 1991]. CDIF itself was not a repository standard but designed for CASE data interchange.

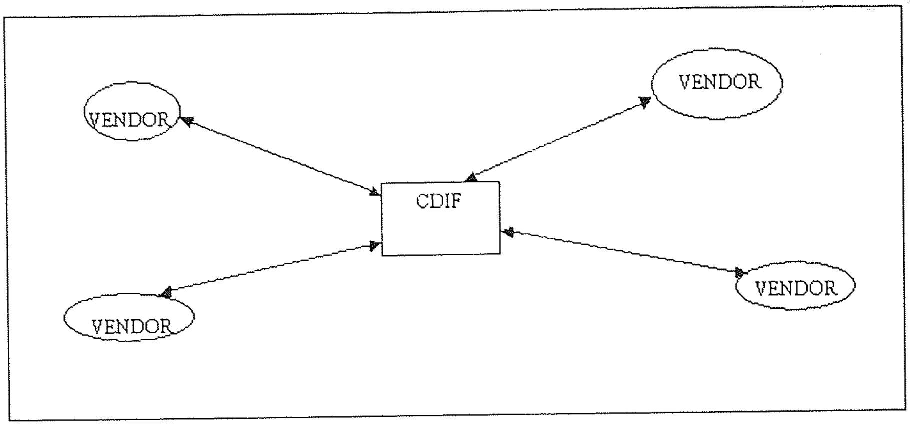

Figure 5.2 The CDIF solution with a standard exchange mechanism

It was acknowledged that data interchange could be useful as a prerequisite stage before the data was imported into a repository. IRDS and PCTE did not have an import/export language and CDIF was considered as a candidate for standardisation as the interchange standard.

Open Systems

During the late 1980's there was a move towards producing Open Systems. These were systems which use non-proprietary standards which were available to everyone. This made it easier for buyers to switch between hardware and software without having major reorganisation. The term "Open Systems" like many terms in the computing industry had a variety of interpretations. One interpretation was that "Open Systems" implied that the standards would be "open" in the sense that any interested party could participate and use the standards. A second interpretation was "open" in the sense that different computers could communicate easily with one another over short and long distances. Thirdly, "Open Systems" was interpreted as systems which would provide standardisation of the computers themselves, for example software which runs on one machine will run on other computers of the same type. In the United States "Open Systems" was often taken to mean portability and scalability of systems, usually using Unix. Whereas in Europe "Open Systems" often meant the interconnection of computers through the ISO OSI standard. [Gray 1991]. Nevertheless the Open Systems movement was driven by the need for compatibility between systems. In chapter two the procedure adopted by national and international standards bodies to produce standards was described as standardisation by consensus. The procedure is an open process. The Open Systems Interconnection (OSI) standard [ISO/IEC 7498] was an example of an open standard for connecting computers together at different levels, which was developed in a traditional standards organisation and motivated by the Open Systems movement. Two organisations which supported the Open Systems movement were the Open Software Foundation (OSF) and X/Open. The OSF was founded in 1988 and included sponsors such as IBM, DEC, Bull and Hewlett-Packard. Its aim was to produce software based on international standards. The procedure for producing the software would allow equal access by members and provide stable licensing terms. X/Open was founded in 1984 and its aim was to create a common applications environment (CAE) which would consist of all the elements of a computer system that would need to be standardised in order to achieve Open Systems.

The Open Systems Interconnection (OSI) Reference Model

The Open Systems Interconnection (OSI) Reference Model [ISO 7498] standard is an example of a guiding or reference model standard developed in traditional standards organisations and influenced by the Open Systems movement. Work began within ISO on OSI in 1978. The aim of the standard was to provide a standard non-proprietary approach to connecting heterogeneous computer networks. The OSI Reference Model became a standard in 1983. The standard dealt with computer to computer communication through the use of protocols with well-defined formats. Some of the protocols were already established and incorporated into the model; other protocols were developed within the standards committees. The OSI model used the concept of layers which was based on a hierarchy between various communication functions. The standard covered communication using a modem, over a public data network, using local area networks and the Integrated Services Digital Network (ISDN). The layers were (1) The Physical Layer, (2) the Data Link Layer, (3) the Network Layer, (4) The Transport Layer, (5) The Session Layer, (6) The presentation Layer and (7) The Application Layer. Similar to the IRDS standard the OSI standard began with a reference model which was used for guidance and understanding of how the layers fitted together. In the years following the publication of the reference model standards were developed for the seven OSI layers. The OSI standards were Open Systems and were intended to guide the development of communications from computer to computer and thereby avoid having a proliferation of different proprietary systems. The OSI Reference Model had a very high profile within ISO and an enormous amount of time and effort by the ISO and its national member bodies was put into the development of the OSI protocols.

CORBA

The Common Object Request Broker Architecture (CORBA) began to gain in popularity in the 1990's. CORBA was developed by the OMG (one of the consortia mentioned in chapter two) and the CORBA technology is based on Object Orientation. It is an open architecture in that it is vendor independent. Computer applications can use CORBA to work together over networks. Using a standard protocol a CORBA based program from any vendor can interoperate with a CORBA based program from the same or another vendor, on almost any other computer, operating system, programming language, and network.

CORBA applications are composed of objects which are individual units of software that combine functionality and data. Typically, there are many instances of an object of a single type - for example, an internet banking website would have many current account object instances, all identical in functionality but differing in that each is assigned to a different customer, and contains data representing the transactions relevant to that customer. For each object type, such as a current account, its interface is defined in OMG Interface Definition Language (IDL). This fixes the operations it will perform, and the input and output for each operation. This interface definition is independent of any programming language, but will map to popular programming languages via a set of OMG standards. OMG has standardized mappings for various programming languages.

Therefore CORBA enables interoperability by communicating with other objects through the IDL. The interface to each object must be defined very strictly. However the implementation of an object - its running code, and its data - is hidden from the rest of the system (in object oriented terms encapsulated) behind a boundary that the client does not cross. Clients access objects only through the object's interface. The acronym CORBA contains the word 'broker' which means 'middleman' and the CORBA software eventually became known as middleware

Chapter Review

This chapter has described other products and standards which were emerging in the late 1980's and early 1990's at the same time as the ISO/IEC IRDS standard was published as an international standard. Integrated CASE tools which were used in an Information Systems environment had central repositories which displayed similar functionality to the ISO/IEC IRDS functionality. Component and Integrated CASE tools were becoming popular. There was an obvious link between the functionality of ANSI and ISO IRDS and the functionality of I-CASE tools. ISO and ANSI were supporting the use of CDIF as a common data interchange standard. There were other competing repository standards. Firstly PCTE which was the preferred repository in Software Engineering environments as it was based on the programming paradigm and secondly AD\Cycle was predicted to become the de facto repository standard as it was produced by IBM. Also, outside of the repository based arena work was continuing within ISO on the OSI Reference Model standards and CORBA and object oriented technologies were gaining in popularity.

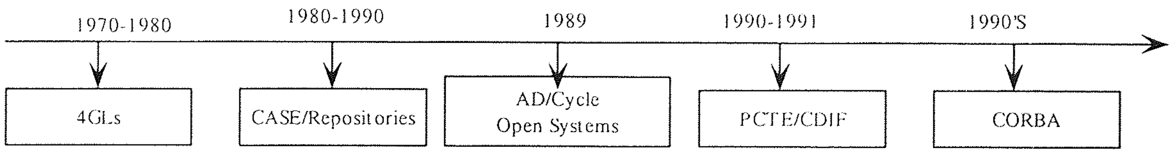

Figure 5.3 Timeline from 1970's – 1990's

This chapter has described other technologies and standards which were emerging in the late 1980's and early 1990's. There appeared to be a race towards a global repository standard. A convergence of ISO/IEC IRDS and PCTE could be a useful solution as it would combine the database paradigm with the programming paradigm. However it was possible that the de facto standard from IBM would become the more popular repository standard as it would have an immediate market presence with IBM's backing. Therefore although the ISO/IEC IRDS standard was the published international standard for an Information Systems style repository there were at the time other repository standards based on different technology.

The next chapter considers previous work which has been carried out on classifying different types of standards and considers why it is useful to classify standards. The chapter also examines previous work on models of the standards process.

CHAPTER SIX: A CLASSIFICATION OF STANDARDS

This chapter reviews previous work in the field of classifying and categorising standards. To fulfil the different needs of IT and ICT there will be an array of different standards. Standards will have different purposes; for example programming language standards are required to ensure portability of the programming language code and to ensure wide-spread use. The programming language may change over the years as new technology evolves and new versions of the standard are produced. The IRDS standard and the OSI Reference Model are standards that contain other standards. The programming language standard could be described as a base standard as it is at a lower level and the OSI and IRDS standards could be described as reference or guiding standards.

It is necessary to have a classification scheme for standards for two main reasons. Firstly different types of standards may need different treatment by the standardisation bodies and in the standards development process and therefore a classification of standards is useful. Secondly a classification or model is always useful to aid our understanding of a complex situation.

There are some anomalies in the terminology used to describe standards and the standards process. Firstly in the literature the words taxonomy, topology and classification are used to describe a categorisation of standards. Also occasionally taxonomy, topology and classification are used interchangeably with the word model. This work takes the point of view that a model is a static example or paradigm which acts as a framework for organising the way we think about a problem. Therefore in this work the word model is regarded as meaning the same as classification or categorisation. This work also takes the point of view that there are two types of standards process; firstly the administrative process of standardisation in formal SDO's or consortia (covered in chapter two) and secondly the process which happens in the industry which leads to standardisation (whether it be de facto or de jure standardisation) this latter process is often affected by economic considerations and is considered in Chapter 7. A clear distinction will be made between these processes in this work. These will be referred to as the administrative standards process and the standards diffusion process. This work will also refer separately to a classification of standards. The terms taxonomy, typology, categorisation and classification are regarded as being synonymous. Therefore there are three main areas to consider:

1. Administrative Standards Development Process - covered in Chapter 2
2. Classification of Standards - covered in this chapter - Chapter 6
3. The Standards Diffusion Process - covered in Chapter 7

This chapter will outline and comment on the various models/classifications which have been proposed to date. In order to illustrate each classification/model in the same way the author has used a conceptual data modelling technique used by Elmasri & Navathe [2000]. This modelling technique, although primarily used for database design, is a semantically rich technique for illustrating supertypes and subtypes and is an extended version of the original Entity Relationship Model [Chen 1976]. In the discussion which follows each classification is represented using this modelling technique. A brief description of the modelling technique is given after Figure 6.1 in the next section. For further details the reader is referred to Elmasri and Navathe [2000].

David's Classification

The review begins in 1987 with a classification of standards produced for economic purposes. David [1987] put forward a classification of standards which covered standards generally (not specifically for IT). David takes an economic view of standards and his reason for classifying standards was for the purpose of formal economic modelling. However David points out that previous work [Kindleberger 1983] classifies standards according to how they affect economics; a standard can aim to reduce transaction costs or a standard can aim to realise physical economies of scale. David points out that this is not an acceptable classification because standards could fall into both classifications. David also suggests that a dichotomous classification between de facto and de jure standards is also not acceptable as again standards could fall into both categories (as indeed the SQL standards does as it was a de facto standard which eventually became a de jure standard). David states that it is difficult to classify standards because there is often no mutual exclusivity as with many classifications standards can often overlap classes and fall into more than one class.

David introduces his classification in two different streams. The first is standards of technical design and the second is standards of behavioural performance. The standards for technical design deal with the nature of objects. The standards of behavioural performance deal with human behaviour, procedures and performance. (An example of the latter this would be the ISO 9000 standard for quality management.) Within both streams David has three groups of standards. His original classification is reproduced below in table 6.1

(taken from David 1987 page 214)

Table 6.1 David's classification

This work is mainly concerned with the first column - the standards for technical design, as this is where the majority of standards for IT and ICT will fall (however there are IT standards which would fall into the stream of behavioural performance for example SSADM). To clarify David's classification the first group, standards for reference are, for example, currency. Within the U.K. we all use the same currency - sterling - which is the unit of reference for money. Sterling is a definite unit of reference and cannot be varied in any way. Other examples of standards for reference would be miles, kilometres, litres, gallons, byte, alphabet and character sets. David's definition of a technical reference standard is "it functions as a reference point in only one dimension" [op. cit page 215].

The second group "standards for minimal admissible attributes" is where a standard serves as a paradigm against which products may be tested to see if they meet that standard. For example a child's toy could be tested to see if it meets certain safety standards. Within this group the standards of reference units, mentioned above, may often be used as base standards. In this way some of the later classifications of standards build on the previous classifications. Also, standards assuring some minimal level of quality would fall into this group. For example to claim conformance with a standard a product must contain certain characteristics.

The third group "standards for interface compatibility" is to facilitate physical interactions at interfaces between objects or between objects and agents. For example the programmer and compiler being used, the touch typist and the keyboard layout (*David's example*). IT and ICT standards which fall into this last group may be built on the unit of reference standards and the standards for minimal admissibility. Therefore the later standards may, but not always, be built on previous standards in the classification. The windows interface standard, although, not a de jure standard is an example of a standards for interface compatibility. Many applications and programs now use a windows interface. End users are familiar with windows and expect the same functionality and the interface to behave in the same way, with similar icons and mouse clicks.

David's classification does not make any further divisions within the "standards for interface compatibility" group. David's classification is useful as a general classification scheme however it is necessary to further sub-divide this classification with relation to IT and ICT as there are major differences in some standards.

David's classification is represented below in Figure 3.1 using the modelling technique described in Elmasri & Navathe [2000]. As stated earlier the author has chosen this modelling technique for two reasons. Firstly, the technique is semantically rich and can deal with the problem of standards falling into more than one classification, and can show a hierarchy of super and subtypes, with overlapping subtypes as will as disjoint subtypes. Secondly, the modelling technique provides a way of bringing together previous work on the classification of standards and presenting these classifications in a uniform way.

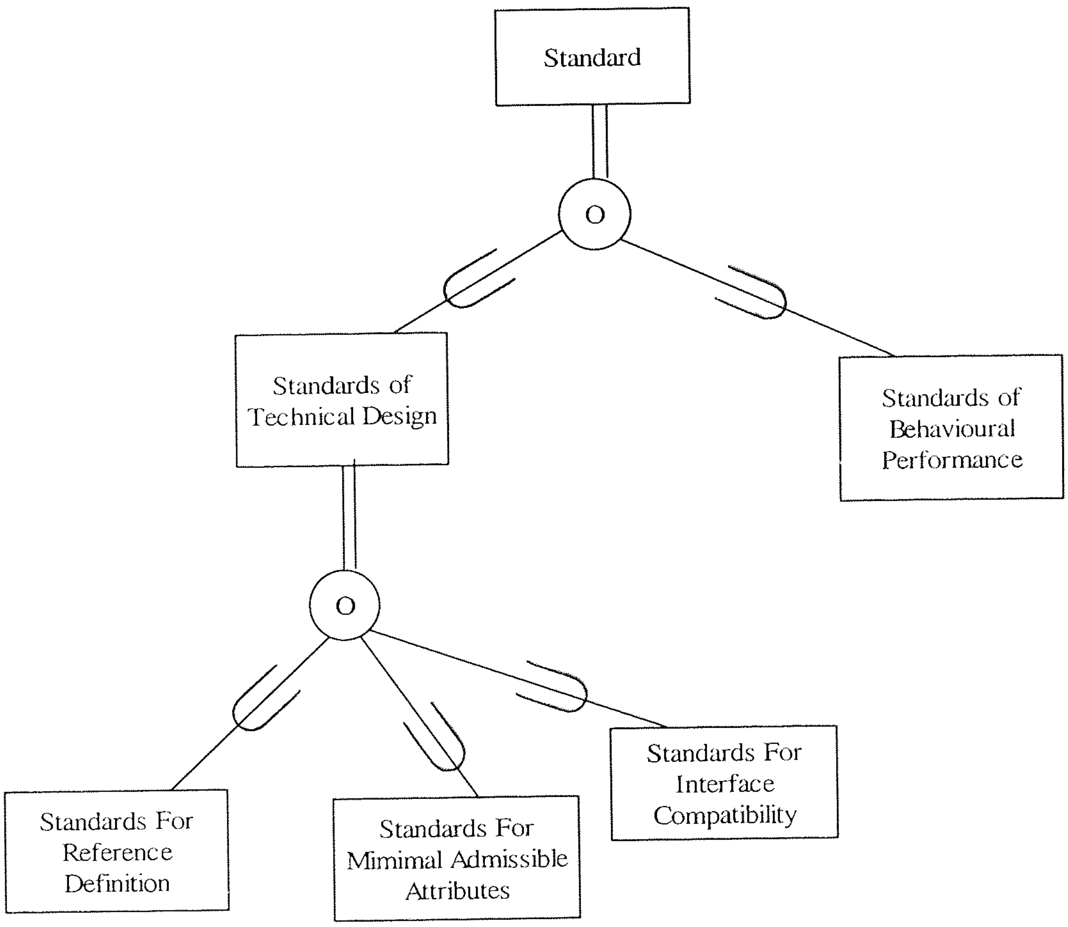

Figure 6.1 David's classification of Standards

Explanation of the Modelling Technique

The model in figure 6.1 shows a hierarchy of supertypes and subtypes in three levels. The first entity, 'Standard', is specialised into the two groups of standards of technical design, and standards of behavioural performance as in David's classification. Specialisation is the process of defining a set of subclasses of an entity type. The entity type which is divided into subclasses is called the superclass. Entity types may have several different specialisations for example an entity type called 'Employee' may have two specialisations – one with subtypes defined on the job type of employee and another specialisation based on whether the employees are full time or part time employees. In David's model above there is just one specialisation of the entity 'Standard' which is specialised into the subtypes of 'Standards of Technical Design' and 'Standards of Behavioural Performance'. Therefore standards of technical design and standards of behavioural performance are both subtype entities of the supertype entity 'Standard'. (This work is mainly concerned with standards of technical design and will not pursue the right hand side of the diagram which deals with standards of behavioural performance.) The relationship between the superclass and its subclasses is an 'IS-A' relationship. The diagram is an example of an enhanced Entity-Relationship diagram and the entities appear as rectangles in the diagram. Sub-type entities will inherit all the characteristics of the supertypes entities it is related to. In figure 6.1 the entity of 'Standards of Technical Design' is specialised into three subtypes, standards for reference definition, standards for minimal admissible attributes and standards for interface compatibility. The letter 'O' in a circle separating the subtypes stands for 'overlap'. In this case the model is illustrating that a standard may fall into both groups and that one particular standard could be in the technical design group and also in the behavioural performance group. With this modelling technique 'O' is the default. The other choice is 'D' for disjoint. As stated earlier one of the difficulties in classifying standards in previous work has been the inability to show that standards can overlap various categories within a classification. This modelling technique allows for this by showing that the subtypes may overlap. In this particular case it is possible that a standard could fall into both subtypes and therefore an 'O' is put in the circle. If a 'D' were inserted in the circle it would state that the two groups are disjoint. Also, in the second level of subtypes (standards for reference definition, standards for minimal admissible attributes and standards for interface compatibility) we are obliged to place an 'O' for overlap as it is foreseeable that these groups may also overlap. This modelling technique can also show whether participation of an instance of the supertype in the subtype is compulsory or not. This is shown by using a double line for total participation of the entities in the supertype in the subtype, or a single line to show partial participation of the supertype entities in the subtype. In figure 6.1 a standard which is classified as a standard for minimal admissible attributes also inherits the attributes of the standard of technical design.

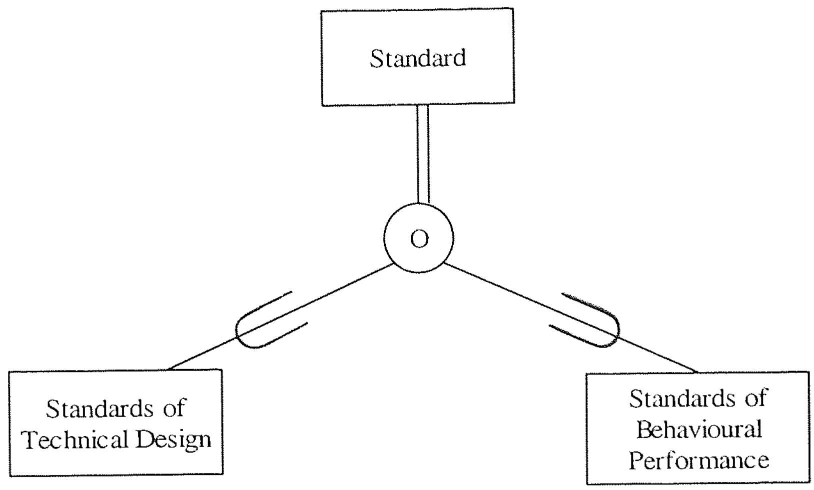

Figure 6.2 Example of total participation of the supertype entities in the subtypes.

In Figure 6.2 participation of the supertype in the subtypes is total. Every standard instance must be either a standard of technical design or a standard of behavioural performance and in this case they could also be both as they may overlap.

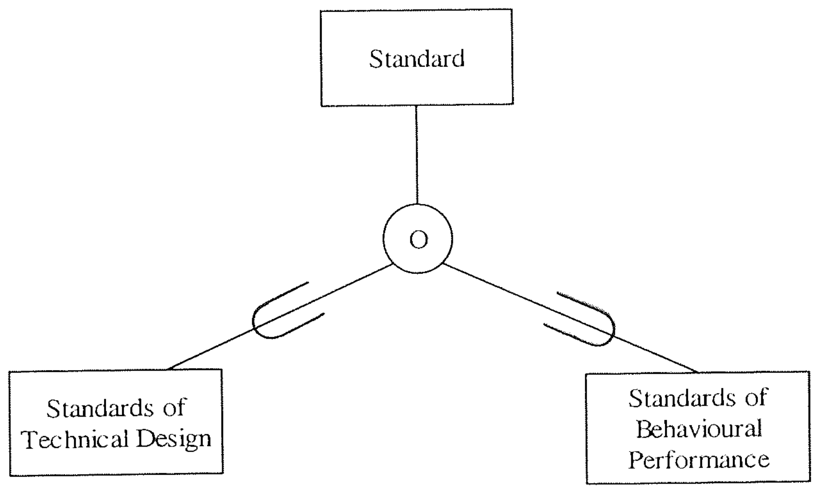

Figure 6.3 Example of partial participation of the supertype entities in the subtypes.

Figure 6.3 states that the participation of the supertype in the subtypes is partial. In this case there may be other standards which do not fall into the categories of standards of technical design and standards of behavioural performance.

The author has found this modelling technique useful for classifying standards as it deals directly with the problem of standards not being mutually exclusive in a classification scheme.

Review of David's Classification

It could be argued that David's standards for reference definition are not necessary in IT and ICT as reference definition standards are dealing with basic requirements, for example currency and weights. However many IT standards could be placed in the 'minimal admissible attributes' class. A programming language implementation could be tested to see if it conforms to the international standard for that programming language. For example an SQL implementation could be tested against the SQL standard. An SQL implementation may conform to the standard but the vendor may have added proprietary enhancements to the basic standard core. This has often happened with SQL implementations and programming languages. This is problematical with regard to IT and ICT standards. Should products to be exactly the same as the standard or do they have to have a minimum core of admissible attributes to conform to the standard? In IT and ICT it is preferable if products are the same as the standard specification as this helps interoperability and portability. Having only a minimum set of admissible attributes to conform to the standard will not necessarily be sufficient or ideal. David's class of 'standards for interface compatibility' are appropriate for IT as there is a need for standardised interfaces between products.

David was writing from the point of view of an economist and was not concentrating specifically on standards for IT. Technical standards for IT and ICT can be placed mainly into David's "standards of technical design" stream. However within this group we need to further categorise the types of standard. The IRDS Framework Standard is difficult to place in David's classification. It could possibly be placed in the standards for minimal admissible attributes group but this would not take account of the fact that the IRDS standard is a reference standard containing other standards within it. The Services Interface Standard could be placed in the same group of 'minimal admissible attributes' although this seems inadequate as it is now in the same classification as the IRDS Framework standard and they are different types of standard. The IRDS Framework is a guiding standard and the Services Interface contains implementational details for an IRDS.

Cargill's Classification

Cargill [1989] puts forward a typology specifically for IT standards. He divides standards into two groups:- implementation standards and conceptual standards. Implementational standards are described as standards for implementing a particular device. Cargill states that these can be regarded as evolutionary standards as the standard evolves over time to meet changes and new advances in the industry. Cargill uses the example of the programming language Cobol as an example of an implementational standard, and obviously other programming language standards would fit into this category. On the other hand the conceptual standards are designed to change industry perceptions and steer the direction of the industry. Cargill describes these as revolutionary as they seek to:

"change industry perceptions and direction, to encourage technology conversion or change, or to redefine an industry problem through a different perspective on the approach to a solution"[op cit]

Cargill points out that conceptual standards are usually time-consuming to produce and also face the problem that forced change is difficult to introduce. As an example of a conceptual standard Cargill used the IEEE 802.3 Ethernet standard. The aim of IEEE 802.3 is to provide stand-alone users access to computing facilities by concentrating on communications between the user and the centralised resources. As the standard developed Local Area Networks (LANs) were created and the standard and the technology in the area continued to grow and evolve.

Within the two groups of conceptual and implementational standards Cargill subdivides the standards in each group into process or product standards. The product standard is described as a model or paradigm for the product within the industry (similar to David's class of standards for minimal admissible attributes). Whereas the process standard is usually future oriented and describes a future need without specifying a certain implementation. Cargill states that there are four possible combinations which the author has shown on the following matrix.

	Implementation	**Conceptual**
Product	✓	✓
Process	✓	✓

Table 6.2 Cargill's Classification of Standards

In the matrix in table 6.2 the standards are either implementational or conceptual and within these divisions the standards can be either product or process oriented. Out of the four possibilities in the typology Cargill describes the Implementation/Product standard as the most common, however, he points out that because of the fast pace of change in the IT industry the process standards are becoming more common. A description of each type of standards follows:

Implementation/Product Standard

Cargill describes this standard as dealing with an established product or service. The standard will usually have known rules and boundaries and be a product that is being updated or revised to keep up with change in the industry, or it could be a standard which is being used to formalise a current de facto standard. These standards are often in areas which have clear boundaries and are clearly defined. These standards are created or revised to serve a "predictable market with a predictable response".

Implementation/Process

Cargill states that this type of standard is difficult to categorise. These standards are usually designed for a situation 'where a well established need requires a standard that is not bound to a single technology or methodology'. The standard should describe the process of obtaining the expected result, not the product that produces it. Cargill's example is the telephone system. The need to communicate by voice over distance exists - the users are not concerned with the technology used to achieve this end.

Conceptual/Product standards

This type of standard usually relates to a product with a technical variation where there is some indecision in the way forward. In this case there is a demand for a standard which is future oriented as the users and providers recognise that it will be best for everyone if a standard is used. The IETF standards, mentioned in Chapter 2, would fall into this group. This type of standard is usually a "potential solution to a future problem" rather than an "immediate solution to a current problem" [Cargill 1989].

Conceptual/Process standard

A standard which describes "an expected outcome for a future need". The danger with this type of standard is that it may be overtaken by alternative processes which make it obsolete. Also because this standard is completely future oriented the description of its processes and the process itself have to be vague to ensure that it can apply to future developments that may happen in the area it is applied to. These standards may often have to have vague areas in order to accommodate a variety of different views or implementations. Table 6.3 the reproduces the matrix with some recent examples.

	Implementation	**Conceptual**
Product	a standard to serve a predictable market with a predictable response. (Cobol, SQL)	a standard to offer a potential solution to a future problem (IETF, OSI standards)
Process	a well-established need requires a standard - the standard describes the process not the product that produces it. (SSADM)	A standard to serve an expected outcome for a future need. (IRDS)

Table 6.3 Cargill's Standards Matrix with examples

In table 6.3 programming language standards are placed in the Implementation/Product category as they are dealing with an established product or service with a predictable market. The IETF and W3C standards are placed in the Conceptual/Product category as they are standards which are leading the way in their area. As explained in Chapter 2 vendors have come together in consortia to produce new standards for the technology as the technology is evolving. The OSI Reference Model standard could also be placed in the Conceptual/Product group as it could be described as being a standard which dealt with "a potential solution to a future problem".

SSADM has been given as an example of the Process/Implementation standard. SSADM describes the process of obtaining the expected result, not the product that produces it. (This has similarities with David's behavioural standards which encompassed behavioural procedures and performance.) The IRDS standard has been placed in the Conceptual/Process class. The IRDS standard did describe "an expected outcome for a future need". The IRDS Framework Standard could be described as completely future oriented as there were no obvious products in the market place which conformed to IRDS, although the I-CASE tools had many similarities. Also within the IRDS family of standards work was ongoing on content modules for different areas to fulfil future needs.

In figure 6.4 Cargill's classification has been produced using the conceptual modelling technique described earlier. In figure 6.4 a single line has been drawn from the supertype standard to the subtypes. This illustrates that not every standard will fall into the subtypes of implementation and conceptual and thus allows for the fact that there may be standards which do not fall into either category. This first level of the model in figure 6.4 can be regarded as a particular specialisation of the entity standard which has the subtypes of implementation and conceptual standards which are in this case disjoint (although it may be argued that these could overlap). On the third level in the diagram there is a further specialisation into product and process standards for both the implementation and conceptual standards. Again as there may be a possibility that these could overlap (a product standard could also be a process standard) an 'O' for overlap has been placed in the circle. In the model in figure 6.4 there is a repetition of the product and process sub-types on both sides of the diagram.

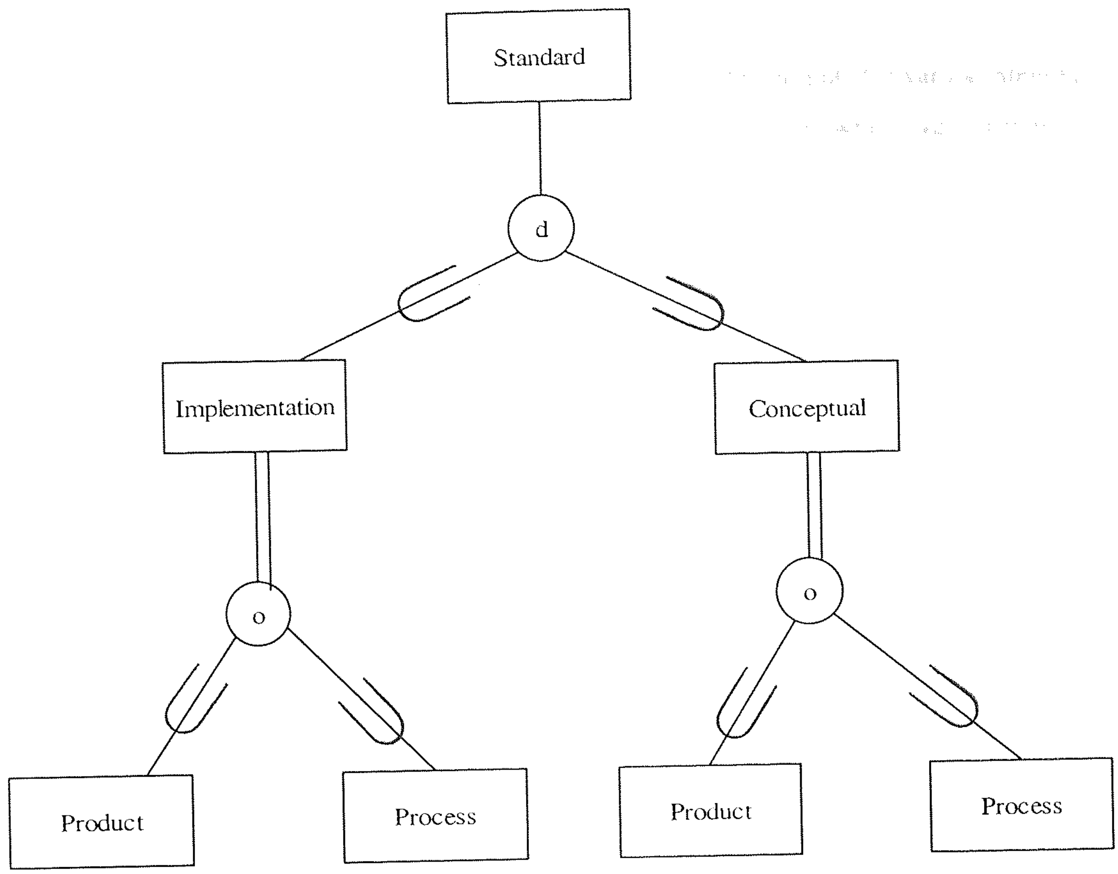

Figure 6.4 Cargill's Classification of Standards

Cargill's five-stage model

In Cargill's [1989] extensive work on standardisation he describes the process of standardisation in 1989 as being a two-stage approach. The first stage is where a de facto standard exists and the second stage is where a formal Standards Development Organisation (SDO) produces a formalised de jure standard from that de facto standard. He states that the standards process has not moved on from this model and he advocates that with the rapid changes occurring in the computing industry that the standards process must move on from the two stage approach and attempt to produce standards which anticipate the creation of products.

Cargill introduced a five-stage model (covered in chapter two) for standardisation within which he shows the different emphasis of providers and users in the standards process. His five-stage model basically advocates more involvement of the users in the implementational details of a standard which could then be fed into the providers view. He recognises that if standards bodies are going to move on from the 'two stage' approach towards anticipating standards for products then the potential users of those standards must be more involved in the process. This explains Cargill's class of conceptual or anticipatory standards which he felt were a way for the standards organisations to move on from the 'two-stage' approach.

Review of Cargill's Classification

Cargill's work[1989] continues David's[1985] work on standards. David puts forward a category, within his classification, which is suitable for IT standards. Cargill's work, which was written specifically for IT standards, further subdivides this category into implementational or conceptual standards further divided into product or process standards. Towards the end of the 1980's it was obvious that the traditional SDO's were having problems producing appropriate standards within the rapidly changing area of IT. Anticipatory or conceptual standards were seen as one way in which the SDO's could deal with this change. Cargill's conceptual category first introduced the concept of anticipatory standards - standards which could be a solution to a future industry problem. Cargill [1995] describes an anticipatory standard as: "creation of a standard from existing technology without an existent commercially available reference implementation". He says that OSI Reference Model is the classic case of this type of standard. Cargill describes the product standard as a model or paradigm for the product within the industry which is similar to David's category of minimal admissible attributes. His work does not explicitly refer to interface standards which were mentioned in David's work. His division into product and process is similar to David's division into standards of technical design and standards of behavioural performance. Where the standards of technical design can be aligned with Cargill's implementational/product standards and the standards of behavioural performance can be aligned with Cargill's implementational/process standards. However this is not a clear division. The new concept in Cargill's work was the introduction of conceptual standards/anticipatory standards which was not present in David's work. Cargill was also concerned that the formal standards industry should move on from the two-stage model of taking a de facto standard and formalising it and his five stage model of the process of standardisation was an attempt to move the standards industry forward towards creating conceptual or anticipatory standards.

In Cargill's classification the author has placed the OSI Reference Model and the IETF standards in the same group of conceptual process standards. However there is a difference between these standards as the OSI standard, similar to the IRDS standard, is a Reference Model or Framework standard which contains other standards within it. The IRDS Framework standard has been classified as a conceptual process standard as would the Services Interface Standard. Although Cargill's classification can show anticipatory standards there is no way to show a reference or framework standard in the classification.

Krechmer's Classification

Writing at a later date in 1996 Krechmer [1996] puts forward a classification of standards which has similarities to David's classification. Krechmer is concerned with comparing the relationship of standards, communications and technology to the ongoing waves of society. He refers to a trading wave, an industrial wave and an information wave. His classification is reproduced below.

In table 6.4 he relates the types of standards to waves of change in society, and in table 6.5 he classifies the standards which refer to each wave in more detail.

Table 6.4 Krechmer's Classification (taken from Krechmer 1996 page 5)

Table 6.5 – Krechmer's Four Classes of Standards (taken from Krechmer 1996 page 7)

Krechmer's work addresses technical standards in particular and does not consider behavioural standards of performance. Therefore his work can be inserted in David's model in the subtype of standards of technical design. His first three classifications; units, similarity and compatibility can be directly aligned with David's three groups of; reference definition, minimal admissible attributes and interface compatibility. Krechmer introduces a new group in the classification called etiquette standards which he describes as being part of the adaptive phase of the information wave. He describes the adaptive phase as beginning with the Internet. The pre-internet phase involved linear processing where there was limited sharing of information. The move from linear to adaptive systems was marked by the beginning of the Internet and the Open Systems movement in the 1980's.

Review of Krechmer's Classification

Krechmer's work particularly addresses technical standards in IT and ICT. His standards classification is similar to David's however Krechmer was writing at a later date and has added a further level to the classification which he has called etiquette standards. In Figure 6.5 below, because of the similarity between the David and Krechmer's classifications, the two have been

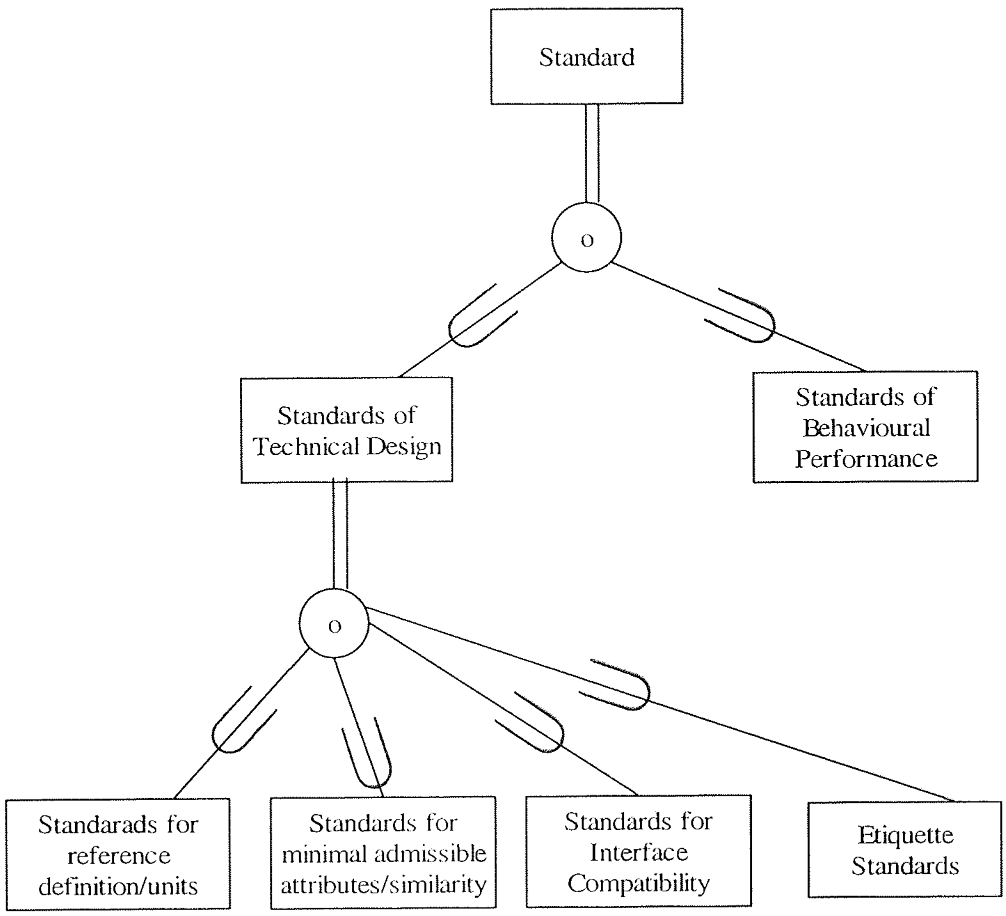

Figure 6.5 David's and Krechmer's Classification

combined. With Krechmer's classification there is no obvious category for the classification of reference or framework standards and therefore no obvious classification for the IRDS Framework Standard.

Bonino and Spring's Classification

Bonino and Spring [1991] take a global view of the classifications which have been proposed up to 1991. Their taxonomy has three groups which are reproduced in the table below. (Bonino & Spring use the word model whereas David and Cargill use taxonomy and typology respectively). In some cases Bonino and Spring use the word model to describe what in this work has been called the economic process - how the standard arises under market conditions in the industry. Bonino and Spring combine the economic process, the administrative process and the classification together and describe them as different types of models within a meta model.

Descriptive or Classificatory Model	A method for classifying events or entities. (for example David's classification model)
Explanatory Model	An abstraction which correlates events and outcomes allowing analysis, for example the economic process models (the economic explanation of why a standard has come into being) and Cargill's five-stage model.
Prescriptive Model	Allow outcomes to be predicted. No model but Bonino and Spring introduce one here based on Cargill's work

Table 6. 6 Bonino and Spring's Classification

David's classification could be used as an example of a descriptive or classificatory model. Cargill's five-stage model is an example of an explanatory model. Bonino and Spring, in this category, also refer to economic process models (which are the subject of the next chapter in this work). The prescriptive model covers the area of anticipatory or conceptual standards outlined in Cargill's work on types of conceptual standards. Bonino and Spring suggest an alternative classification of standards in this category which they describe as a prescriptive model.

Bonino and Spring's Prescriptive Model

Reference standards

Design standards

Implementation standards

They state that in this model standards are either reference, design or implementation standards. Engineers will be concerned with implementation standards whereas the reference standards will be the concern of strategic planners. The reference standards are anticipatory standards which are "writing for the future now". They suggest that different categories of standards will require different treatment. Implementation standards can be dealt with in the traditional way - where a product is developed and then becomes standardised, however, the reference standards are acting as "change agents" in the IT market. They do not comment on the design standards.

Review of Bonino and Spring's Classification

Bonino and Spring state that there needs to be a paradigm shift from the historical technical standards (which Cargill refers to as the two-stage model) towards viewing standards as change agents. However they do not deal with how the administrative process and the economic process differs for these standards. They do not comment specifically on the design standards classification in their prescriptive model. They point out that further work is necessary in this area. The overriding point in Bonino and Spring's work is that they felt that the IT industry should be moving towards anticipatory standards.

If Bonino and Spring's model is regarded as a meta model, David's classification can be placed in the descriptive or classificatory model. Krechmer's model is both classificatory and explanatory as he juxtaposes the progression of different standards with different waves in society. For this reason in the model in Figure 6.6 below an 'O' for overlap needs to be inserted into the circle to show that the different subtypes may overlap. The IRDS standard would be classified in the reference standard category as would the OSI standard. The Services Interface Standard could be placed in the implementation standards subclass.

However Bonino and Spring have included the three processes of the classification of standards, the administrative standards development process and the diffusion of standards in the same model. In this work these will be treated separately. As stated earlier the author would prefer to regard these as the separate issues of; the classification of standards, the administrative Standards Development Process and the Standards Diffusion Process. The author feels that to combine the three leads to over-complication. However Bonino and Spring's classification does have a suitable category for the IRDS standard which can now be placed in the reference standard group.

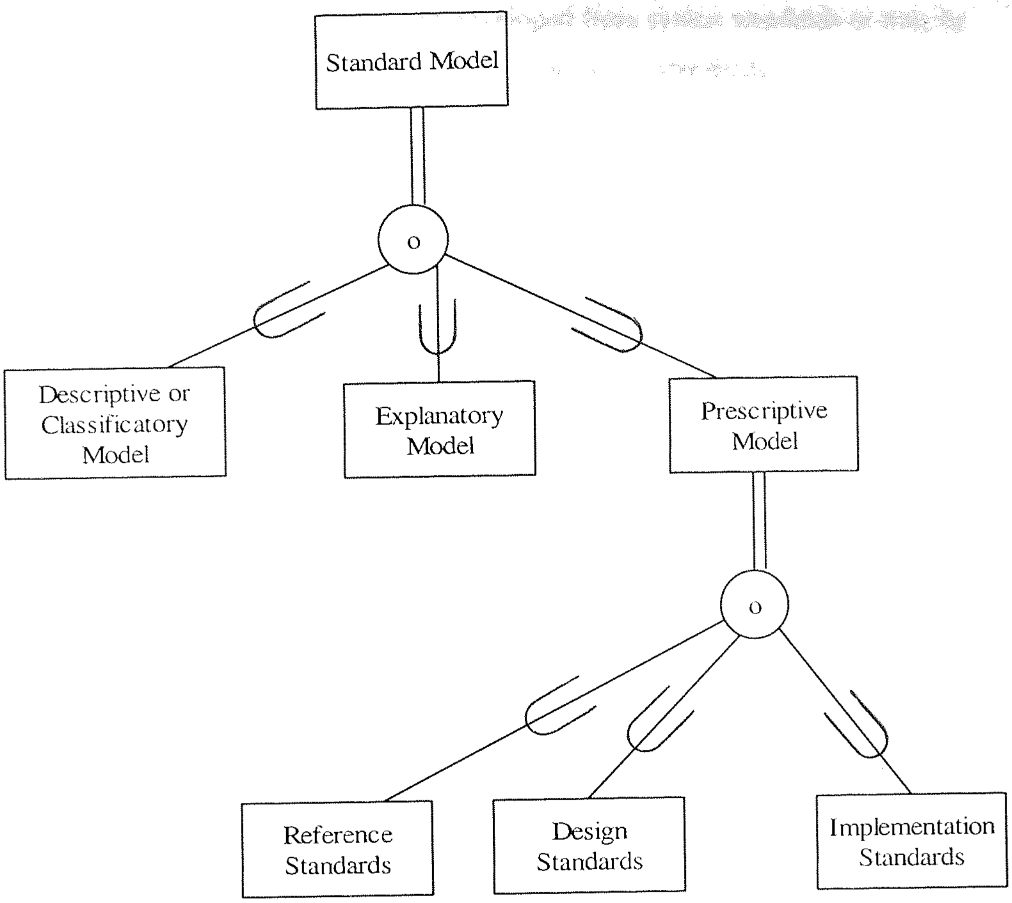

Figure 6.6 Bonino & Spring's Classification

Spring and Weiss' Classification

Spring and Weiss [1995] put forward a classification dividing standards into four groups. The first group, reference standards, are standards written in order to focus other standards and they are not implemented directly. They cite the OSI Reference Model as a good example of a reference standard. Spring and Weiss state that reference standards are nearly always anticipatory. The second group are base standards which describe 'a measurable and implementable product or process'. Base standards are, according to Spring & Weiss, the main standards developed by SDO's. The cite computer language standards as examples of base standards. The third group are syntax standards which are languages or procedures which may be used to develop other standards. The example used here is SGML which is intended to produce implementations of the language that are themselves standards. The syntax and base standards they can either be traditional base or syntax standards or anticipatory base or syntax standards. They describe traditional standards as those standards which are based upon products that exist in the marketplace and anticipatory standards as those standards which are developed before products exist. The final group are implementational or derivative standards which are developed from syntax standards or may be linked to a base standard. This gives the following classifications of standards:

- Reference
- Anticipatory Syntax
- Traditional Syntax
- Anticipatory Base
- Traditional Base
- Implementation

In the model below in figure 6.7 the author has placed anticipatory and reference standards on the same level. The 'O' in the model shows that these standards may overlap. A standard can be an anticipatory standard, it can be a reference standard, it can be a traditional standard, or it can overlap any of these categories. There is repetition of the syntax and base standards on either side of the model. In the model in figure 6.7 the derivative standards are regarded as the same as syntax standards. Spring and Weiss describe the syntax standards as being languages or procedures used to develop other standards and the derivative standard as being developed from syntax standards or linked to a base standard. There is a modelling technique called a 'category' [Elmasri and Navathe 2000], which allows the selective union of instances from a group of superclasses to form a new subclass, which could show the derivative standards on a fourth level in the model in figure 6.7. Alternatively the relationship between the syntax standards and the derivative standards could be regarded as a recursive relationship in that a syntax standard could make up another or many other syntax standards. However in this model syntax and derivative standards are regarded as being the same as they are both linked to base standards.

Review of Spring and Weiss' Classification

Spring and Weiss make direct references to anticipatory standards. Similar to Cargill's model they describe anticipatory standards, which they refer to as conceptual standards, and they also describe traditional standards which Cargill refers to as implementational standards. They regard the reference standard as being a guiding standard which is not implemented directly and is nearly always anticipatory.

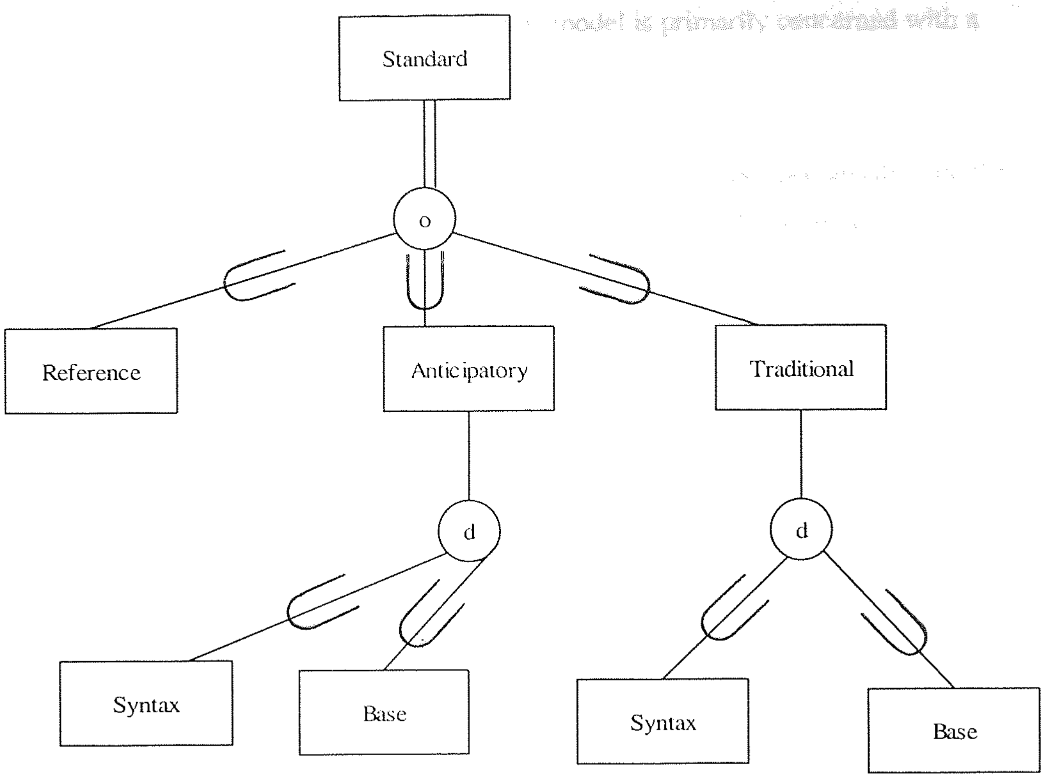

Figure 6.7 Spring & Weiss' Classification

In the model in Figure 6.7 there is now a definite category in which to place the IRDS standard. It can be classified in the reference subclass and the anticipatory subclass. These subclasses may overlap as there is an 'O' in the circle above. The IRDS standard fits into Spring and Weiss' definition of a reference standard as "a guiding standard which is not implemented directly and is nearly always anticipatory". The Services Interface Standard can be placed in the anticipatory and syntax subclasses as it was an anticipatory standard based on SQL and Pascal.

A New Classification of Standards

There are similarities between the various classifications of standards which have been developed however there are also many differences. One of the main problems has been to allow for the fact that a standard may fall into more than one classification. In Figure 6.8 below an intermediate classification has been created by combining the elements in the previous models/classification schemes which are relevant to IT standardisation. David's class of standards of technical design is taken as being the obvious classification in which to place IT and current ICT standards. Krechmer's classification has been combined with David's classification to provide the third level in the model. Spring and Weiss' model is combined with Cargill's to provide the right-hand side of the model which includes anticipatory/conceptual standards. Bonino & Spring [1991] introduced an over-arching model which combined the three areas of; the standards administrative process, the standards dissemination process and classification schemes. Bonino and Spring's

Model has not been included in the new model as the new model is primarily concerned with a classification of standards.

In the Figure 6.8 the supertype 'standard' has two specialisations. The first specialisation on the left hand side of the diagram contains the sub-types 'standards of technical design' and 'standards of behavioural performance'. This is taken from David's original classification. Computing and IT standards will fall mainly into the standards for technical design. An 'O' for overlap has been placed in the specialisation as it is feasible that a standard for technical design may also be classified as a standard for behavioural performance. The sub-type of standards for technical design has a further specialisation of sub-types for reference definition standards, standards for minimal admissible attributes, standards for interface compatibility and etiquette standards. This specialisation combines David and Krechmer's work. Again it is possible that these standards may overlap, for example an etiquette standard may also be regarded as a standard for interface compatibility and therefore an 'O' for overlap has been placed in the specialisation.

The second specialisation on the right hand side of the diagram divides standards into either anticipatory/conceptual standards or traditional /implementational standards. A 'D' for disjoint in the specialisation shows that these standards are disjoint and do not overlap. The anticipatory/conceptual standards are further specialised into reference, syntax or base standards. The traditional/implementational standards are specialised into syntax or base standards. The right-hand specialisation is taken mainly from Cargill's work and Bonino and Spring's work.

However the traditional standards and the syntax and base standards on the right hand side of the specialisation in this intermediate model can be considered as the same as the standards of technical design on the left hand specialisation. The intermediate model is repeating the type of standard. For example a syntax or base standard could be a standard for interface compatibility or an etiquette standard.

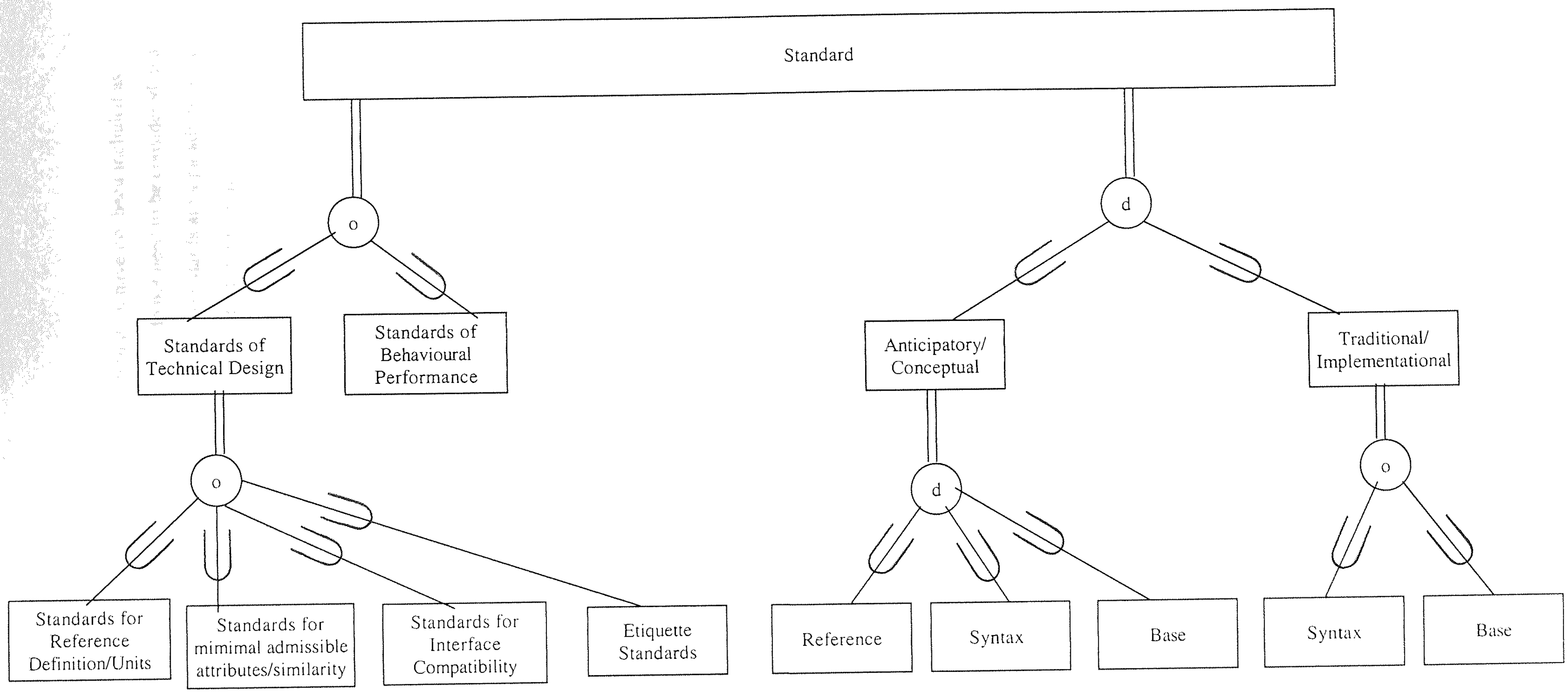

Figure 6.8 An Intermediate Classification of Standards

A New Classification of Standards

In the final model in Figure 6.9 the standards for reference definition have not been included as these are base standards, for example weights and measures, and do not need to be considered in a classification of IT standards. The final model assumes that all IT standards are a paradigm for a product of process and therefore all technical IT standards are standards for minimal admissible attributes – although the use of the word minimal has consequences for interoperability and this might be better termed as standards for 'admissible' attributes. David and Kerchemer's classifications of standards for interface compatibility and etiquette standards are classifications which are based on attributes of the standards. The same standards could also be syntax or base standards.

In the final model the standards are divided into 2 specialisations. On the right-hand side of the model the specialisation consists of only one subclass for anticipatory/conceptual standards. This is a particular specialised subclass of the supertype 'IT Standard'. It is a single subclass as only a particular group of standards will be specialised into this group – the anticipatory/conceptual standards. On the left hand side of the diagram the specialisation is initially divided into the two subclasses of 'Standards of Technical Design' and 'Standards of Behavioural Performance', which were from David's original classification. This work has dealt mainly with the standards of Technical Design as this is where the majority of IT technical standards will fall. The subclass of standards of technical design is specialised into the subclasses of Base Standards, Syntax Standards, and Reference Standards. The author has taken the view that the standards for reference definition are not necessary for IT as they are dealing with basic items, for example, weights and measures and therefore are not included. The new model assumes that all technical IT standards are a paradigm for a product or process and therefore are all standards for minimal admissible attributes. Therefore the sub-class of standards of minimum admissible attributes has not been included. David and Kerchemer's classification of standards for interface compatibility and etiquette standards are classifications which are based on attributes of the standards and do not need to be placed into a separate classification. In the new model there are three subclasses in the specialisation of standards of technical design. These are base standards, syntax standards and reference standards. (Although Cargill states that reference standards are nearly always anticipatory – in this model it allows for the fact that they may not be.) The new model avoids the repetition of the syntax and base standards which occurred in the intermediate model.

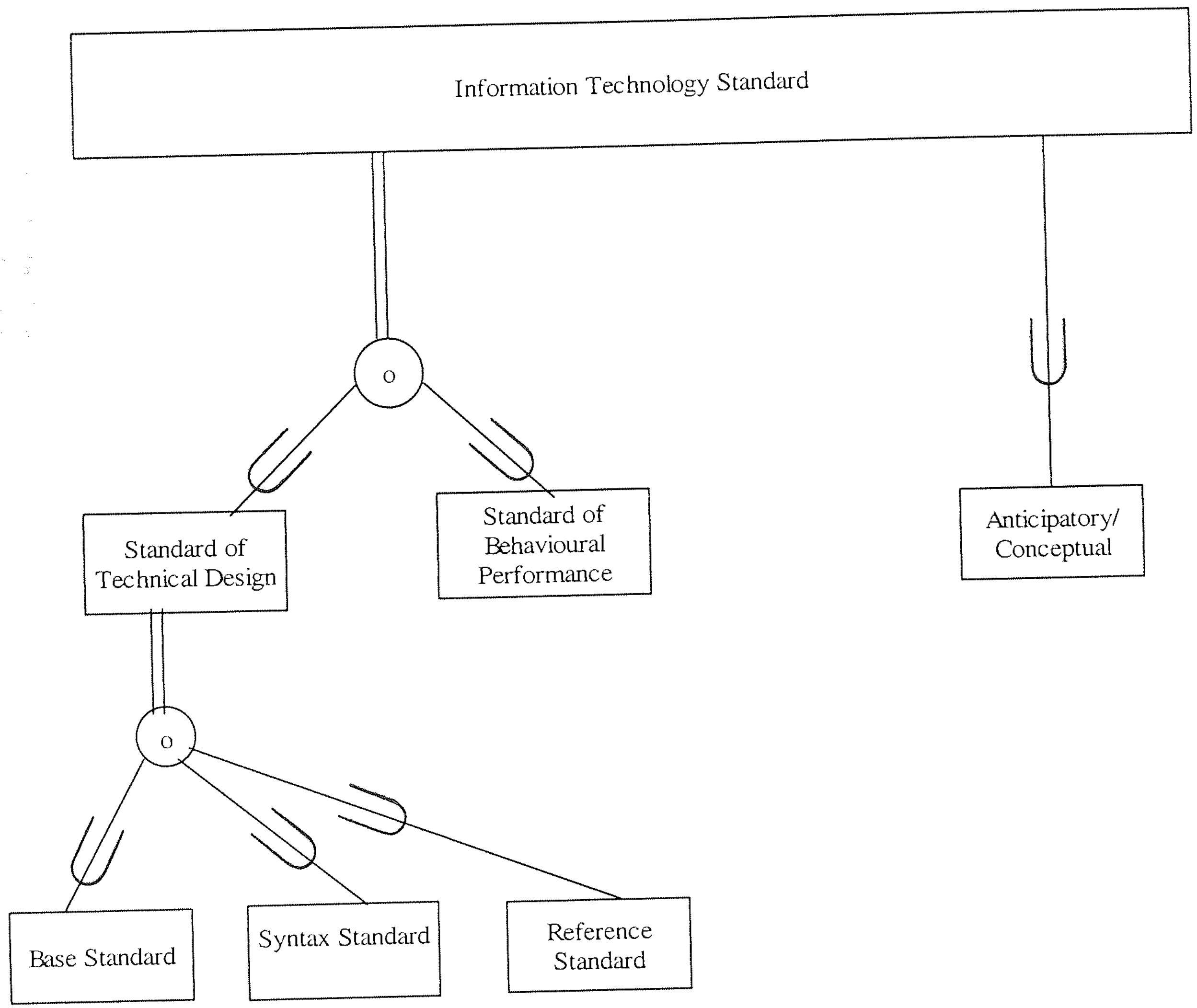

Figure 6.9 A New Classification of Standards

Standard/Product	Classifications
ANSI IRDS	Standard of Technical Design, Anticipatory/Conceptual, Reference
ISO/IECCOBOL	Standard of Technical Design, Base
ISO/IEC IRDS	Standard of Technical Design, Anticipatory/Conceptual, Reference
IRDS SERVICES INTERFACE	Standard of Technical Design, Anticipatory/Conceptual, Syntax
ISO/IEC OSI	Standard of Technical Design, Anticipatory/Conceptual, Reference
ISO/IEC SQL	Standard of Technical Design, Base
PCTE	Standard of Technical Design, Anticipatory/Conceptual, Reference, Syntax
CORBA	Standard of Technical Design, Anticipatory/Conceptual, Syntax

Table 6.7 Classification of Standards/Products

Chapter Review

The IRDS Framework standard which is the subject of this work needs to be placed into a classification. From the earlier descriptions of standards classifications the IRDS standard is an anticipatory standard. The IRDS Framework Standard can be placed in the reference standard subclass and will inherit the attributes from the 'standard of technical design' superclass. It will also be placed in the anticipatory/conceptual specialisation on the right-hand side of the model. The Services Interface Standard can be placed in the syntax standard subclass and will inherit the attributes of the 'standards of technical design' superclass. It will also be placed in the anticipatory/conceptual specialisation on the right-hand side of the model. Table 6.7 above shows standards and products mentioned in this work and their possible classifications.

In this chapter a synthesis of previous classifications and models of standards has been carried out to produce a new model. A modelling technique has been used which is semantically rich and can show different specialisations of standards. The new model shows a classification of standards which combines the previous work in the area and also has the advantage of being able to capture the fact that a standard may be a member of more than one classification. It is necessary to have a model/classification of standards in order to organise how we think about standards and to produce a manageable picture of the standards arena. The next chapter deals with the economic processes which affects the diffusion of standards.

CHAPTER SEVEN: THE DIFFUSION OF STANDARDS

This chapter will examine the diffusion of standards in a compatibility market in order to better understand the diffusion of the IRDS standard. One of the research questions raised in chapter one was: "What are the underlying forces which drive the diffusion of standards". By the diffusion of a standard is meant the relative success of the standard, how quickly it is adapted and in what number. In chapter 6 the author stated that work on standards falls into one of three areas; the administrative standards development process, the classification of standards and the diffusion of standards. This chapter will examine the diffusion process of compatibility standards and the economic issues which affect the progress of standards in order to examine the diffusion of the IRDS standard.

In the past 20 years economists have begun researching Information Technology standards with a view to analysing the economic issues raised. [Berg & Schumny 1990] [David & Greenstein 1990], [David 1987], [Katz & Shapiro 1985], [Farrell 1990], [Farrell & Soloner 1985]. Compatibility is a major consideration in the IT market because often when purchasing products the user has to consider whether those products are compatible with other products. However, for other industries this will not be a consideration. For example in the motorcar industry when a person purchases a car, the car can then be run and maintained as a discrete item. It does not have to interface with other cars in any way and the number of cars sold of a particular model has no technical effect on the person purchasing or driving their current car. However with the computing and Information Technology industry people will want to purchase hardware that can be linked to other hardware and software that can be run on various machines and be exchanged between machines. Therefore it is important to purchase hardware and software that is compatible with other users' machines. Hardware, software and interfaces need to be compatible. In this way the economics of the Information Technology industry differs from other industries in that one user's value for a product increases when another user has a compatible product. In the economics literature this is referred to as a positive network externality [David and Bunn 1988]. There are other industries as well as the IT and ICT industry where compatibility and network externalities are an issue. For example the electricity industry and the telecommunications industries. In these industries compatibility standards are necessary - these may be standards for the products themselves, standards for components or standards for linking products together. This chapter considers the economic theories which have so far been applied to the economic behaviour of different categories of compatibility industries and their standards.

Compatibility Markets

In a communications market users will prefer compatibility. For example the user of a compact disk player will want to be able to play his/her compact disks and other available compact disks on his/her machine and on any other compact disk player they may buy in the future. In this case all the available compact disk machines are compatible as are all the compact disks themselves. Braunstein and White [1985] describe in generic terms the possible advantages and disadvantages of overall compatibility. Their work attempts to describe the factors which would have to be considered if it were necessary to carry out a cost-benefit analysis of compatibility. One of the advantages of compatibility is that if products, or product components are compatible then there are savings made from not having to produce gateway or converter devices to link incompatible products together, and, there are also savings to be made from not having to produce duplicate machinery. (In the compact disk example only one type of compact disk player is produced.) However a possible disadvantage of compatibility is that it does mean that the industry may lose some products which had unique, useful characteristics which are not present in the main product. A recent example of this is the Betamax video player which was claimed to be of superior quality to the VHS recorder which later became the standard. Also in a compatibility market a firm may start off with an advantage over other firms. If one firm has the installed base for the compatible product, and patents for the product, that firm may establish a dominant position in the marketplace and other firms may find it difficult to compete. (This was the argument made in the recent Microsoft Anti-trust case.) After considering the advantages and disadvantages of a compatibility market Braunstein and White conclude that achieving compatibility can be worthwhile. Although they do point out that if the cost of products is low compatibility may not be worthwhile pursuing, for example, if the price of compact disk players was very low people may be prepared to buy two different types that were not compatible.

Economides [1989a] carried out an economic analysis of the incentives which firms have to provide individual components which are compatible with the components of other manufacturers (for example producing a computer keyboard which will plug into any computer) rather than producing components which are their own proprietary products and incompatible with other products (producing a keyboard which will only plug into the producer's proprietary computer). Economides' definition of incompatible is where a system is incompatible with other systems and cannot be broken down into components (for example the Apple Mac PC) whereas compatible is where components are produced that have the same specification and are therefore compatible with other manufactures components. Using a duopoly model where he compares extreme situations of full compatibility and complete incompatibility (using the definitions of compatibility and incompatibility described above) and he concludes that profits are higher under the compatibility regime. In an incompatible model of systems, competition will be fierce and this will force down prices, whereas in a compatibility market prices and profits will be higher.

David and Bunn [1988] have also defined different types of compatibility. For example a computer from one manufacturer can be compatible with another if they use the same operating system, can run the same software and can be connected in a network. David and Bunn term this type of compatibility 'compatible complements'. On the other hand components from different manufacturers are compatible if their designs are co-ordinated enabling them to work together. That is something in the design has to be changed in order to make the component compatible. They use as an example of this the Microsoft Word word-processing program which has two versions one for the IBM PC and one for the Apple Macintosh computer. They refer to this type of compatibility as compatible substitutes. Therefore there are different degrees of compatibility and different definitions of compatibility.

The law of diminishing marginal utility

One of the traditional economic theories of market behaviour is the law of diminishing marginal utility. This law states that if a commodity is in demand and the supply of the commodity increases so does the satisfaction derived from it. However, there will come a point when the marginal utility (satisfaction) derived from the commodity becomes progressively less. The additional satisfaction derived from any additional unit is called the marginal utility. The fact that usually as the quantity of a commodity increases, the marginal utility decreases is known as the law of diminishing marginal utility. For example if more people demand organic food, the output of organic food will increase and the number of purchases will increase, so the price the marginal purchaser is prepared to pay will fall. Therefore the economic returns from producing organic food will decrease, as the number of producers increase (all other variables being equal). This theory has long been associated with traditional industries like the motorcar industry where the more others are producing a particular type of car the less attractive it is to you to do the same. However, in a compatibility standards driven environment the economic principle of decreasing returns or diminishing marginal utility is not relevant and often in a compatibility market, the more others are using or producing a product "the <u>more</u> it pays you to do so -- at least up to a point" [Farrell 1990 p195] In fact the situation with compatibility standards is almost completely opposed to the economic theory of diminishing returns and could be described as one of increasing returns. As an example of the differences between a traditional market and a compatibility market Farrell(1990) describes three situations which can occur with the traditional law of diminishing returns but which do not occur in a compatibility market. These are, firstly in a traditional market all tastes are likely to be satisfied, secondly small-scale experimentation is encouraged and thirdly the industry does not have to continue to use an out-of-date product and can easily change to new products. These three situations are described in more detail as they occur in a traditional decreasing returns market, but do not hold in an increasing-returns compatibility market.

All tastes are likely to be satisfied.

In a decreasing returns market all tastes can be satisfied. For example garlic-flavoured ice cream can be produced if some people want it (Farrell's example) or Reliant Robin motorcars can still be produced if there is sufficient demand for them. Therefore in a decreasing returns market extreme tastes can be catered for. However in the compatibility markets of Information Technology and

Information Communication Technologies it will be difficult to satisfy all tastes. For example, it will be difficult for an operating system which only has a small market share to survive, as it needs to rely on having compatible software and components. The component products and software may not be available or may not be economic to produce. In the computing industry at the moment there is one major PC operating system and a standard suite of software which runs on it. If there were another competing operating system for PC's there would need to be a bulk of software which would run on it and this would reduce the compatibility between personal computers as there would be two incompatible operating systems.

Small scale experimentation

With a diminishing returns market small-scale experimentation is encouraged as this is profitable and there is little risk involved. A recent example of small-scale experimentation is the success of the "bagless" Dyson vacuum cleaner which was invented by one man. As people replaced their old vacuum, they could easily switch to a Dyson, as the only compatibility issue was an electrical socket. However in the IT and ICT markets small-scale experimentation if not usually profitable due to the compatibility issue with other products, and therefore risk is involved. The major exception to this was the Apple Macintosh PC's which survived with their own operating system and software for many years after the success of the IBM PC and the IBM clones. However today the Apple Mac has had to build in interfaces to the PC standard.

The industry can change to new products.

Thirdly with a diminishing returns market with no compatibility issues changes in what people want can be accommodated. The industry does not have to continue to use an out-of-date product but can change to new products. For example everyone could change to a "bagless" vacuum cleaner if they wanted to as there are no compatibility issues to consider. As a consequence of this the diminishing returns industry does not tend to get locked in by history to inappropriate products and can change to new products. With IT the industry can get "locked-in" to a sub-optimal product and it is difficult to change. An example of sub-optimal lock-in is described in a later section of the chapter.

In summary the traditional conditions of diminishing marginal utility will not always apply in a market where compatibility is an issue and traditional economic theories cannot always be applied in an increasing returns compatibility market, of which IT and ICT is an example.

Categorisation of Standards for economic purposes.

For the purpose of discussing the diffusion and therefore the success of a standard it is useful to categorise the standards according to their origins. Two obvious classifications for the purpose of economic modelling are de jure standards and de facto standards. De jure standards are formal standards, which are decided on by a committee of experts and interested parties as outlined in the description of Standards Organisations in Chapter two. Although the term "de jure" implies that the standards have the force of law behind them, standards which are government sponsored and standards which emanate from formal standards bodies are both usually referred to as "de jure" standards. De facto standards are standards which have grown out of widespread use, have not been publicly developed, and often start as the product of one company. The economic conditions which influence the life and diffusion of a standard depend on the origins of the standard itself. De jure standards will usually have different economic conditions to the de facto standard.

The rise of a de facto standard is usually a case of the best selling, best marketed, or more easily available product becoming the industry standard. The use of the de facto standard could mean vendor lock-in for the purchasers of the product, and, not just for the original product, but also for the other products that may link to it. A classic example is the windows interface. Although originally developed by Apple it was taken up by Microsoft who now dominates the market for the operating system and interface for PC's as well as for the software which runs on it (the MS Office suite). The windows interface is a de facto standard. The programming language Java and the relational database language Structured Query Language (SQL) are both examples of standards which started as de facto standards and later have, in SQL's case, become a de jure standard and in the case of Java is in the process of becoming one.

Unsponsored De Facto Standards

David and Greenstein [1990] divide de facto standards into two groups – sponsored and unsponsored de facto standards. Unsponsored de facto standards occur where no one person or company has a major influence over the product and therefore there is no one company that has the power to influence the choice of a majority of the users. An example of an unsponsored de facto standard is the emergence of the QWERTY keyboard as the dominant keyboard layout for typewriters and then subsequently for computer keyboards. When the typewriter was invented in c1880 there were several different competing typewriting machines each of which had technical differences in how they operated. The QWERTY layout was devised as an attempt to slow the typists down as the keys had a tendency to jam if operated too quickly. Touch-typing came at a later stage and the first touch-typing method was on a Remington Qwerty Keyboard, however there were other touch typing methods on different keyboard layouts. Even though the "key-jamming" problem was later solved people still continued to use the inefficient QWERTY layout. David [1985] gives three economic explanations as to why Qwerty became locked in as the dominant keyboard layout. These are; compatibility between products, economies of scale and the quasi-irreversibility of investments.

Compatibility between products

Technical interrelatedness is described as the need for compatibility between the hardware (the typewriter in the QWERTY case) and software (the typist's knowledge of the arrangement of the keys). The value of the typewriter depended upon the availability of typists who could touch type on that particular keyboard. In terms of the IT market the value of the hardware is related to the software that is available to run on it. The value of many software products today is dependent on the number of users who are trained in using the software. Because of the compatibility issues between hardware and software it is difficult for the industry to change to new products.

Economies of scale

Economies of scale are explained as, the more people who bought typewriters with QWERTY layouts increased, the greater the likelihood that more typists would choose to learn QWERTY. The overall costs of a QWERTY system would decrease as it became more popular relative to other systems. This would eventually lead to de facto standardisation. David explains this phenomenon by describing a generalised "Polya urn scheme" [David 1985]. In this example an urn contains balls of different colours. One by one balls are taken out of the urn. Every time a ball is drawn out it is replaced in the urn along with another ball of the same colour. The probability that balls of a particular colour will be added increases. If the drawing of balls goes on indefinitely it is likely that all the balls in the urn will eventually be the same colour. This can be juxtaposed with the economic explanation of why Qwerty became the dominant keyboard layout and is an explanation of the effect of increasing returns in the area of compatibility standards. The employers did not have a preference for a particular keyboard layout but were more concerned with the touch typists and which layout they preferred. If typists decided at random which touch method to acquire, every selection for Qwerty would increase the probability that the next selection would be Qwerty. There may be many rivals in the process and at the beginning it is difficult to predict which one will eventually be dominant. (Many of us were faced with a similar dilemma when deciding whether to buy a Betamax or VHS video machine in the 1980's). The eventual success of a product (having all the balls, or nearly all the balls in the urn) may in the end be brought about by "historical accidents - the particular sequencing of choices made close to the beginning of the process". [David op cit p 335]

Quasi-irreversibility of investments.

It was less expensive for non-Qwerty typewriter manufacturers to change the keyboard layout than it was to retrain the stock of typists. It was difficult to reverse the investment which had already been made in training the stock of people who could touch type on a Qwerty keyboard. Therefore as more typists chose Qwerty it had an effect on the amount of Qwerty keyboards produced. There was a recent attempt in the 1970's to topple QWERTY from its dominant position by the Dvorak Simplified Keyboard (DSK). The DSK keyboard layout had been proved to increase typing speeds by a considerable amount and was widely advertised. However although DSK was obviously the superior layout it has not replaced the QWERTY keyboard.

Summary

The Qwerty keyboard is an example of an unsponsored compatibility standard which followed the law of increasing returns. David states that the reason why Qwerty became the standard was probably influenced by the sequence of choices made in the initial stages which, like the Polya Urn, encouraged a band wagon effect for Qwerty. In this particular case the industry eventually became locked into a sub-optimal product. Attempts to change it with the DSK keyboard were unsuccessful because of the quasi-irreversibility of investments – a large stock of people were trained on the use of the Qwerty keyboard for touch typing and this 'investment' was quasi-irreversible. With this example of an unsponsored de facto standard in an increasing returns market the industry became locked into a sub-optimal product. The three factors mentioned, compatibility between products, quasi-irreversibility of investment and economies of scale are all prevalent in an increasing returns market of which IT is an example.

Sponsored De Facto Standard

With the example of the Qwerty keyboard there was no major sponsor or producer of keyboards in the market but several different types of keyboard layouts. Eventually (similar to the Polya Urn process) Qwerty emerged as the dominant keyboard layout. There are situations where emerging standards may have several sponsors in the market place. In this case rivalries can develop between the different sponsors which may have an effect on the market. There have been many recent examples of this, for example the race for an Internet browser standard between Microsoft (Internet Explorer) and Netscape. An example of a sponsored de facto standard is the development of the relational database query language SQL. Several relational database languages were created in the early 1970's as a response to Codd's work on databases [Codd 1969], [Codd 1970] and the need to develop a language for the relational model. One of the languages was SQL which was part of IBM's System R project. SQL was then implemented in two of IBM's products. Other vendors also brought out SQL based products. By 1986 there was an ANSI standard for SQL [X3H2 ANSI]. C J Date [1985] criticised the ANSI standard stating that the standard was "virtually identical to IBM SQL". Codd [1985] and [Darwin 1991] also criticised the ANSI standard and the erosion of relational theory which was taking place in implementations of relational databases. C J Date also criticised the standard as being "the intersection of existing implementations" [1987a] and also heavily criticised the standard in other publications. [Date 1987b] [Date 1990] [Darwen & Date 1995]. Although there were other competing relational languages developed at the time, IBM's SQL based on System R got a head start and other vendors rushed to implement their own versions of it and to claim that their products were relational. From an increasing returns point of view IBM had more 'balls in the urn' to start with and therefore had a better chance of its product becoming the de facto standard. From a historical analysis of the diffusion of the Qwerty Keyboard a new query language for relational databases will not be successful because of the irreversibility of investment factor.

IBM's version of SQL became the de facto standard. It was not developed in a standards organisation nor developed by a consensus approach between vendors or organisations but was originally sponsored by IBM. Because the relational model was new and appeared to have obvious advantages over older models there was a rush from vendors to implement and market databases based on the relational model with an appropriate query language. There is now an international de jure standard for SQL [ISO/IEC 9075] and work is being completed on the draft for an SQL3 standard. However some of the original limitations of SQL have still not been solved [Date 1998] and similar to the QWERTY keyboard the industry has locked into a sub-optimal product.

Therefore SQL had two main problems; firstly it was not, according to the experts in the field, the best relational language and secondly there were many different versions of SQL among different vendors' products. The industry ended up with a sub-optimal language with several limitations and a variety of implementations of it. Attempts to replace it are likely to be unsuccessful because of the irreversibility of investment factor and technical interrelatedness. This clearly illustrates some of the problems of a de facto standard.

The Open Systems movement was an attempt to provide non-proprietary standards which were available to everyone. It was driven by the need for compatibility between systems and the desire to avoid suppliers locking customers into their products. In an Open Systems environment there would be less chance of a competing sub-optimal product becoming the prevailing standard as theoretically the standard will not belong to any one vendor and will be available to anyone who wants to use it.

An interesting study of the effect of Open Systems (although not originally intended as an open system) is the success of the IBM Personal Computer and its clones in the mid 1980's. [Grindley & McBryde 1990]. IBM, who had fallen behind in the PC market, put together an architecture for a microcomputer by buying in parts from various suppliers (for example the operating system, MSDos, was commissioned from Microsoft). IBM named the microcomputer 'Personal Computer' (PC). The IBM PC was then imitated by various other suppliers, for example Compaq and Tandy). IBM did not have any patents and did not make any sustained attempt to stop the IBM clones. At the time IBM would not have known how successful the PC would become, and, as most of the parts were bought in from other suppliers it would have been difficult to license and patent. The PC was an immediate success brought about by the fact that it was an open architecture which could be copied by other vendors. Other microcomputers which were in the market at the time were squeezed out and the IBM PC architecture became the de facto standard for PC architecture. IBM did try to reclose the market by introducing a new computer, the PS/2, which was an attempt to tie users into the IBM range of products. This is an example of how a sponsor of a standard may not want to encourage compatibility among its products and other vendors' products. If the sponsor has a large installed base, there will be a large network of the users of the product and it is to the sponsor's advantage if he can tie those users in. However as the PS/2 was not entirely compatible with the PC and the PC market had become so large users were wary about tying themselves into a proprietary product. Because of the irreversibility of investment factor which affects the diffusion of a product the PS/2 was not successful in reclosing the market to IBM's advantage.

Summary

Two examples of de facto standards, Qwerty unsponsored and SQL sponsored, have shown that in both cases the industries in question have become locked into sub-optimal products, and, because they are in an increasing returns compatibility market it is difficult to change to better suited products. In an increasing returns market it is not necessarily the best-suited product that becomes the most widespread. On the other hand the diffusion of the architecture of the IBM PC is an example of how successful an open standard can be.

Gateways and Converters

Economides [1989a] considered the extreme cases of complete compatibility and complete incompatibility in a market. He compared cases where firms provide individual components which are compatible with the components of other manufacturers (complete compatibility) with cases where firms produce components which are their own proprietary products and incompatible with other products (complete incompatibility). He was concerned with the economics of compatibility and incompatibility. After economic analysis he concluded that prices and profits for the firm will be higher in a compatibility environment. He did not introduce the effect of the use of gateways and converters in a compatibility market. The use of gateways or converters to provide compatibility between different products is a situation which may occur either before a leading standard has been established in the market or may occur after the standard has been established. [David & Bunn1988] A definition of gateway and converter technologies is that they:-

> "overcome some pre-existing technical incompatibility which would otherwise prevent some components from being formed into an integrated production system". [David & Bunn 1988]

The use of gateways and converters after standardisation may help to reduce the social cost of failing to standardise in the first place. If an existing user has failed to standardise on a product, the use of a gateway or converter device to make his products compatible with the standard would be a suitable way to use the standard without the expense of purchasing new products. Therefore gateways and converters can be produced in order to link to an existing standard by a user who has failed to standardise. A vendor who has a product which has become incompatible with the standard may produce his own gateway or converter device which he then sells with his own product. Or, an industry may develop whose sole function is to provide gateways. [David and

Greenstein 1990]. An example of a gateway device is the electrical adapter which the British will have to use if they want to use their own electrical products (for example a hairdryer), in Europe. The adapter provides a gateway between the two incompatible systems. In this particular case the cost of the gateway device is minimal. Also, vendors of products which would otherwise be incompatible with other products can offer a gateway to connect their products to others. A recent example of a gateway device is the use of open database connectivity (ODBC). This software provides a gateway between various databases. For example the Microsoft Access database can be linked, using ODBC, to an Oracle RDBMS. The Microsoft Access database can then use the concurrency and locking facilities of the RDBMS as well as linking to tables in the Oracle RDBMS. In this case two incompatible versions of a relational database can be linked together.

The use of gateways and converters introduces a new element into previous theories on complete compatibility and incomplete compatibility. If there are firms with incompatible but similar products vying for market leadership the 'invisible hand' of market forces may be thwarted by others who develop and market gateway or translator devices to link between the incompatible products. The gateway product may tip the balance in favour of one of the technologies leading to the de facto standardisation of that product, or, the gateway may hinder the standardisation completely and no standard emerges [David and Bunn 1988].

Although gateway and converter technology may reduce some of the loss from not having standardised earlier, there may come a point when users immediately consider a gateway or converter product rather than an emerging standard and therefore the failure of that standard is more likely. Converters and gateways may be anticipated or may be made available too soon. Research has been carried out on the economics of converters and how they affect standardisation [Economides 1989b], [Farrell & Saloner 1989]. David and Greenstein [1990] point out that although the use of converters can reduce the social cost of failing to standardise they can also increase the likelihood of the standard failing. If converters or gateways are introduced too soon users may find it easier and cheaper to use a converter rather than standardise.

De jure standards

In the previous section the discussion has centered on the economic conditions which can prevail with de facto standards. De facto standards have been categorised as either unsponsored de facto standards or sponsored de facto standards. The other category to be considered is de jure standards. These are usually considered as standards which have been developed in formal traditional standards bodies either national or international (for example ANSI and ISO). Some de jure standards may be the result of a de facto standard which has become the accepted standard and is being formally specified by a formal standards organisation. An example of this type of standard is SQL, previously mentioned, which now is on version 3 of an ISO standard and continues to evolve along with new technology. Other examples are programming language standards which continually evolve. In these cases, where a standards body formally accepts a previous de facto standard, the economic battles for supremacy have already been won or lost and a standard has been established. In which case the diffusion of the standards will have the same patterns as described earlier in the section on de facto standards. However since the 1980's the international standards bodies have also begun to produce anticipatory standards – standards created in advance of the technology in order to try to guide the market, as mentioned in the classification of standards in Chapter 6. In these cases there may be no economic factors to consider, as the products are not yet in the market place and may have no installed base. The IRDS standard is an example of an anticipatory standard as was the OSI standard. By trying to produce open standards ahead of the technology the standards bodies could attempt to guide the market and also avoid some of the problems of sub-optimal lock-in which can occur with de facto standards.

Chapter Review

The IRDS standard was an anticipatory standard which was defined with a reference or framework standard followed by more detailed specifications for services and/or interface standards. The ISO/IEC IRDS standard did not have an installed base and is an example of an anticipatory standard developed entirely in a formal standards organisations with no obvious commercial sponsor. It was also an open standard and was available for anyone to use. The only cost involved was in purchasing the standard on paper. This chapter has examined the diffusion of standards in a compatibility market in order to better understand the diffusion of the IRDS standard. The next chapter covers the diffusion of the IRDS standard from when it became a full international standard in 1990.

CHAPTER 8: THE DIFFUSION OF THE IRDS STANDARD

This chapter covers the diffusion of the ISO/IEC IRDS standard from when it became a full international standard in 1990. This chapter also considers the diffusion of other repository standards; the ANSI IRDS Standard, AD/Cycle, PCTE, the diffusion of the OSI Reference Model and the beginnings of second generation repositories. In the early 1990's there appeared to be a race towards a repository standard, the ISO/IEC IRDS standard was published in 1990 and the Services Interface in 1993. In the early 1990's IBM was working on AD/Cycle, and PCTE was in the process of becoming an ISO/IEC standard. There were many proprietary repository products on the market. Bernknopf [1990] provides a list of over 40 proprietary repository tools some of which had stated that they would conform to either IBM AD/Cycle, ISO/IEC IRDS or ANSI IRDS standards when the standards matured. More recently, in 1997, Microsoft have produced the Microsoft Repository and the OMG have produced the Meta-Object facility (MOF) which is also a repository product.

Diffusion of the ISO/IEC IRDS Framework Standard

The IRDS standards, both ISO and ANSI, did not made the progress which was predicted. Many people thought that the IRDS standards would diffuse quickly and that dictionaries which did not conform to the ISO/IRDS standard would be out of the mainstream within a few years. [Cashin 1989] [Hitchcock & Brown 1990] [Spurr 1990]. The ISO IRDS was an anticipatory standard developed almost entirely within the ISO and its working groups. There was no installed base for ISO/IRDS and therefore no products on the market. As there was no installed base there were "no balls in the urn" to allow it to even begin a bandwagon diffusion process. It was an unsponsored standard in that there were no vendors who were marketing it. Also it may have been hindered by the fact that the standard was not freely available but had to be purchased from, in the case of the United Kingdom, the British Standards Institute. It was nevertheless an 'Open' standard which was not tied to any proprietary solutions and did not involve purchasing any royalties or licences. The initial framework standard for IRDS was published in 1990 and it was not until 1993 that the first detailed standard in the family, the Services Interface Standard, was published. The initial reference standard was, and was intended to be, a broad and general document. However the fact that there was a three-year gap between its publication and the publication of more detailed information did not help get the IRDS accepted.

In the early 1990's at the time that the ISO IRDS standard was published there were also developments in the uptake and rise in popularity of Integrated CASE (I-Case) tools and Component CASE tools. The parallel rise in CASE tools and the rise in standards for repositories did have an overlap [Byrne & Shah 1994]. I-CASE tools, which were defined as tools which could help to automate all stages of the systems development life cycle, needed to operate in a repository environment, and, component case tools theoretically could map into a repository to share information with main systems. However, the major vendors of CASE tool repositories did not claim any conformance with the IRDS standard. Oracle Corporation was one of the main vendors of an I-CASE tool that was repository based. The Oracle* CASE tool was an I-CASE tool which offered near full life cycle support for systems development. The I-CASE tool consisted of a repository called CASE*Dictionary which could be run as an extra layer above the Oracle DBMS. Information on stages of the systems development life cycle could be stored in the repository. Drawing tools were available for Data Flow Diagrams, Function Hierarchy Diagrams, Entity Relationship Diagrams and a Matrix diagrammer. These were not stand-alone diagrammers. Information entered through a diagrammer was stored, through a forms based interface, in the repository database. For example information entered in an Entity Relationship Diagram could be accessed through entity description forms and attributes and data types could be added to the entities. The entity relationship diagram could also be mapped to a set of tables ready for customisation or creation in the underlying DBMS. Although the link between the repository and the underlying DBMS was a passive link, the link could be made active in order to create the tables in the underlying database. However the link was passive in that any changes made in the DBMS tables would not be automatically propagated to the repository, although there were facilities to make the link active and reconcile the DBMS with the repository and vice versa. Version control was available at the application level in the early versions of the tool. The whole of the CASE*Dictionary could be partitioned into different applications although objects from one application could not be copied from another. Applications could be shared between users using read and write access controls. The user could retrieve earlier versions of an application but this was the only level of granularity available. Applications could be copied and renamed. In the early versions of the Oracle*CASE Tool there was no provision for Impact Analysis but Impact Analysis facilities did gradually become available in later versions. A facility for unique naming within an application was enforced automatically. Owners of an application could grant read or write access to other users or groups of users.

Comparing the functionality of the Oracle*CASE tool with the ISO IRDS functionality for a repository, reproduced below from Chapter four, it can be seen that the Oracle*CASE tool supported most of the functionality of a repository as specified in the ISO IRDS standard.

The architecture of the Oracle*CASE tool was a four level architecture as described in the ANSI and ISO IRDS Framework standard. The Oracle*CASE tool also provided menus which allowed the user to extend the repository at the meta data level which was one of the requirements of a repository mentioned in Chapter four. The Oracle*CASE meta model manual also specified a complete mapping of the dictionary using CDIF (CASE Data Interchange Format Standard) which was the data interchange standard which was being supported by ISO [Oracle Corporation 1992].

FACILITIES FOR INFORMATION RESOURCE MANAGEMENT

Naming	A facility for the unique naming of objects and for imposing constraints on the naming.
Status of Dictionary Content	The user should be able to distinguish between data which is no longer used (archived), data which is unstable and data which is stable.
Information System Life Cycle Management	The IRDS should be able to support the idea of phases in a Life Cycle. The life cycle of an object can be split into a number of phases. This is useful for controlling the development and use of information systems.
Version Control	Users should be able to create and maintain versions of objects, or groups of objects and be able to use different versions of an object if required.
Partitioning	For convenience the IRD may be divided into partitions. A partition is a logical subdivision of an IRD or an IRD Definition. The standard states that an object can be in only one partition, but other objects representing different versions of the object may exist in other partitions.
Copy Creation	Users should be able to make copies of an object or group of objects in order to avoid duplication of work.
Impact Analysis	An IRDS should be able to answer queries concerned with impact analysis. Particularly the concepts of dependency of one object on another. If one object is modified you should be able to analyse which other objects are dependent on it and which may be affected.
Subsetting	It should be possible to define subsets of an IRD or an IRD Definition. This is to control user access and provide each user with their own context. Each subset provides update access to one partition only and optional read-only access to other partitions.

Table 8.1 Facilities for Information Resource Management

However the Oracle*CASE tool manuals and accompanying books did not claim any conformance with ISO or ANSI IRDS. [Barker 1990], [Oracle Corporation 1991 & 1992]. The other major integrated repository based CASE tool on the market at the time was Information Engineering Facility (IEF) from Texas Instruments. This tool also supported a four level architecture and had similar functionality as the Oracle CASE*Tool.

CASE tools did not make the expected advances in automating and speeding up the development of software. It began to be realised that automated tools brought with it another set of problems. There were problems with the introduction of CASE tools into organisations. The tools did not always meet expectations and did not seem to justify the amount of investment which had been made in them. Mathiassen and Sorensen [1996] argue that the introduction of CASE tools into an organisation should be deferred until the organisation is at a suitable level of maturity in its software processes and that automating an ineffective existing process with a CASE tool is not going to produce an effective outcome. They use the Capability Maturity Model (CMM) [Humphrey 1989] as a framework for the introduction of CASE tools into an organisation and state that a high level of maturity of an organisation's software processes is necessary to achieve productivity gains from the use of CASE tools. By the time the Services Interface Standard was released in 1993 some doubts were beginning to arise about the effectiveness of tools used to automate the software development process and this may have affected the diffusion of the standard.

Chapter 7 discussed the economics of compatibility standards and introduced gateways and converters. The use of converters after standardisation of a product may help to reduce the social cost for those who failed to standardise on the product in the first place; as they may not have to replace existing systems but can use a gateway or converter. Vendors of products which would otherwise be incompatible with other products can offer a gateway to connect their products to others and this has become increasingly common in IT. Gateway technology can be described as

> "some means (a device, or a convention) for effectuating whatever technical connections between distinct production sub-systems are required in order for them to be utilised in conjunction, within a larger integrated production system".[David & Bunn 1988]

By 1997 there had been a rise in middleware products. Middleware made it possible for users to access the applications and the data they needed regardless of differences in platforms, operating systems and protocols. Also the 'larger integrated production systems' referred to by David & Bunn above had become 'larger <u>distributed</u> production systems' and middleware was used to bring about the integration. Middleware can be described as a layer of software residing between the user's application and operating system which conceals the complexities of networking protocols, database dialects, and operating system differences. [NDISD 1998]. By the mid 1990s there was a need to integrate disparate applications from legacy systems. Middleware services are to help solve connectivity and interoperability problems across a network with heterogeneous platforms. With the growth in client/server architecture it was also necessary to enable mainframe applications to migrate to client/server systems. There were also new technologies which had matured, for example the internet, intranets and object oriented systems. The OMG had had continued success with its Common Object Request Broker Architecture (CORBA) and the OMG today claim that CORBA is : "the de facto standard for ensuring open, distributed computing". The OMG CORBA standards are middleware standards for an Object Oriented environment. The OMG home page [http://www.omg.org] describes CORBA as the "Middleware that's Everywhere" through world wide standard specifications. The middleware products provide comparable function to the converters and gateways described by David & Greenstein [1990]. David & Greenstein's research concluded that although the use of converters can reduce the social cost of failing to standardise they can also increase the likelihood of the standard failing. If converters or gateways are introduced too soon users may find it easier and cheaper to use a converter rather than standardise.

However Middleware products differ from the definition of gateway and converter technology. The Middleware products are not connecting to any type of standard, or possible standard, but are connecting and providing gateways to other applications through a brokering service. Whenever there is a need to connect heterogeneous systems together the use of gateways and converters in the form of middleware is anticipated and there is no attempt to standardise on a global model for sharing and exchanging information. David and Greenstein [1990] argued that gateways or converters could be provided by vendors who have products which have become incompatible with the standard, or, an industry may arise whose sole function is to provide gateways. Today there is a proliferation of middleware services many of which use proprietary implementations and thereby make applications dependent on a vendor's product. It was cheaper, easier and quicker for developers to produce middleware products then to consider using a shareable repository to define the information resources of the organisation.

Work continued separately on the ANSI version of an IRDS standard and the ISO/IEC version of an IRDS standard. The ISO never approved the ANSI standard and continued with the ISO/IEC IRDS framework. Both standards were similar and had a considerable overlap however ANSI and ISO could never reach a compromise and converge the two standards. If there had been one standard rather than two similar, but competing, standards there could have been more chance for a successful diffusion of an IRDS standard. At the time of writing ISO support for IRDS is being withdrawn and there is no further development work being done within the ISO committees on the family of IRDS standards. The ANSI IRDS standard is also in the process of being withdrawn.

Diffusion of AD/CYCLE

IBM announced the development of Application Development Cycle (AD/Cycle) in September 1989. AD/Cycle was set to become the application development platform for the whole range of IBM equipment from mainframe to PC. It would be based on CASE technology and form an I-CASE tool with a common repository for systems development across a range of IBM platforms. Other major manufacturers already had CASE tools on the market or near completion (Oracle*CASE and IEF for example) and at the time IBM had fallen behind in the CASE market. AD/Cycle was an extension to IBM's System Application Architecture (SAA) which was launched in 1987 with the aim of achieving application portability across IBM's varying architectures and operating systems [Richards 1990].

The repository of AD/Cycle was to be the first deliverable. As this was a huge project and IBM were committing vast amounts of money and resources to AD/Cycle it was assumed by many that AD/Cycle would become the industry de facto standard for repositories [Richards 1990, Gillenson & Frost 1993], and that component CASE tool manufacturers would have to conform to AD/Cycle standards and be compatible with the AD/Cycle repository. However a few years into the project IBM withdrew funding and halted any further development on AD/Cycle. As far as the author is aware IBM did not release any specific details as to why work on AD/Cycle had been withdrawn. Therefore as AD/Cycle was never released as a complete product it did not start a diffusion process. As it was developed by IBM it would have had a large installed base in IBM products but there were other problems which prevented IBM from continuing work on it.

Portable Common Tool Environment (PCTE)

The Portable Common Tool Environment (PCTE) was described as a "standard for a public tool interface for an open repository" [Jowett & Wakeman 1993]. Similar to IRDS PCTE describes a model for (framework) and operations to manage (services interface) what is described as "an open CASE repository". Both IRDS and PCTE were repository standards however they were based on different paradigms and had different models. PCTE was part of the IPSE movement which used the programming paradigm. ISO IRDS has its roots firmly in database standards and uses SQL as its data model. PCTE became an ECMA standard in December 1990 [ECMA 149] and in June 1993 it was submitted to ISO/IEC using the fast track submission route and became an ISO standard in 1994. [ISO/IEC 13719] A PCTE working group was created in sub committee 22 of JTC1. As stated in Chapter 5 PCTE was regarded as a competing standard to IRDS [Bird 1994] and there were some publications comparing PCTE and IRDS[Oliver 1990] [Olle 1992] as well as work within ISO/IEC JTC1 on improving compatibility between the two standards. The ISO JTC1 view was that there could be a rapprochement between the two standards with the possibility of an underlying repository having a PCTE interface and an IRDS services interface (using SQL) [Gradwell 1993]. However, the PCTE standard has not been successful in providing an open CASE repository. PCTE had a small installed base. The original implementation was Emeraude version 12 and there were a small number of other implementations from proprietary vendors [Sagols 1992] as well as implementations linked to the original ESPRIT project. However it did not have the successful diffusion which was hoped for and at the time of writing is no longer in use in any wide-scale capacity.

Diffusion of the OSI Reference Model

Although the OSI Reference Model had considerable exposure combined with the backing of the main standards organisations and the support of some governments it did not have the success that was expected.[Hansethe et al 1996], [Kalin and Barber 1992],[Egyedi 1996],[Egyedi 1997], [Sloane 1999]. Many of the OSI protocols were anticipatory in nature and did not have an installed base. When referring to the unsuccessful diffusion of the OSI standard Hansethe et al [1996] describe the situation of not having a market, or not being close enough to the market as "installed base hostile". Kalin and Barber [1992] describe the diffusion and take up of the OSI standard as being slow and uneven. They attribute this to the fact that because the standard was anticipatory in nature the people developing the standard would add features which were not essential but that they felt might be useful in the future. Cargill [1995] also described this facet as anticipatory standards having "maybe" bits in them. This had the result of making the standards vague and unspecific in places which in turn would make it difficult to test to ascertain if products conformed to the standard. Kalin and Barber examine how the transport protocol in the OSI layers, which began as an operational non-proprietary service, resulted in 5 different transport layer standards which could not interoperate (and also took 10 years to produce). Whereas the Internet TCP/IP protocols, which perform a similar function to the OSI standards, but were not de jure standards, have been hugely successful.

Kalin & Barber [1996] and Egyedi [1997] also argue that the OSI standards were slow to produce as compromises had to be made by different participants as to how the future market would develop. [Egyedi 2000] refers to anticipatory standards as "ex ante" standards, (that is standards developed looking to the future where a current implementation of the standard may not be extant). She describes the TCP/IP architecture as "ex post" (the standard was derived out of a working solution). Although the OSI standard was designed to develop "non-proprietary, widely useable and future-proof solutions for compatibility problems" [page 232] it did not gain market dominance. She concludes that OSI was not successful because:-

- it had no substantial installed base,
- the standard was too comprehensive with too many options which made it difficult and expensive to implement,
- it was difficult to provide interoperability because of the various options,
- it was difficult to have testing procedures because it was described as an implementation independent standard
- and the slow process of the traditional Standards Development organisations meant that the standards were produced too slowly and were, in some cases, overtaken by other products which were available.

The OSI Reference Model and the IRDS Framework Standard had similarities. In the model of standards developed in Chapter 6 both standards fall into the category of anticipatory reference standards and both standards were developed in traditional standards development organisations. The OSI standard had definite specifications with implementational detail for some of the levels in the model, and, work was ongoing on further specifications for the different levels. The OSI standard did incorporate some protocols which were already extant but much of the work within the development of OSI was on new protocols.

Review of the Diffusion of the ISO/IEC IRDS Standard

By 1995 neither the ISO IRDS, ANSI IRDS, PCTE nor AD/Cycle had become the dominant standard for a repository. The use of gateways or converters to provide compatibility between different products is a situation which may occur either before a leader has been established in the market or may occur after the standard has been established. In this particular case no market leader had been established in the repository market and middleware products became dominant. Also the CORBA products which were based on Object Oriented technology were gaining in popularity. In 1996 work had begun on an amendment to the Services Interface Standard to provide a CORBA Interface Definition Language (IDL) binding. However CORBA as middleware had became widely diffused.

By 1997 component-based development had emerged as a new approach to systems development. With this approach software is developed from collections of components and sub-components which can be re-used. The components communicate through well-specified interfaces. The advantages of component based development was that components could be re-used, they were easy to maintain and test and the use of components could help to increase productivity in software development. However although there are advantages in component development the full advantages can only be realised if the development environment is repository based [Byrne and Golder 1998] [Kara 1998]. There is little value in having reusable components if you have no way of cataloguing those components or locating different versions of the components and the links to other components.

The IRDS standard was published at a similar time to the growth in popularity of CASE tools. As ISO IRDS used SQL as the data model it seemed likely that it could be used as the basis for a relationally implemented repository. Although the repository could be the basis of an Integrated CASE tool, CASE tools were not mentioned in the first version of the ISO/IEC IRDS standard.

In summary from this research it is inferred that the reasons why the IRDS was not successful in providing the basis for Information Resource Dictionaries were:

- **It had no installed base**. The IRDS standard has been classified as an anticipatory reference standard. The IRDS standard was developed entirely within the ISO committees and national bodies standards organisations and had no installations and no commercial sponsors. Therefore it could not begin a diffusion process as it had 'no balls in the urn' to start with in order to do so. It also had no large commercial backers who were prepared to introduce it into their products, although some relationally based I-CASE tools had similar characteristics. In some respects it is a contradiction in terms to talk about an anticipatory standard having an installed base, because if the standard is being developed for a future use it will not have an installation. However some of the more successful consortia bodies mentioned in Chapter 2 have developed successful anticipatory standards by ensuring that the standard always has an implementation.

- **The Framework standard was too general**. The ISO/IEC IRDS Framework Standard was intended to be a general and broad document. It was published in 1990 and it was three years later when the next standard in the family, the Services Interface Standard was published. The Services Interface Standard contained more specific detail but also stated that it was an implementation independent standard. As the IRDS Framework was a generic document it was not possible to have any conformance testing. The Services Interface Standard contained a list of general services and data structures which an IRDS must support in order to conform to the standard. It was not until 1993 that products could claim to conform to the Services Interface standard and the IRDS Standard. The three-year gap between the Framework and the Services Interface did not help in the diffusion of the standard.

- **The standard was overtaken by new technology**. The IRDS standard was overtaken by new technology. Firstly object oriented based products became popular. Object-oriented and object-relational databases were receiving much hype in the computing press. The IRDS standard, because of its foundations in relational databases, and the definition of the Services Interface data structures in relational tables, appeared to have fallen behind. A CORBA binding for the Services Interface was in the process of being developed in 1996 but this did not help to get the Services Interface Standard accepted. Secondly middleware, based on OO technology, and backed by the OMG was becoming widely used. Middleware provided a cheaper, easier and more expedient way to link applications together as opposed to the global infrastructure provided by an IRDS.

- **There were two versions of an IRDS Standard.** ANSI and ISO could not reach a compromise and converge the two different versions of the IRDS standard and work continued separately on the two different versions.

The Microsoft Repository - second generation repository

The Microsoft Repository was launched in 1997. Microsoft, Texas Instruments and Platinum Technologies have contributed to the design and architecture of the Microsoft Repository. The Microsoft Repository is distributed with Visual Basic 5.0 and Visual Studio 97. Also independent software vendors can licence it from Microsoft and embed it in their applications. The repository engine is based on an open database connectivity (ODBC) relational database but it is an object-oriented repository. The main goals of the Microsoft repository were to provide compatibility with Microsoft's ActiveX object architecture which consists of the Component Object Model (COM) - a middleware service - and also to provide extensibility for customers to add their own extensions [Bernstein et al 1997]. Microsoft did not expect end users to see any improved productivity from the use of the repository until early 1999, by this time third-party tools and independent software vendors will have had time to implement the repository.

According to Microsoft [http://microsoft.com/repository] in 1998 there were two competing repository standards: the Microsoft Repository and the OMG Meta-object Facility (MOF) standard. Microsoft hope that their Repository will become pervasive because of the wide use of the Microsoft development tools. The components within the repository are based on Microsoft's Component Object Model (COM). COM is a standard that describes component-to-component calling conventions (middleware). Components written in one language can call components written in other languages. Component Object Models were based on object-oriented technology, using classes, multiple interfaces, methods, unique identifiers, interface definitions, aggregation and different interfaces for classes.

The OMG has a common metadata interchange standard called Common Warehouse Metamodel (CWM) [http://www.omg.org/cwm/]. The Oracle Corporation IS one of the contributors to the CWM standard. The CWM has three objectives each of which is concerned with interchange specifications and mechanisms:

- Establishment of an industry standard specification for common warehouse metadata interchange
- Provision of a generic mechanism that can be used to transfer a wide variety of warehouse metadata.
- Maximised Leverage of existing vendor-neutral interchange mechanisms.

From the above objectives it can be seen that the CWM is primarily an interchange standard. The Microsoft Repository has also been described as merely an interchange vehicle because its emphasis is on well defined (middleware) interfaces which allow components to communicate. In which case both the Microsoft Repository and the CWM could be described as primarily interchange standards. The Oracle*CASE tool which was an example of an integrated case tool which had a similar architecture and implementation to the IRDS standard and services interface standard, is now called Designer 2000 and uses CORBA technology to extend the models in the repository. If the repository needs to be extended in order to map to other tools and software CORBA is used as the exchange mechanism. The initial meta model manual for the Oracle CASE tool [Oracle Corporation 1992] had a complete mapping of the model to CDIF, however Designer 2000 now uses the Extensible Markup Language (XML), for data interchange.

The market has developed in the direction of interchange standards. Products can share information by having a common interchange standard. The common repository model for sharing data outlined in ISO/IEC IRDS has not been realised. Rosen [1991] stated that the interchange standard should be an intermediate stage for tools to exchange information but in the long term information should be shared through a common repository model. It was realised that the development of a repository standard was critical for the data integration of CASE tools [Bernstein & Dayal 1994] and that repositories would benefit from more research by the database community. Bernstein [2000] in an article titled "Is Generic Metadata Management Feasible?" commented that after 30 years research on meta data related problems:

> "we have so far been unable to offer general purpose database technology that factors out the similar aspects of these tools into generic database infrastructure."

This particular case study examined the diffusion of an anticipatory reference standard developed in a traditional standards organisation. This particular standard was not successful because firstly, it had no installed base which is necessary for the diffusion process to start. Secondly, it was overtaken by new technology. Thirdly the market for converters (middleware) increased thus hindering the standard from diffusing. In the late 1980's traditional Standards Development Organisations were moving towards creating anticipatory standards as a way of coping with the fast growth in new technology in the computing industry, in the case of the ISO/IEC IRDS Framework Standard this approach was not successful.

CHAPTER NINE: CONTRIBUTIONS OF THIS RESEARCH

In the opening chapter it was stated that this work is a case study and that the case study as a research method is a form of empirical enquiry. This work is based on the interpretive paradigm and the work is concerned with making a reading of a situation in order to gain understanding. The research has used a longitudinal case study of the development and diffusion of the IRDS standard in order to draw inferences about the standards setting arena and the use of anticipatory standards in formal standards setting organisations. This chapter summarises the contributions of this research to the field of standards development.

In chapter one it was stated that this work will answer the following research questions:

1 What is the role of formal standards bodies in the current standards arena and what is the current process of producing standards within these bodies.
2 What current models of standards and classifications of standards are available for understanding the standards process?
3 What are the underlying forces, which drive the diffusion of standards?
4 Are the development of anticipatory framework standards in traditional standards organisations suitable for the computing industry, with particular reference to the progress of the ISO/IEC IRDS Framework Standard.
5 To what extent has the IRDS standard been successful in providing a common basis for the development of Information Resource Dictionaries?

Research Question 1

What is the role of formal standards bodies in the current standards arena and what is the current process of producing standards within these bodies?

In Chapter 2 the administrative process for creating standards in the ISO and IEC have been outlined. This process has been compared with the production of standards in the less formal consortia bodies. The role of formal standards bodies appears to be changing. Whereas formal standards bodies used to be the only producers of standards, in recent years consortia have developed which appear to be capable of producing standards more quickly. In some cases standards produced by consortia are later submitted to formal standards bodies for their stamp of approval. Chapter two outlined the measures which the formal bodies have produced to attempt to speed up the standards development process. In particular the introduction of the PAS system which has allowed some consortia to submit their standards to ISO through a fast track system and also by using email and electronic communications among committees to speed up parts of the consensus process. The role of formal standards organisations is changing as they are now accepting and formalising standards from consortia bodies.

Research Question 2

What current models of standards and classifications of standards are available for understanding the standards process?

In Chapter 6 previous work on models and classifications of standards have been described in chronological order of their development. Some of the models were specific for Information Technology and ICT whereas some of the models were generic. The work on previous models has been synthesised to produce a new classification of standards which overcomes some of the limitations of the previous models – in particular the inability to show that standards can overlap various classifications. The new classification produced in this research is the first classification that can illustrate in diagrammatic form that classifications of standards are not mutually exclusive and that a standard can fall into more than one category. Previous classifications have struggled with representing this. This new diagrammatic form of classifying standards uses a modelling technique used for semantic data models such as the extended entity relationship model. The model can show that standards can overlap various categories. The new classification also provides a uniform way of illustrating the different classifications which have been produced by previous authors, and this was illustrated in chapter 6. In this work the process of developing standards (whether in formal SDO's or in consortia) has been separated from the classification of standards in order to achieve a clearer view.

Research Question 3

What are the underlying forces, which drive the diffusion of standards?

Chapter 7 discussed previous work on increasing returns and compatibility markets. In compatibility markets users derive benefits from other users joining the same 'network' and this is referred to as positive network externalities. However, because of this it is possible that in the diffusion of a standard a bandwagon effect can take over leading to one product becoming the de facto standard. This product may not necessarily be the best solution. Chapter 7 outlined two examples where a bandwagon effect had taken over in the diffusion of a standard: - the sub-optimal standards of the Qwerty keyboard and the relational database language SQL. By understanding the economic processes which influence the diffusion of a standard it may be possible to avoid sub-optimal standards becoming de facto standards in the future. The underlying forces which drive the diffusion of standards are firstly; the standard must have an installed base and secondly the providers of the standard should be aware of positive network externalities in a compatibility market and how they affect the diffusion of the standard.

Research Question 4

Are the development of anticipatory framework standards in traditional standards organisations suitable for the computing industry, with particular reference to the progress of the ISO/IEC IRDS Framework Standard?

During the production of the new classification of standards carried out in this research, one of the classifications identified was the classification of anticipatory/conceptual standards. The IRDS standard was placed in this category. Framework and reference standards are usually anticipatory standards (standards developed ahead of the technology). The development of anticipatory standards was seen as a possible way forward for the formal standards bodies to keep abreast of changes in IT. By creating standards ahead of the technology the standards would act as "change agents" and guide the market. This was intended to alleviate the fact that the formal Standards Development Organisations were having problems keeping up with the changes in the computing industry. The anticipatory standards could also be seen as one way of addressing the problem of arriving at sub-optimal de facto standards. If the industry can be guided before the technology develops this will encourage the use of optimal products.

However, the IRDS standard, the subject of this work, did not have an installed base and the diffusion of the standard never got started. It did not have the sponsorship of any large vendor, even though there were similar products at the time. From this it can be generalised that anticipatory standards should already have an implementation and an installed base. Some of the consortia bodies, the IETF, the OMG and W3C have had success with anticipatory standards. However they have always ensured that the standard has an implementation before proceeding.

In the new classification of standards the reference anticipatory standards pose a problem as because they are reference standards they are conceptual in nature and will not have an installed base. In the family of IRDS standards the services interface standard did have implementational detail and could have been installed. From this a generalisation can be made that any reference standards should be published with implementational guidelines. If the services interface standard had been published with the IRDS Framework standard it is possible that it could have had more success, however there still would not have been an installed base. Also it could be said that standards organisations should not issue standards which do not have an existing implementation. Comments have been made that the ISO has lost influence since it began introducing anticipatory standards [Cargill 1995] [Rada 2000] [Soley 1995]. Although the development of anticipatory standards was an attempt by traditional Standards Development Organisations to move on from what Cargill [1989] describes as the two-stage model, where an existing product is taken and formally standardised, the ISO/IEC IRDS standard has not been successful. Two successful consortia the OMG and the IETF developed anticipatory syntax and base standards which were successful. In the OMG's case the standards were developed before any large vendor had proprietary solutions. The successful anticipatory standards have come from organisations where the users and providers of the standards have been closely integrated and involved from the start. The process followed in the IETF and the OMG for developing standards is similar to the process recommended by Cargill [1989] in the five-stage model where the producers and the users of standards are more closely involved.

Research Question 5

To what extent has the IRDS standard been successful in providing a common basis for the development of Information Resource Dictionaries?

The IRDS standard has not been successful in providing a common basis for the development of Information Resource Dictionaries. The standard was overtaken by new technology; object oriented technologies in particular, CORBA, and also the use of gateway and converters in the form of middleware hindered the development of a repository standard. In the industry at the moment there is no common repository model.

The industry moved towards an interchange standard. Products can share information by having a common interchange standard. Although there was an exchange standard available, CDIF, which was supported by the ISO and the IRDS family of standards, the CDIF standard has not been successful and at the moment vendors are developing their own proprietary standards or gateways to communicate with other tools.

Middleware reduced static efficiency loss by cutting the cost of incompatibility by providing gateways between products. Products can be compatible when their designs are co-ordinated in some way, enabling them to work together. When products are incompatible gateways and converters will be used in order to bring the products together into an integrated system. Before the rise of middleware in the late 1980's and early 1990's the emphasis was on compatibility between products and many standards, for example IRDS, OSI and PCTE pursued this aim. However, with the inevitable growth in legacy systems, and the rise in new technologies, the aim of having compatibility with a common model was overtaken by the rise in gateway and converter technology which was called middleware. There were a large variety of different technologies and legacy systems which could interoperate through middleware. In Chapter 7 it was stated that encouraging the development of gateway devices (converters, adapters etc) linking otherwise incompatible systems can help reduce efficiency loss, but it can also prevent a common standard from maturing.

Contributions and Conclusion

Therefore the overall contributions of this research are:

Contribution 1 – a new classification of standards has been produced.

Contribution 2 – It is concluded that anticipatory reference standards should have an implementation and installed base. Referring to the new classification of standards produced in contribution 1 above, the author would state that, by generalising from this case study, the class of anticipatory reference standards should have an implementation and a guaranteed installed base if they are to start a diffusion process in the market.

Contribution 3 - The use of a longitudinal case study as the research method has enabled the author to document this phase of repository technology and place it in the history of computing.

The IRDS Framework Standard can be described as an anticipatory or ex ante standard. In the new classification of standards introduced in Chapter 6 it falls into the categories of an anticipatory/conceptual standard and a reference standard. Anticipatory standards face the problem of anticipating future requirements. The long standards development process in traditional Standards Development Organisations can result in anticipatory standards being overtaken by advances in technology, which occurred with the IRDS family of standards. Also, because of the compromises which have to be made at sub-committee level to accommodate the different viewpoints from members of world-wide national standards bodies, the standards can often result in vague and wide ranging specifications, which are subsequently difficult to implement. The IRDS Framework Standard did not have an installed base. It is possible that, similar to the OSI Reference Model, the sub-committees hoped that government procurement would help to produce an installed base. However, this did not occur. The market developed in the direction of interchange standards. A common repository model for sharing data, as outlined in the ISO/IEC IRDS Framework Standard, has not yet been realised. It is possible that the ISO/IEC IRDS standard, with suitable backing, could have provided the "generic database infrastructure" to support a generic meta data model. However, a new repository race is now being fought between Microsoft, with the Microsoft Repository, and Oracle and the OMG's CWM in a further attempt to find the "holy grail" of a common repository model.

The research method chosen was a longitudinal case study based on empirical enquiry, where empirical is taken to mean based on experiment, observation or experience. The author was in a position to follow the progress of the IRDS standard over an 11-year period and to document the findings in the form of a case study. The author believes that it is possible to generalise from a single case study or multiple case studies and produce theoretical propositions. This research used a case study of the development and diffusion of the ISO/IEC IRDS standard to draw inferences about the development of certain types of standards in formal standards setting organisations. In order to do this a new classification of standards for Information Technology was produced and the IRDS standard was placed in the categories of anticipatory/conceptual and reference standards. The longitudinal study has also helped to place this phase of the development of repository standards in the history of computing.

The global model/repository outlined in the IRDS standard did not come to fruition. The proliferation of different incompatible systems meant that it was easier to have middleware to exchange data rather than have the global model for data sharing. It is possible that the availability of middleware came at the wrong time and hindered the development of a global metamodel. The industry now has an interchange system as prevalent. With the proliferation of different systems which had developed by the 1990's this was the most expedient solution. Gateways and converters overcome some of the static efficiency loss in the short term but in the long term have meant that there has not been any standardisation on a global model.

If a similar situation arises in the future it will be necessary to look at the lessons that have been learnt from the past. Anticipatory standards have to be developed quickly and have an installed base if they are to guide future technology otherwise sub-optimal standards may become prevalent or the use of gateways and converters will result in no standard being formed. The ISO/IEC IRDS standard could have been a solution to the problem of managing meta data and linking and sharing information between heterogeneous systems, but for the reasons outlined in this research it did not have the success that was anticipated.

REFERENCES

[Baer 1992] "4GLs and CASE Tools: More Promised Than Delivered" *Managing Automation*, Vol 7, Issue 9, 1992

[Barker 1990] "CASE*Method - Entity Relationship Modelling" Barker, Richard Addison-Wesley 1990

[Berg & Schumny 1990] "An Analysis of The Information Technology Standardization Process proceedings of the International Symposium on Information Technology Standardization" , Berg J.L, Schumny H, Editors Elsevier Science Publishers B.V. The Netherlands 1990.

[Bernknopf 1990] Bernknopf, Jeff "Repository Race Getting crowded" Software Magazine July 1990 p 39-46.

[Bernstein et al 1997] Bernstein P, Harry B, Sanders P, Shutt D, Zander J "The Microsoft Repository" in proceedings of the 23rd VLDB Conference, Athens, Greece, 1997

[Bernstein, & Dayal 1994] Bernstein P, Dayal U "An overview of Repository Technology" *Proceedings of the 20th International conference on Very Large Data Bases (VLDB)* Chile 1994

[Bernstein 2000] Bernstein P "Is Generic Metadata Management Feasible?". Panel topic in *Proceedings of the 26th International conference on Very Large Data Bases (VLDB)* Egypt 2000

[Bird 1994] Bird, Barry "The Significance of IRDS in a PCTE World" Proceedings of the PCTE'94 Conference San Francisco Nov 1994

[Bonino 1991] Bonino M J, "An exploration of anticipatory information technology standards" Master's thesis, University of Pittsburgh, Department of Information Science, Pittsburgh PA, 1991.

[Bonino & Spring 1991] Bonino M J, Spring M B "Standards as change agents in the information technology market" Computer Standards & Interfaces volume 12 (1991) pages 97-107 (Also re-published in Computer Standards & Interfaces volume 20 (1999) pages 279-289)

[Braa & Vidgen 2000] Braa Kristin, Vidgen Richard, "Research: from Observation to Intervention" Chapter 12 in *Planet Internet* Sorensen C, Braa K, Dahlbom B (editors) Studentlitteratur, Lund, Sweden. Previously published in *Accounting, Management and Information Technologies,* Issue 9, pages 25-47 Elsevier Science Ltd 1999

[Braunstein & White 1985] Braunstein, Y, White, L, "Setting technical compatibility standards: an economic analysis" *The Antitrust Bulletin* 1985 30 Part 2 pp337-355

[Brooks 1975] Brooks Fredrick p Jr, "The Mythical Man-Month - Essays on Software Engineering" Addison-Wesley Pubishing Company 1975

[Byrne & Shah 1994] Byrne Bernadette M, Shah Hanifa "Information Resource Dictionary System Standards and CASE Tools" Proceedings of the 5th workshop on the next generation of CASE tools (NGCT*94) Utrecht. Memoranda Informatica 94-25.

[Byrne & Golder 1998] Byrne Bernadette M, Golder Paul, "ISO Information Resource Dictionary System (IRDS) standards and support for Component Based Information Systems Engineering". Proceedings of CBISE*98 CAiSE*98 Workshop on Component Based Information Systems Engineering . Grundy J editor. Published by University of Waikato, New Zealand

[Cargill 1989] Cargill Carl F., "Information Technology Standardization" Digital Press MA 1989.

[Cargill 1995] Cargill Carl F., "A Five-Segment Model for Standardization" pp 79-99 Standards Policy for Information Infrastructure edited by Brian Kahin and Janet Abbate. 1995 A Publication of the Havard Information Infrastructure Project.

[Cargill C 1997] Cargill Carl F., " Sun and Standardization Wars" *Standard View* Vol 5 no 4 December pages 133-135

[Casanova et al 1992] Casanova Pietro, Leone Annamaria, Tangorra Filippo, "The Implementation of a standard IRDS repository for the TASIS software engineering environment". *Computer Standards & Interfaces* 14 1992 p. 277-295

[Cashin 1989] Cashin J "Data Model Standards in Competitive Stage" Software Magazine December 1989 pages 78-81

[Chambers 97]Chambers Pocket Dictionary 1997 Chambers Harrap Publishers

[Chen 1976] Chen Peter P "The Entity-Relationship Model - towards a unified view of data". *ACM transactions of Database Systems,* 1, 9-36 1976

[Codd 1969] Codd E. F. "Derivability, Redundancy and Consistency of Relations stored in Large Data Banks", IBM Research Report RJ599 (August 19th, 1969)

[Codd 1970] Codd E. F. "A Relational Model of Data for large shared data banks" *Communications of the ACM* Vol 13 No 6 (June 1970)

[Codd 1982] Codd E. F. "Relational Databases: A Practical Foundation for Productivity" CACM 25, No 2 Feb 1982

[Codd 1985] Codd E. F. "How Relational Is Your Database Management System? Computerworld (Oct 14th and 21st 1985)

[Date 1987a] Date Chris J, "A guide to the SQL standard" Addison-Wesley, Wokingham, U.K. 1987

[Date 1987b] Date Chris J, "Where SQL Falls short" *Datamation* May 1987

[Date 1990] Date Chris J "Relational Database Writings 1985-1989" Addison-Wesley 1990

[Darwin 1991] Darwin H, "The Askew Wall: A Personal Perspective" in Relational Database Writings 1989-91 C J Date with Hugh Darwin. Addison-Wesley Publishing 1991

[Darwen & Date 1995] H Darwen and C Date, "The Third Manifesto", SIGMOD Record 24 (1), Marach 1995

[Date 1998] C J Date "Relational Database Writings 1994-1997] Addison-Wesley 1998

[David 1985] David Paul A. "Clio and the Economics of QWERTY" *American Economic Review* Volume 75 No 2 May 1985 pages 332-337

[David 1987] David Paul A. "Some new standards for the economics of standardization in the information age" pp 206-239 in "Economic policy and technological performance" Dasgupta, Stoneman 1987 editors, Cambridge University Press, NY

[David & Bunn 1988] David Paul A, Bunn Julie A, "The Economics of Gateway Technologies and Network Evolution – Lessons from Electricity Supply History" *Information Economics and Policy* 3, 165-202 North-Holland 1988.

[David & Greenstein 1990] David Paul A, Greenstein S "The Economics of Compatibility Standards: An Introduction To Recent Research" *Econ. Innov. New Tech.* Vol 1 pages 3-41 1990 Harwood Academic Publishers GmbH

[Deming 1982] Deming W. Edwards, "Out of the Crisis" Massachusetts Institute of Technology, Cambride USA 1982

[Dolk & Kirsch 1987] Dolk Daniel R, Kirsch Robert A "A Relational Information Resource Dictionary System" *Communications of the ACM* January 1987 Volume 30 Number 1

[Durell 1985] Durell William "Data Administation – A Practical Guide to Successful Data Management" McGraw-Hill 1985

[Economides 1989a] Economides, Nicholas "Desirability of Compatibility in the Absence of Network Externalities" *American Economic Review* 1989 Vol 79 part 5 December

[Economides 1989b] Economides, Nicholas "Variable Compatibility Without Network Externalities" Center for Economic Policy Research Publication Number 157, Stanford University March 1989

[Egyedi 1996] Egyedi, T.M., "Shaping Standardization: A study of standards processes and standards policies in the field of telematic services". Delft, The Netherlands: Delft University Press. Dissertation.

[Egyedi 1997] Egyedi T. M. "Examining the relevance of paradigms to base OSI standardisation". *Computer Standards & Interfaces* 18 1997 431-450

[Egyedi 2000] Egyedi, T. M. "Institutional Dilemma in ICT Standardization: Coordinating the diffusion of technology?" in *Information Technology Standards and Standardization: A Global Perspective* Kai Jakobs - editor. Idea Group Publishing 2000

[Elmasri & Navathe 2000] Elmasri R, Navathe S B "Fundamentals of Database Systems" Third Edition Addison-Wesley 2000

[Evered & Louis 1981] Evered R, Louis M "Alternative perspectives in the organizational sciences" Academy of management Review 6(3) Pages 385-396

[Farrell 1990] Farrell J, "The Economics of Standardization: A Guide for Non-Economists" in An Analysis of The Information Technology Standardization Process proceedings of the International Symposium on Information Technology Standardization , Berg J.L, Schumny H, Editors Elsevier Science Publishers B.V. The Netherlands 1990.

[Farrell & Soloner 1985] Farrell J and Saloner G "Standardization, Compatibility and Innovation" *RAND Journal of Economics,* 16(1, Spring) pages 70-83

[Farrell & Soloner 1989] Farrell J and Saloner G "Converters, Compatibility and the Control of Interfaces" Working Paper Number 89-130, Department of Economics, University of California, Berkeley, December.

[Gable 1994] Gable G G "Integrating case study and survey research methods: an example in information systems" *European Journal of Information Systems.* Vol. 3 No 2 pp 112-126 1994

[Gibson R B] "The Global Standards Process: A Balance of the Old and the New" Standards Policy for Information Infrastructure edited by Brian Kahin and Janet Abbate. 1995 A Publication of the Havard Information Infrastructure Project.

[Gillenson & Frost 1993] Gillenson M, Frost R "The Evolution of the Meta-Data Concept: Dictionaries, Catalogs, and Repositories" Journal of Database Management Summer 1993 Vol 4 no 3 pp 17-26

[Glass 1991] "4GLs and CASE: What's the Payoff?" *J Systems Software* 1991. 14 pp 131-132

[Goldfine & Konig 1988] Goldfine Alan, Konig Patricia "A Technical Overview of the Information Resource Dictionary System" Second Edition Jan 1988 NSBIR 88/3700 national Bureau of Standards, Department of Commerce, Washington D.C. 202234

[Gradwell 1993] Gradwell, David "A Comparison of PCTE and IRDS Standards" Project PRT 073/00100 (1052) BSI

[Gray 1991] Gray Pamela "Open Systems – A Business Strategy for the 1990s" 1991 McGraw-Hill Publishing Company

[Grindley & McBryde 1990] Grindley P, McBryde R, "Standards Stratege for Personal Computeres" in An Analysis of The Information Technology Standardization Process proceedings of the International Symposium on Information Technology Standardization , Berg J.L, Schumny J, Editors Elsevier Science Publishers B.V. The Netherlands 1990. Page 227-236

[Hanseth et al 1996] Hanseth O, Monteiro E, Hatling M, "Developing Information Infastructure: The Tension Between Standardization and Flexibility" Science Technology, & Human Values, Vol 21 No 4, Fall 1996 pages 407-426 Sage Publications.

[Hitchcock 1990] Hitchcock P, Brown A "A Comparison of Databases for Software Engineering". In Proceedings of Software Engineering 90, Brighton, July 1990. Patrick A V Hall editor.

[Holloway 1990] "Fourth Generation Systems" Simon Holloway (Editor) Chapman and Hall 1990

[Humphrey 1989] Humphrey, W S "Improving the software development process". *Datamation*, 35, 28-30

[Imber 1991] Imber, M "CASE Data Interchange Format Standards" *Information and Software Technology* Vol 33 no 9 November 1991

[Jowett & Wakeman 1995] Jowett Jonathan, Wakeman Lois "PCTE – The Standard for Open Repositories" Prentice Hall 1995.

[Kalin & Barber 1992] Barber, D, Kalin T "Has the OSI opportunity been fully realised?" *Computer Networks and ISDN Systems 25* (1992) 227-239

[Kara 1998] Kara D "The Repository's Role in Component Development" accessed in 1998 from www.cool.sterling.com

[Katz & Shapiro 1985] Katz Michael L and Shapiro Carl "Network Externalities, Competition and Compatibility" American Economic Review, 75(3, June), pages 424-440

[Kavanagh 1991] Kavanagh, John "CASE will change the face of system work" *Computer Weekly* November 14th 1991

[Kindleberger 1983] Kindleberger Charles P. "Standards as Public, Collective and Private Goods" Kylos, 36(Fasc. 3), pp 377-96

[Krechmer 1996] Krechmer Ken "Technical Standards: Foundations of the future" Standard View Vol 4 No. 1 March/1996

[Lee 1991] Lee A S "Integrating positivist and interpretive approaches to organizational research". *Organization Science* 2(4), 342 - 365

[Lehr 1995] Lehr William "Compatibility Standards and Interoperability: Lessons from the Internet" pp 121-147 Standards Policy for Information Infrastructure edited by Brian Kahin and Janet Abbate. 1995 MIT Press A Publication of the Havard Information Infrastructure Project.

[Leong-Hong & Plagman 1982] Leong-Hong B W, Plagman B K "Data Dictionary/Directory Systems Administrtion, Implementation and Usage" John Wiley & Sons 1982.

[Mähönen 2000] Mähönen Petri "The Standardization Process in IT - Too Slow or Too Fast" in *Information Technology Standards and Standardization: A Global Perspective* Kai Jakobs - editor. Idea Group Publishing 2000

[Martin 1986] Martin J "Fourth Generation Languages" Prentice Hall, Englewood Cliffs 1986

[Mathiassen & Sorensen 1996] Mathiassen L and Sorensen C "The capability maturity model and CASE" *Information systems Journal"* (1996) 6, 195-208

[Mitchell 1983] Mitchell, J Clyde "Case and Situation Analysis" *The Sociological Review 31* 1983

[Moad 1990] Moad, Jeff "The Standards Process Breaks Down" *Datamation* September 15 1990

[NDISD 1998] "Middleware" New Directions in Systems Development (NDISD) Seminar Series, Wolverhampton University U.K. 1998

[Navathe & Kerschberg 1986]Navathe Shamkant B, Kerschberg Larry "Role of Data Dictionaries in Information Resource Management" Information & Management 10 (1986) pages 21-46 (This paper was a revised version of the outcome of a panel from the Database Directions III workshop in 1980.)

[O'Gara 1997] O'Gara Maureen "The Analysts View; What the world heard" 1997 *Standard View* Vol 5 C41 Pages 195-197

[Oliver 1990] Oliver, H "An Initial Study of IRDS and PCTE" Hewlett Packard Laboratories Technical Report

[Olle 1992] Olle, T W "A Comparative Review of the ISO IRDS, the IBM Repository and the ECMA PCTE as a Vehicle for CASE tools" in CASE:Current Practice, Future Prospects editors Spurr K, Layzell P, Wiley 1992

[Oracle Corporation 1991] "CASE*Dictionary Tutorial. Version 5" Oracle Corporation 1991

[Oracle Corporation 1992] "CASE*Dictionary Meta-model. Version 5" Oracle Corporation 1992

[Parker 1992a] Parker, B "Introducing ANSI-X3.138-1988: A Standard for Information Resource Dictionary System (IRDS)". 0-816-2620-8/92 1992 *IEEE pp 90-99*

[Parker 1992b] Parker, B "Introducing EIA-CDIF: The CASE Data Interchange Format Standard" 0-816-2620-8/92 1992 *IEEE pp 74-82*

[Rada & Berg 1995] Rada Roy, Berg, John "Standards: Free or Sold?" *Communications of the ACM* February 1995 Vol 38 No 2

[Rada 1995] Rada Roy "Consensus Versus Speed" *Communications of the ACM* October Vol 38 No 10 1995

[Rada 1998] Rada Roy "Corporate Shortcut to Standardization" *Communications of the ACM* January 1998 Vol 41 no 1 pages 11-15 title

[Rada 2000] Rada Roy "Consensus versus Speed" in *Information Technology Standards and Standardization: A Global Perspective* Kai Jakobs - editor. Idea Group Publishing 2000

[Rankine 1995] Rankine, John L "Do we need a New Standards System?" *Standards Policy for Information Infrastructure* edited by Brian Kahin and Janet Abbate. Pages 565-578 1995 A Publication of the Havard Information Infrastructure Project.

[Richards 1990] "AD/Cycle – A New Unofficial Standard?" in Case on Trial, John Wiley & Sons 1990 editors Spurr, K and Layzell P

[Rohde & Niccolai 2000] "Sun confers with licensees over control of Java". Niccolai J and Rohde L, IDG News Service, San Francisco Bureau March 2000. Accessed September 2001 from [http://www.javaworld.com/javaworld/jw-03-2000/jw-03-idgns-sun_p.html.

[Rosen & Fontaines1991] Rosen, B.K., Fontaines, I. (1991) "Guide to Schema and Schema Extensibility" National Institute of Standards and Technology (NIST) Special Publication 500-197

[Rutkowski 1995] pp 594-600 "Today's Cooperative Competitive Standards Environment and the Internet Standards-Making Model " Standards Policy for Information Infrastructure edited by Brian Kahin and Janet Abbate. 1995 A Publication of the Havard Information Infrastructure Project.

[Sagols 1992] Sagols, Germain "Overview of PCTE Standardization" *IEEE* 1992 0-8186-2620-8/92

[Sankar 1991] Sankar Chetan S "Evaluation Criteria for Data Dictionaries" Journal of Database Administration Winter 1991 Vol 2 No 1 pages 1-6

[Sloane A 1999] "Computer Communications – Principles and Business Applications" Second Edition McGraw Hill

[Smith 1989] Smith Craig N "The case study: a vital yet misunderstood research method for management" in Frontiers of Management: Research and Practice edited by Roger Mansfield Publisher Routledge

[Smith 1994]Smith, Mike Computer Standards and Interfaces "Standardization Activities" Vol 16 pages 375-419

[Soley 1995] "OMG: Building Industry Consensus" pp 610-615 Standards Policy for Information Infrastructure edited by Brian Kahin and Janet Abbate. 1995 MIT Press A Publication of the Havard Information Infrastructure Project.

[Spring et al 1995] Spring Michael, Grisham Cristal, O'Donnel Jon, Skogseid Ingjerd, Snow Andrew, Tarr George, Wang Peihan "Improving the Standardization Process: Working with Bulldogs and Turtles" *Standards Policy for Information Infrastructure* edited by Brian Kahin and Janet Abbate. Pages 220-250 1995 A Publication of the Havard Information Infrastructure Project.

[Spring & Weiss 1997] Spring Michael B, Weiss Martin "The Responses Begin" *StandardView* Vol 5 No 4 December 1997

[Spring & Weiss 1995] Spring Michael B, Weiss Martin "Financing the Standards Development Process" *Standards Policy for Information Infrastructure* edited by Brian Kahin and Janet Abbate. Pages 289-320 1995 A Publication of the Havard Information Infrastructure Project.

[Spurr 1990] Spurr K, "CASE Tools: Does the ISO Standard IRDS Provide Sufficient Support" Applied Information Technology Reports "Fourth Generation Systems – their scope, application and methods of evaluation". Simon Holloway editor. Chapman and Hall pub. 1990.

[Stenton 1983] Stenton David, "Dictionaries – not the universal panacea". Data Dictionary Update British Computer Society Database Specialist Group pages 71-83 1983 Editor Geoffrey J Baker.

[Thompson 1992] Thompson A. K "CASE Data Integration: The Emerging International Standards" *ICL Technical Journal* May 1992

[Updegrove 1995] Updegrove, Andrew "Consortia and the Role of the Government in standard Setting" pp 321-349 Standards Policy for Information Infrastructure edited by Brian Kahin and Janet Abbate. 1995 MIT Press A Publication of the Havard Information Infrastructure Project.

[Walsham & Waema 1994] Walsham G, Waema T "Information Systems Strategy and Implementation: A Case Study of a building Society" ACM Transactions on Information Systems, Vol 12, No 2 April 1994 pages 150-173

[Walsham 1995] Walsham Geoff "Interpretive case studies in IS research: nature and method" European Journal of Information Systems (1995) 4 pages 74-81

[Walsham 1996] Walsham Geoff "Organizational secrecy: some proposals for a way forward". European Journal of Information Systems (1996) 4 195-197

[Weiss & Cargill 1992] Weiss Martin, Cargill Carl "Consortia in the Standards Development Process" *Journal of the American Society for Information Science* 1992 43(8) 559-565

[Wertz 1989] Wertz C J "The Data Dictionary – Concepts and Uses". Wellesley, MA: QED Information Sciences Inc.

[Yin 1994] Yin Robert Y, "Case Study Research – Design and Methods" Second Edition. Applied Social Research Methods Series Volume 5 Sage Publications 1994.

Standards

[ISO/IEC 7498] Open Systems Interconnection (OSI) Reference Model 1983

[ISO/IEC 10027] Information Resource Dictionary System (IRDS) Framework. 1990

[ISO/IEC 10728] Information Resource Dictionary System (IRDS) Services Interface. 1993

[ISO/IEC 13719] Portable Common Tool Environment (PCTE), parts 1, 2 and 3. 1994

[ISO/IEC 9075] Information Technology - Database Languages - SQL 1999 Part 1 Framework

[ECMA 149] PCTE Abstract Specification, December 1990

[ECMA 158] PCTE C Language Binding, June 1991

[ECMA 162] PCTE Ada Lanugage Binding, December 1991

Documents from Standards Organisations

[ANSI X3-138-1988] "Information Resource Dictionary System" New York 1988

[BSI Project 1.21.6.8] "UK Proposed Working Draft for the IRDS Framework Revision" Gradwell, D 1996

[ISO/IEC JTC1 N 5746] "The Transposition of Publicly Available Specifications into International Standards - A Management Guide" Revision 1 of JTC1 N 3582 Jan 1999

[ISO/IEC Directives Part 1 Fourth Edition 2001 page 25 2.5.6] ISO/IEC publications also from IEC web page http://www.iec.ch/tiss/iec/Directives-Part1-Ed4.pdf

[CD 13645.2] Committee Draft 13645.2 SC21 N10406 ISO/IEC] "Guidelines for the Design of IRDS Content Modules" Gradwell D (editor)

Electronic References

[http://microsoft.com/repository] accessed in 1998 from the Microsoft home page.

[http://www.omg.org/cwm/] Datawarehousing, CWM™ and MOF™ Resource page accessed September 2000

[http://www.bsi-global.com] The British Standards Institute *Home Page accessed September 2000*

[http://www.ecma.ch] European Computer Manufacturers Association (ECMA)*Home Page accessed July 2000*

[http://www.iec.ch] International Electrotechnical Commission (IEC) *Home page* accessed September 2000

[http://www.ietf.cnri.reston.va.us] The Internet Engineering Task Force *Home page accessed September 2000*

[http://www.iso.ch] International Organisation for Standardisation. *Home Page* accessed July 2000

[http://www.ispo.cec.be/standards/conf97/welcome.html] Global Standards Conference *Home Page* On line proceedings accessed August 2000

[http://www.omg.org] Object Management Group *Home Page* accessed September 2000

[http://www.w3.org] W3C World Wide Web Consortium *Home page* accessed September 2000

Bibliography

[Begg, Fischer, Dornbusch 1984] "Economics" McGraw-Hill Book Company (UK Ltd)

[Denenberg 1988] Denenberg, Ray "Telecommunication Standards" *Book Research Quarterly* Vol 4 No 3 Fall 1988

[Jones 1989] Jones R "SQL - problems with an emerging standard" *Information and Software Technology vol 31 no 1 1989*

[Martin J 1989]Martin James, Information Engineering , Book 1: Introduction . Englewood Cliffs, NJ: Prentice-Hall

[Martin J 1990] Martin James, Information Engineering, Book 2: Planning and Analysis. Englewood Cliffs, NJ: Prentice-Hall

[Norton 1991] Norton Peter, "Inside the IBM PC and PS/2" Fourth Edition. Brady Publishing.

[Thirkettle 1984] Thirkettle, G L Basic Economics, Macdonald & Evans Ltd, 1984