COMPUTING LABORATORY

Modelling Organisational Complexity using the ORDIT Framework

D.F. Poulson, G. Oswald, J.S. Chudge and M.R. Strens

TECHNICAL REPORT SERIES

No. 383 July, 1992
Modelling Organisational Complexity using the ORDIT Framework

D.F. Poulson, G. Oswald, J.S. Chudge and M.R. Strens

Abstract

The ORDIT project holds as a central tenet the importance of assisting designers in considering organisational requirements, and has developed techniques and tools to this end. Conventional systems analysis has largely focussed on defining information processing requirements rather than looking at information technology from a wider perspective. Thus it is common for systems to be created which, although technically sound, do not satisfy the needs of their human operators. All too often technical solutions to problems are created which do not adequately support the way in which the human components of the work system are organised. It is argued that for information systems design to be successful it is essential that these aspects are also taken into account during the analysis and specification stages of product development.
Bibliographical details

MODELLING ORGANISATIONAL COMPLEXITY USING THE ORDIT FRAMEWORK
Modelling Organisational Complexity using the ORDIT Framework
[By] D.F. Poulson [and others]

Newcastle upon Tyne: University of Newcastle upon Tyne: Computing Laboratory, 1992.

(University of Newcastle upon Tyne, Computing Laboratory, Technical Report Series, no.383)

Added entries

UNIVERSITY OF NEWCASTLE UPON TYNE.
Computing Laboratory. Technical Report Series.383
POULSON, D.F.

Abstract

The ORDIT project holds as a central tenet the importance of assisting designers in considering organisational requirements, and has developed techniques and tools to this end. Conventional systems analysis has largely focussed on defining information processing requirements rather than looking at information technology from a wider perspective. Thus it is common for systems to be created which, although technically sound, do not satisfy the needs of their human operators. All too often technical solutions to problems are created which do not adequately support the way in which the human components of the work system are organised. It is argued that for information systems design to be successful it is essential that these aspects are also taken into account during the analysis and specification stages of product development.

About the author

David Poulson: Background in Psychology and Phd in Ergonomics; has been at Hussat Research Institute for 10 years and has an interest in the design and introduction of IT products; has been on Ordit since its inception.

Graeme Oswald: Degree in Mathematical Science; systems analyst and designer at Marconi and later at Ferranti; has been at Mari for 5 years and is the Project Manager for Ordit.

Jarnail Chudge: Philosophy was first degree followed by MSc in Cognitive Science; has been on Ordit for 2 years.

Ros Strens: Phd in Geology; background in computer modelling; has been on Ordit for 2 years.

Suggested keywords

DESIGN METHODOLOGY ENTERPRISE MODELLING
ORGANISATIONAL REQUIREMENTS SOCIO-TECHNICAL SYSTEMS

ORDIT

Suggested classmarks (primary classmark underlined)
Dewey (18th): 621.381958 651.8
U.D.C. 681.322.02 681.3.01
Modelling Organisational Complexity using the ORDIT Framework

By Poulson D. F., Oswald G., Chudge J. S. and Strens M. R.

1. Introduction

ORDIT (Organisational Requirements Definition for Information Technology) is a five year project (ESPRIT Project No 2301) which includes collaborators from both industry and academia. These are the HUSAT Research Institute, MARI Computer Systems Ltd, The University of Newcastle upon Tyne, Work Research Centre Ltd and Algotech srl.

The ORDIT project holds as a central tenet the importance of assisting designers in considering organisational requirements, and has developed techniques and tools to this end. Conventional systems analysis has largely focussed on defining information processing requirements rather than looking at information technology from a wider perspective. Thus it is common for systems to be created which, although technically sound, do not satisfy the needs of their human operators. All too often technical solutions to problems are created which do not adequately support the way in which the human components of the work system are organised. It is argued that for information systems design to be successful it is essential that these aspects are also taken into account during the analysis and specification stages of product development.

The project has the objective of creating a practical design methodology and supporting tools which will enable those engaged in systems analysis and design to consider in detail the human requirements of complex socio-technical systems. The following paper provides an overview of this methodology and design infrastructure, but focusses on describing the notation being developed and highlighting what it has to offer the IT designer.

2. The ORDIT Philosophy

Many of the difficulties encountered in the design of IT products can be attributed to designers not adequately taking human issues into account, rather than to limitations in technology creating difficulties (eg Hirshheim and Feeney
1986, Harker et al 1990). The incorporation of organisational requirements has always been important for successful system implementation, but it is only in recent years that this has become an explicit focus of attention for designers. To some extent this change has been market led since IT systems are no longer confined to single users, to self contained local operations or to the data processing department, but are now commonly enterprise-wide or even shared by more than one enterprise. Recent developments in computer supported co-operative work (CSCW) mean that users in a multiplicity of roles in widely distributed sites are interacting directly with each other via computers and electronic communications as never before. A system which fails to fulfil its aims thus may prejudice the effectiveness of the entire organisation.

The ORDIT philosophy is based on the belief that it is important to consider human as well as technical issues in the design of IT products, and design is viewed as being the construction of relevant socio-technical systems. Socio-technical theory developed out of the work of organisational theorists such as Emery and Trist (1965), who discovered that many of the issues that determined the success or failure of work systems were social in nature rather than specifically attributable to the physical working environment. Emphasis was placed on the need to consider work in social as well as technical terms, and, for change to be effective, it was argued that both human organisation as well as technology need to be considered. The early research in this field did not investigate settings where information technology was being applied, but more recently the value of using a socio-technical approach has also been demonstrated in this area (e.g. Pava 1983, Mumford 1986, Eason 1988). However, it is not sufficient to create only a framework whereby such issues can be identified, as designers also need a suitable language with which to discuss the human requirements of socio-technical systems, and to demonstrate how these are linked to the technical features of the system design.

3. The Problem of Complexity

The design of large and complex systems requires a methodology which allows the formation of consensus between a diverse set of actors, handles the complexity of future systems and defines and makes explicit the policies of interested parties. In analysing traditional, life-cycle based approaches, we have concluded that, although these methods tend to be good for the development of small "closed" products, they are much less effective when tackling either "large
systems" or small systems which have a major impact upon the organisation(s) for which they are being developed. In our opinion, there are two radically different types of complexity that need to be handled when designing large and complex systems. These are the technical complexity of the system, and the organisational complexity.

3.1 Technical Complexity
The handling of technical complexity is fairly well understood. The principle tool here is abstraction, the first step being abstraction away from detail, allowing the designer to concentrate on the overall picture, with more and more detail being added as the design process proceeds.
Design of a large system cannot be undertaken by one person and this has led to a problem of communication. Individual members of a design team can make decisions that could have a dramatic effect upon the "success" of the system although, in general, they have had no contact with the users. There have been many attempts to overcome this communication problem, but they all have failed because of two basic flaws. Firstly, the very nature of design teams tends to exclude the real users and, although various attempts have been made to include them in the design team, in practice the individual who can be spared by the organisation is unlikely to be the best person to represent the real needs and concerns of the actual users. Secondly, in large, complex systems the detailed knowledge and overview that is necessary to understand such a system is likely to be distributed across dozens of people.

3.2 Organisational Complexity
The handling of organisational complexity is even less well understood and handled. The notion of a software development life-cycle, with requirements capture being completed before the design stage, is unrealistic and dangerous. In reality, requirements are always changing, particularly in response to the realisation of the capabilities of the technology. Frequently the requirements as stated are informal, ambiguous and inconsistent, but the process of formalising requirements during the design stage should lead to any ambiguities being identified. Requirements capture and design are in fact symbiotic, the design process needing an initial set of requirements to drive it, and the design process involving the refinement of these concerns into a systematic and coherent statement of requirements.
Equally important is the fact that there will be many users in large systems each with concerns that will often be conflicting and will affect different stages of system development. For a method to be effective, it must provide a framework
for the identification of these conflicts and allow the consequences of any
decision made to be evaluated by the users concerned.

4. The ORDIT Methodology

4.1 Basic Principles of the Methodology
The ORDIT project has developed a framework within which design
requirements may be explored with designers and potential system users, and
has also focussed on the development of techniques that assist designer/user
communication. User participation in design is endorsed as being the most
effective way of developing complex products, and it was considered essential
that the methods developed would identify conflicts in interest between different
end users of systems, and would assist in the communication of these viewpoints
so that suitable design compromises could be reached. The ORDIT
methodology rests on the underlying philosophy that design is essentially a
conflict resolution exercise. For any given problem there is likely to be a number
of appropriate solutions, and therefore one of the roles of ORDIT is to assist in
the communication of these solutions, and to help those involved in design to
select those designs which are the best compromise between the conflicting
requirements.
The ORDIT project also emphasises the need to create a design framework that
can easily be integrated with conventional design methods, as the success of the
methodology and notation in the design market place will depend on the ease
with which it can be adopted by analysts trained in more conventional design.
For the same reasons the project emphasises the production of a practical
compendium of techniques, that includes some software support as well as
paper and pencil based tools.

An overview of the ORDIT methodology as it is currently envisaged is shown in
figure 1. Design begins with a group of activities, labelled Scoping, which
identify the problem boundary. Output consists of an analysis of relevant
stakeholders for future interviewing and summaries of the current
understanding of the problem, including a provisional list of problems with the
current system (if appropriate), requirements on any future designs, and a list of
design conflicts that may need to be resolved. At the end of this activity the
consultants should have performed preliminary interviews with all the relevant
stakeholders and summarised these for later use in modelling or requirements
capture.
A more detailed requirements capture process takes place in the group of activities labelled Requirements Capture. This involves the elicitation and classification of the nebulous sets of requirements that different stakeholders may have of a future system, along with any conflicting requirements that may exist. A parallel activity is a formal modelling process using the ORDIT notation. This process, which takes as its inputs the Scoping interviews data, entails first understanding and representing the current socio-technical system within the problem organisation, and then producing an abstract representation or model for use as a template for generating design options in the Solution Options phase of design. The design options produced can then be evaluated against any requirements generated and conflicts to be resolved. It should be noted that these phases in design are iterative in nature rather than sequential, with new requirements possibly being generated on the basis of design solutions being produced. Design options need to be tested and validated against the requirements placed on such systems with a view to an agreed design being selected for implementation by software engineers. It is anticipated that ORDIT could thus act as a front end to conventional design methods.

4.2 Advantages of the ORDIT Methodology
The ORDIT methodology is substantially different from most other design methods in that it provides a framework whereby both technical and organisational complexity can be addressed in an integrated fashion. It ensures that the rationale for all design decisions is recorded and made explicit, and that the relevant information is visible and open to inspection. It tackles organisational complexity by advocating a process which involves the policy-makers throughout the design of the system. In addition, one of the central recommendations of the ORDIT methodology is the development and maintenance of an abstract conceptual model that provides the framework in which requirements or design decisions can be understood and evaluated. The ORDIT methodology is truly user-centred, in that at each stage of system development the design must be validated by reference to the relevant set of policy-makers. The conceptual model ensures that the dialogue between the requirements capture and design activities is meaningful, by providing a common and consistent framework in which requirements can be explained to the designer, and the consequence of design decisions can be demonstrated to the policy-maker.

The ORDIT methodology is a technology independent, well-defined procedure involving the maintenance of a complex set of documents. It has evolved from considerations of what constitutes good engineering practice, and is open and
flexible enough to allow new languages and approaches to be incorporated within this basic framework. One of the central and most difficult aspects of the ORDIT methodology is the production of conceptual models to bridge the gap between the possible world of the policy-makers and the actual world of the designers. The conceptual models have to be adequate to accommodate all the views and distinctions made by the policy makers but also suitable as input to the formalisation process. It is our belief that the ORDIT modelling notation meets both of the above criteria.

4.3 The ORDIT Modelling Notation

The modelling notation is one of the key aspects that makes ORDIT different from more conventional approaches to design. Conventional systems analysis has focussed on describing the information flows and processes required to support any system under investigation, and, by not explicitly considering human organisation, has largely represented the human operator as an isolated individual. The formal relationships that link different members of organisations, and that define the obligations and responsibilities existing when people and technology are organised to perform some collective task have largely been ignored. Thus it is common practice in systems analysis to represent the agent that performs a particular activity, but to ignore structural issues regarding that activity, such as who is responsible for planning or monitoring, or who is ultimately held accountable for that activity to be performed correctly. This is a major shortcoming, as such obligations define in part the requirements of information systems used to support such socio-technical systems. One of the main consequences of working within a conceptual framework that admits only activity and information is that policies about the allocation of responsibility and obligation must remain implicit.

In order to consider such issues in a more rigorous fashion a notation has been developed within the ORDIT project which, as well as allowing function to be represented, also allows a consideration of the supporting structure which is an integral part of an organisation and essential to its operation. The notation is constructed using three basic building blocks representing the functions or activities needed to support an organisation’s objectives, the agents that perform such activities, and the resources that are utilised when such activities take place.

The key elements of the notation and the relationships between them are shown in figure 2. This diagram illustrates the distinction being made between representing the relationships between agents and activities, and the
relationships that agents have to each other. The relationships that agents have to activities will vary depending on their work responsibilities. Thus one agent may be responsible for executing an activity, whilst another might plan how that activity should take place. These are termed functional relationships, and when a number of agents have functional relationships to a given activity then it is implicit that the agents concerned also have some structural relationships to each other.

![Diagram](image)

**Figure 2.** Elements in the ORDIT notation

For example, if one agent executes an activity and another directs, then these two agents are said to have different functional relationships to that activity. This also implies that the two agents have some structural relationship to each other. As one directs the activity whilst the other executes, the former exercises some power over the latter who is therefore a subordinate with regards to that particular activity. This structural relationship will be mediated by an information resource (see figure 3), which may be formalised, eg written procedures, or may be informal, eg face to face. A specific example of this relationship, drawn from a medical case study, describes the situation where a doctor (Agent A) directs a nurse (Agent B) to treat (Activity) a patient. The information resource used is the treatment instructions written on the patient's record card, and there is a clear supervisor/subordinate relationship between the two parties concerned.
Structural relationships between agents support both the activity system and the agents' functional relationships to that activity system. The framework has been constructed to maintain a high degree of internal consistency, so that it should be possible to identify structural relationships from the distribution of functional relationships, and vice versa. The framework not only has descriptive power, but also has a prescriptive component, in that process and structure are implied from each other. For example if two agents have some relationship to a common activity, then a structural relationship is implied between them, and it should be possible to identify if an organisation is lacking structure to support process, or conversely lacking process to support structure. A lack of structural relationships, which have been logically inferred from functional relationships, will indicate that the particular organisation may have a problem in supporting these activities. In addition a structural relationship has to be mediated by some form of communication, and if it is revealed that this is non-existent then the relationship cannot be supported adequately, and the socio-technical system is likely to be ineffective.

5. The Problem of Variance

5.1 Current Challenges for the Notation Development
The emphasis that ORDIT places on structure, in contrast to conventional analysis techniques, means that the ORDIT notation provides a powerful tool
with which to describe socio-technical systems. However the notation is under
development, and many representation issues still have to be addressed.
Perhaps the most significant issue is how to represent variance adequately, both
in terms of activities and structures. Variance creates significant difficulties for
any representation scheme, and ORDIT is no exception to this.

5.2 Variance in Activities
Current approaches to IT design address the issues of selection, sequence and
contingency regarding activities, but emphasis usually focusses on describing
activities and information flows that pass through information technology rather
than through the whole socio-technical system consisting of both man and
machine. It is also rare for IT designer to look explicitly at the allocation of
function between man and machine, at the frequency with which specific human
operators need to perform tasks, and at the degree of variation in the components
and/or sequencing of the tasks themselves. For example, during a medical case
study, it was found that in an accident and emergency unit the activity of
recording patient details normally took place before diagnosis and treatment,
but this was not a hard and fast rule, and in an emergency this activity would be
delayed until treatment had taken place.

Another source of variance often ignored is the degree to which human beings
have to time share between different tasks. In some clerical tasks for example
work may be interrupted by higher priority tasks, so that some tasks are
incomplete when others are started. This has implications for the design of IT
systems, because if interruption is not allowed by the IT system an inefficient
usage pattern results.

In order to consider such issues adequately the ORDIT methodology and
notation will eventually need to integrate human activity task analysis
techniques into its wider framework, and if necessary extend these to cope with
variance in more detail. Techniques that will be explored in more detail for
possible inclusion into the ORDIT framework include Task Allocation Charts,
Hierarchical Task Analysis, and Personal Task Representation.

5.3 Variance in Organisational Structure
Variance in the activities performed by users of a given IT system creates some
problems for an adequate description of socio-technical systems, but these are
compounded by another problem, namely that of dealing with variance in the
organisational structure established to support those tasks. In many organisations a significant degree of flexibility is deliberately maintained in the work roles of particular individuals, and often such flexibility is related to work load or other environmental factors. As an example, clerical staff in an accident and emergency unit normally record patient details, but outside normal office hours this activity may be undertaken by a nurse. Variance in this context refers to the flexibility with which an organisation can vary the way that agents are allocated responsibilities for activities, and the designers of IT systems need to be aware that such variance can and does take place. This is particularly significant in the development of systems requiring a high level of security, where access to information may need to be restricted to those parties who are judged as 'needing to know'. Some degree of variance can currently be dealt with using the existing ORDIT notation, since organisational variance was seen as an issue to be addressed by ORDIT.

The ORDIT representation deals with organisational variance by distinguishing between agents and their roles. Thus more than one agent may have the same role, making it possible to model variance in who may take on a particular set of responsibilities. However, the framework cannot yet show explicitly the context in which this role variance can occur. For example the current ORDIT representation would show that a nurse and a clerk could both record patient details, but it would require annotation of the diagrams to record that these roles occurred at different times of the day. The extent to which such variance can be represented adequately is as yet unclear, but it appears difficult to represent in any dynamic way. When an organisation adopts a series of fundamentally different organisational structures in certain cases, eg emergency procedures, different ORDIT diagrams may have to be constructed to show each case.

Another aspect of organisational variance is that of change to organisational structure with time. This necessitates the development of IT products which will allow that change to take place without restricting the development of the organisation. A common problem in the development of IT systems is that they support the current organisational structure, but fail to allow the organisation sufficient flexibility to change its structure. The solution to this problem is not easy, but it has been addressed by ORDIT by using the concept of an invariant system template, consisting of activities and structure determined by policy, when creating design options. For example, essential activities in an accident and emergency unit will always be the diagnosis and treatment of patients, and there must always be agents with certain responsibilities for these activities. It may be
a matter of policy that only a doctor may diagnose a patient, and that the agent who treats a patient must have their work monitored by someone else. By identifying the components of a socio-technical system that are unlikely to change, design solutions can be produced that support this essential structure, but allow flexibility in those areas where change is possible. This is clearly an oversimplification of what remains a complex problem, but such variance can be dealt with to some considerable extent within the ORDIT framework as it is currently envisaged, and it is anticipated that such issues will be resolved in more detail as the project continues.

6. What the ORDIT Framework has to Offer Design

The ORDIT methodology and tools provide a framework that assists the designer in creating IT systems for complex and changing environments. The philosophy behind ORDIT is to provide designers with appropriate tools and methods, so that the human aspects of IT systems can be taken into account along with technical considerations. The power of the notation is that it integrates the elements of agency, activity and information, and shows how they fit together in a particular socio-technical system. It is argued that all three of these building blocks and their inter-dependencies are needed for an adequate description of a socio-technical system, and that some of these elements are missing from current approaches to design.

Returning to figure 2, it is our contention that current approaches to design emphasise information resources and their relationships to activities, and that agents are rarely included in any explicit representation. This means that structural relationships are largely ignored and little is said about functional relationships and access rights to information. This is a major shortcoming because, as already stated, the market for IT products is changing, and designers are being called upon to design products that will support the operation of whole enterprises, rather than the work of isolated individuals. The move towards creating products that will adequately support cooperative work places new demands on the skills of designers, and it is our hope that the framework and notation developed as part of the ORDIT project will provide some assistance in tackling these issues.

An important issue in the development of methodologies and notations to support the design of complex systems is how to deal with the dynamics of change. It is misleading to think of organisations operating in static
environments, as in nearly all cases some change occurs over time, and even in relatively stable operating environments, the need for flexibility in the way that organisational structure supports function can be considerable. The challenge for IT designers is to create systems that will support variability, both in the tasks that the organisation has to perform, and in the structures that are adopted to support such tasks. Whilst ORDIT cannot claim to have resolved these difficulties completely, it is argued that it is a significant step to have created a notation in which process and supporting structure can be adequately described as part of one framework. As we have indicated, the ORDIT framework does allow for some consideration of what we have described as organisational variance, and we will be investigating in more detail how task variance can be more adequately considered in the future.

References