9.1 Preamble.

In this thesis some of the important formal and design problems relevant to the BDMS with an ODS approach has been dealt with. One of the major objectives of this work is to develop a formal semantics of a bibliographical database management model. The formal semantics developed here is base on the concept of NODAL (New Object Oriented Database Language) and on the concept of accumulated arrows (AAs).

Accumulated arrow is used in the BDM model for modeling all the facets of an object, including its attributes, methods etc. Even when the rules are used to describe object behaviour, the accumulated arrow approach to formal semantic description has been found to be adequate and elegant. AA and NODAL has been found useful due to the following reasons.

1. All the important concepts of OO paradigm, e.g. aggregation, grouping, methods can be captured by one single concept.
2. Methods may be used to define predicates, which help in associative operations in queries.
3. Predicate may be used inside a method for associative retrieval as well as for conditional expressions with NODAL.

The results derived here provide a framework to synthesize two different programming paradigms. At one extreme there is scope to use a purely database suite for library in-house operation with rules, whereas at the other extreme there is provision only to use imperative constructs for defining methods in an OO fashion (as a queries passing protocol) for this purpose. In the either case, it may be noted that the schema consisted of one or layers.
The synthesis could be achieved partly due to the AA, partly due to the encapsulation and abstraction of objects. The locally defined NODAL scopes of different variables are discussed in chapter 8.

In this thesis only the acquisition and WPAC modules are considered for study. In the proposed models the algebra of AA served as the model of semantics of ODS schema and the NODAL language. The calculus of AEC, forms and the basis of the language NODAL and the backbone of ODS are defined in chapter 4. AA gives the rules of construction of well-formed expressions involving variables. All variables are typed. The subclass hierarchy defined by ISA relationship is used to identify a subtype hierarchy. The rules of ODS serve an important purpose in integrating declarative knowledge with the object system, Integration is aided by interpretation of methods as predicates.

The language NODAL based on $\langle \text{hip} \rangle$ incorporates further constructs such as for loop and allows embedding the C-expressions within NODAL, expressions, for performing different types of computations.

The formal ODS model honors encapsulation and abstraction of objects. Interface is separated from the implementations (e.g. methods). A query may involve communication among objects through methods. A query expression may be constructed with the methods names and parameters. The ODS compiler statically binds the method to be invoked as a response to the methods. If the code of the method changes after this binding done by the ODS compiler, ODS automatically re-links the new code with the already compiled query. If, however, the method is deleted or its signature is changed, one has to explicitly recompile the original query.

A methodology to design an interface for BDM modules for different in house operation of a library is proposed in chapter 7. Stochastic criteria derives for optimal design of record structures. The three levels of abstraction of ODS store helps to combine the two different implementation approaches, namely the file approach and the complex object.
approach. This also helps in structuring BDM transactions into multiple levels thereby increasing the possibility of concurrent execution of transactions.

The proposed storage organization partially captures class constraints called equational constraints. Even when these constraints cannot be captured fully by the storage organization design, they play a key role in reducing the number of file access or evaluation of a query. This is a source of semantic optimization of ODS queries.

The theory of concurrency control with multi-level nested transactions, proposed by Beeri et al., is adopted for the design of lock-based concurrency control scheme of multi-user ODS. The tree levels of ODS storage abstraction identify the levelization of ODS transaction (WebDb.reqbk) (scheme: relation). Scheme for concurrency control and recovery is designed based on this theory. Proper visibility of different sub-transactions of the transaction is ensured and a hypermedia engine is tested based on the procedural language NODAL for Web Enabled Application (WEA).

9.2 Direction of Futures Research.

The formal model developed in the present work essentially characterizes the fundamental aspects of ODS. The techniques used may be extended further for supporting other requirements of library in-house operation, without violating the fundamental principles of OO. The systematic treatment of OO which explains the principles on algebra should go further in developing methodologies for analysis of OO system.

NODAL does not support easy to use constructs to implement aggregate operators. One has to implement these constructs in an indirect way. It is worthwhile to enhance the language based on the concept of AA to incorporate such aggregate operators with the language.
In the present work, the dynamic aspect of OO schema, which is also called as schema evolution has not been considered to a great extent. The proposed model only allows for monotonic increase in the ISA relation. One may add a new class to the ISA hierarchy. However, it cannot say anything about the potential effects of a change of an ODS schema on its semantics. The model does not provide the semantics of various possible schema evolution operators, such as addition/deletion of parts and sets, modification/addition/deletion of methods/constraints etc. Naturally every possible schema evolution operator should not be allowed, for the sake of well-definedness of semantics of evolution. It is necessary to properly characterize a minimal complete set of operators whose semantics can be safely provided. Semantics of an evolution operator must provide a methodology for consistently propagation the effects of the changes to different part of the schema, as well as to the existing object instances. This issue needs further treatment.

In the present work semantic optimization of queries is implemented based on equational constraints. There are other sources of optimization, like algebraic optimization. Algebraic properties of different constructs of the algebra (e.g. commutativity of parallel join, a restricted form distributivity of the two joins etc) of an AA may be used to optimize certain expressions for other operational aspect of the library. A/OODBMT (Active Object Oriented Technique) which integrates and extends the Object Modeling Technique (OM). Database application may be adopt by implementing time critical reaction through integrating even-condition-action (ECA) rules. This aspect, however, requires further research.