D.C. Tsichritzis and F.H. Lochovsky, "Hierarchical Data-Base Management: A Survey", Computing Surveys 8, 1 (March 1976), 105-123.

This survey provides a useful introduction to some of the facilities of IMS (which I know something about); it appears to be an equally useful introduction to System 2000 (which I know nothing about). I do wish, however, that (1) the paper had been more successful in its attempt to provide a general motivation for hierarchical data structures, and (2) the paper had provided a more complete overview of all aspects of the systems. In particular, in spite of the central role accorded data description in Sibley's introduction (in the same issue of Computing Surveys), this survey makes no mention of the data definition facilities in these systems.

The paper is at its best when it describes hierarchical data structures and their manipulation, as tools which are useful in the management of data. There is a good tutorial overview of the basic manipulative language, the execution environment, and some implementation considerations. The comparison of two levels of manipulative language is important. Some people judge the merits of a data structure by the kind of manipulative language provided for it; these examples show that different kinds of languages can be implemented for the same structure.

The paper makes a needless attempt to establish that hierarchies are familiar and natural constructs normally encountered in information. There is an unfortunate play on words here; while we do naturally encounter a number of constructs we call "hierarchical", they rarely correspond to the kinds of structures supported by these systems. In its most widespread use, "hierarchical" refers to things which can be stratified, ranked, classified, etc. But in our use of the term, the restriction to one-to-many relationships is central; I was unable to find this condition mentioned in the definition of "hierarchy" in any of a half dozen dictionaries (both general and technical).

But even in the framework of one-to-many relationships, we are misled. The kinds we encounter very often in nature (and the ones cited as examples in this paper!) are homogeneous trees, which are <u>not</u> the kinds of structures supported by these systems. A homogeneous tree is comprised of relationships among the same kinds of things, as in a personnel organization chart, parse tree, game tree, etc. The structures supported by these systems may only represent relationships between different kinds of things, e.g., presidents and congresses.

Thus, to serve the tutorial function of providing a helpful perspective for the reader, it might have been better to indicate the differences rather than the similarities between the formal and intuitive concepts of hierarchy.

Incidentally, I was annoyed by the "quantification switch" by which the "universality" of hierarchical structures was established in the opening paragraphs. I fail to understand how we leap from the existence of some hierarchical structures to the premise "since the world appears to us to be arranged hierarchically...."

The excursion into abstract information concepts will also have an adverse effect on lay readers. It is confusing; but the poor reader is more likely to blame his lack of comprehension on his own incompetence. Several terms are introduced (e.g., "attribute relationship" and "entity relationship") without any real definition, in the hope that the ensuing examples will indicate the meaning. But I have absolutely no idea of what principle to infer from an example in which a political party is an attribute of a president, whereas an election is a distinct entity to which the president is related. ("Both the president and the election exist independently of one another..." Should we infer the contrary for his party?) And, similarly, what is the motivation for structuring the relationship between an administration and a president differently from the relationship between an administration and a vice president? (For another exercise, we could speculate on how the information should be modelled to reflect the fact that a vice president and a president might be the same person -- in different administrations, of course.)

I fear we have here a classical inversion of cause and effect, an attempt to rationalize a theory to justify a pragmatic result. The theorists would like to believe that such data is aggregated into a single record <u>because</u> these are attribute relationships. However, I suspect if we probed the meaning of "attribute relationship" we would undoubtedly discover that the term is applied to certain information <u>because</u> there are economic motivations to aggregate the data into a single record.

Without a clearer grasp of the concepts intended by these and similar terms, I had difficulty understanding the concluding remarks comparing hierarchical with relational and network data systems. (But the summary of advantages and disadvantages of hierarchical systems is good.)

The transition from a "general relationship graph" (Figure 1) to a "hierarchical definition tree" (Figure 3) is likely to be misinterpreted as a generalizable process for casting information into hierarchical structures. It should have been made clearer that the process only applies when (1) there are no relationships involving just one entity type (e.g., president succeeds president), and (2) there isn't more than one relationship between a given pair of entities.

There should have been some discussion of data definition constructs and facilities. A tutorial is needed here even more urgently than for the manipulative facilities, at least for IMS. Perhaps it was omitted because it then becomes

difficult to maintain the posture that IMS is a purely hierarchical data system. With the inclusion of logical relationships and secondary indexes, IMS supports a complex data structure which in aggregate begins to resemble the networks of the CODASYL system. The status of being a hierarchical system is preserved only in the sense that applications are limited to dealing with hierarchical subsets of that data complex. (And even that position is arguable, since a given application may see several such views simultaneously, with overlap permitted. The application could thus see several different parents for a given segment, one in each view.)

W. Kent
IBM General Products Division
Palo Alto, California

A. S. Michaels, B. Mittman and C. R. Carlson. "A Comparison of Relational and CODASYL Approaches to Data-Base Management", Computing Surveys 8,1 March 1976, 125-151.

It is a pleasure to read this well written and finely structured paper. Nowhere else can be found as lucid, concise and unbiased a comparison of this complex subject.

Disappointment will most assuredly set in to any reader who expects to read this paper and be told which approach is "best". Although in a number of places the authors cite the strengths and weaknesses of each approach, more often than not a simple, straight forward comparison is presented.

The crux of the problem is determining what is "best" lies in the fact that the user community is so varied, each with its own requirements set, that each approach works "best" against different elements. When the user community is viewed as monolithic, both approaches fall far short of satisfaction. The CODASYL DBTG approach directs itself at the programmer/DBA while the relational approach is directed at the end user or the structurally independent and parametric user (as classified by Senko).

Which system is better depends entirely on the way in which one views the world of computing in the future. If, as in Bachman's terms, the