The Case for Software as Documentation

by David Bearman[1]
Archives & Museum Informatices
5600 Northumberland Street
Pittsburg, PA 15217
(412) 421-4638

Abstract

A study of the potential value of software as historical evidence, conducted by the author in 1987 [published as "Collecting Software: A New Challenge for Archives & Museums, Archival Informatics Technical Report, vol. 1 #2, Summer 1987], raised a number of critical issues for social science data archives. First, a number of social control functions that would traditionally have been the subject of social scientific analysis that are today embodied in software. Second, the meaning of much unobstrusively gathered data about our society is revealed only through analysis of the software systems that contributed to its collection. And thirdly, the potential information value of social science data is captive of the technologies used to analyse it, and it can only be understood as an historical agent within the actual context of its use; e.g., as software of its day permitted its utilization. For these reasons it is suggested that software must itself become a focus of collecting activity in social science archives and the implications of the requirement are explored.

[1]Presented at the International Association for Social Science Information Service and Technology (IASSIST) Conference held in Washington, D.C., U.S.A. on May 26–29, 1988

Introduction

Last year I was contracted by the Computer Museum in Boston to examine the viability of collecting software as a museum objective. Previously, I had given little thought to software *per se*, and like most archivists had considered data to be the natural holding of a machine–readable archive. Since then, however, I have come to regard social science data archives artifactual because they have failed to collect software and its documentation. I have been forced to recognize, that in important ways the "facts" we possess about our contemporay world are created by software and the assumptions built into instruction sets, and that social scientists cannot, therefore, be taken seriously until they fully master the systems that generate their data. It is increasingly clear that much of what is esstinaial in the study of modern economics, political science, and sociology is the product of software that we cannot afford to know only from its own account of itself.

The basis for my persepctive is documented in a recent issue of Archival Informatics Technical

Report.[2] One finding of that study will come as no surprise to data archivists; they are not alone in failing to collect software as documentary evidence (a few libraries are collecting it as a user requested item, but no archives collect software as documentation). Secondly, like most other archives and museum managers, data archivists find the concept of collecting software daunting because they usually assume that it must be retained in its original form and be "run" on the machine for which it was written. Neither assumption turns out to be valid, as very little can be learned about software code by executing it. Thirdly, we haven't got very good methods of classifying software, and the commercialization of some standard software functions has not led to a reduction in the number of applications developed. Finally, the history of software is already sufficiently obscure that it would be impossible to find the actual algorithms (as opposed to the formulas developed by the economists in charge) used in the calculation of the Commerce Department's monthly cost of living index, an index which affects virtually every aspect of American economic life, or the calculation used to determine welfare benefits in any of the fifty states in 1976.

Several recent contracts in which I have been involved have extended the findings I reported in the summer of 1987. In documenting the events following the takeover of the U.S. Embassy in Tehran in 1983, the importance of software, in the form of scenarios in guiding the rescue mission, has become clear.[3] In documenting United Nations policy making, the role of software in decision support systems and mail networks has begun to concern records managers.[4] In documenting business strategies of the largest corporation in the United States, the role of software embodied in electronic switching circuits has been one focus of a monopolistic practices suit.[5]

Early in 1987, Alan Kowlowitz, a records analyst with the New York State Archives, was assigned to appraise the records of the New York State Division of Criminal Justice Services including two integrated, online, information systems: the Computerized Criminal History System (CCH) and the Offender Based Transaction Statistics System (OBTS).[6] Like some other observers,[7] Kowlowitz reported that the changing nature of documentation practices in organizations employing electronic information systems is posing more fundamental appraisal problems for archivists than the appearance of traditional records in electronic media formats up until the past few years ever did. The guidance

---

[2]Bearman, David; "Collecting Software: A New Challenge for Archives and Museums", Archival Informatics Technical Report, vol. 1, #2, Summer 1987.

[3]The National Security Archive, a non-partisan, non-profit organization devoted to documenting contemporary U.S. foreign and security policy, is compiling the record of these events. The assumption built into software used in desert rescue "games" is a critical element of the record.

[4]The United Nations Administrative Coordination Committee Information Systems Group (ACCIS), has hired me to develop policy guidelines for management of electronic records as a reflection of this concern.

[5]AT&T is not, of course, the only firm charged with using software in this way - American Airlines admitted its Sabre reservation system was biased in its favor and agreed to desist. Any reader of business journals knows that many firms are using software as an conscious element of business tactics.

[6]Kowlowitz, Alan; "Hands up, you're under arrest: Appraising Criminal History Data in the Age of the Electronic Case File", paper prepared for the SAA meeting, 6 September 1987; "Appraisal Study of the Computerized Criminal History System (CCH) and the Offender Based Transaction Statistics Systems (OBTS) of the Division of Criminal Justice History", New York State Archives, Albany, NY, 1987.

[7]Aronsson, Patricia and Brown, Tom; "Government Archivists and Government Automation: The Odd Couple", Government Publications Review, 13, 1986, p.561-570.

developed for procedural, centralized, information management systems designed to serve a single organization is found wanting for non-procedural, decentralized, inter-organizational databases. Kowlowitz' study of New York's CCH & OBTS illustrates the nature of these challenges.

The New York CCH was created, along with similar systems in other states, by grants from the Law Enforcement Assistance Administration starting in 1966 to improve the inter-jurisdictional flow of information in support of more effective administration of criminal justice. Offender-based Transaction Systems, OBTS's, were introduced in 1972. Both represented a significant departure from agency based and function based information systems, but one that is increasingly common in other domains as well as criminal justice. Both are also part of a larger conceptual (although not fully implemented) network emanating from the FBI's National Crime Information Center (NCIC) which maintains an "Inter-state Identification Index" running on its National Law Enforcement Teletype System Network. Each is fed by agencies and events at a local level - local police forces, district attorneys, court hearings, parole boards and the like.

In this system, one agency (local police) records information about an arrest, and others (DA's, Courts, Corrections, Parole) add information about the offense and the offender, disposition of the case etc. Various agencies are entitled to see and use different information in the online system in conjunction with their daily work. All CCH/OBTS data is considered confidential and access is restricted, but some data is sealed either by court order or laws pertaining to juvenile offenders. From time to time data may be purged from the system either in order to make room in the computer or to comply with a court order. The DBMS itself is quite complex, but it contains little unique information and is not as complete in its parts as information resident in the contributing agencies. The database contains few clues as to its use; query audit trails are not provided and data entry audit trails disappear with transactions. The system is linked dynamically with PROBAMIS (the Probation Management Information System) and PARMIS (the Parole Management Information System), but not all information is passed along in a timely fashion and some stages in many cases are missing. Doubtless some cases of mistaken identify are recorded and some national searches in the NCIC/FBI system result in false drops. Data may also be sealed by court order (indeed 25% of all cases are) and be unavilable when a subsequent record is made, leading to data redundancy and duplicate (or near duplicate) records. Needless to say, the database is not entirely clean and not completely trustworthy.

Data As Artifact: Comments

My reasons for introducing the Kowlowitz study are to discuss the data wehave been accustomed to retaining from such databases and what information we as social scientists should be studying about such systems.

The first point we can safely make about such a database is that if the data from it can be saved in a machine readable form, respecting privacy, it could be a valuable source for a broad range of sociological, political, economic and other studies. And it will doubtless be used.

The next, equally obvious point is that data obtained from such active governmental records series is problematic because its collection is not controlled well. Already we are aware from Kowlowitz that some police agencies are too small to report in a timely fashion and some do not report at all. Some courts are more regular

than others and the actions of the Corrections Department occasionally go unreported. We may hope that the causes of bias in the data are non-systematic, but are they? For reasons of scholarly integrity, we will want to determine establish any sources of artifact in this database before using it, how would we begin?

If these were manual files we would ask about the training of the clerks, the procedures they used to determine whether or not a record of arrest was for someone already represented in the database, what information was available to the arresting officer (DA, judge, parole officer, etc.) and how it was used, and how often each office made what additions/corrections to the records. To ask these questions in the electronic environment is to ask about how the software operating the system was programmed.

Let us consider a more complicated social use of the data, in which the focus is on the individual case. A victim sues the state because his attacker was prematurely released from prison due to inaccurate data in the system. The case comes to court and the data in the system is correct. We must establish whether the data was correct when the parole hearing took place and why the parole board did not consider the damaging evidence. Are they negligent, or was the system incorrectly designed? We cannot know without studying the software. Only the programming code will tell us whether the system withheld restricted data from the parole board, failed to find the data due to error on the part of the parole board or on the part of the system designers, or didn't contain information it now includes, and if so how it can be established when it was added. Data from such systems is as mute about what it means, and what is missing, as data from an experiment without protocols.

Software and Organizations

Software is being developed by organizations to respond to the conditions in which they find themselves. Whether it is designed to calculate buying or selling points for stocks, or best routes for air travel, it embodies the assumptions of an organization and contrains the way they subsequently do business. These assumptions and contraints may backfire, as the collapse of some Wall Street brokerage firms in October 1987 appears to demonstrate (and in this I am following the Rogers?? Report). On the other hand, American Airlines appears to have demonstrated in two decades of tweaking its Sabre reservation systems, it can also make for success. But how will we understand these organizations, and their strategies, in the future without access to software documenting just how they operated? Will a future historian be able to write a contemporary equivalent to Alfred Chandler's classic Strategy and Structure[8] without software to study?

As more and more of the "business rules" that guide organizations become embodied in software, we are less able to understand an organization without understanding its software. In the electronic mail system, who has access to the President's mail box? If the comptroller did not see the negative cash flow trend, could he have with a single command on his PC, or was part of the problem with the firm the way in which case flow had been obscured in setting up accounts?

In the private sphere, we do not have the clout to insist that software be retained, except in some heavily regulated arena's, but for public agencies, that affect the lives of numerous individuals and groups in administration of

[8]Chandler, Alfred D. Jr.; Strategy & Structure: Chapters in the History of the American Industrial Enterprise, Cambridge, MIT Press, 1962.

every regulation, and do so with criteria built into software, it is an issue of accountability. When the immigrant, a Pakistani child-bride of a permanent resident, was denied temporary resident's status, was the denial (made by computer processing of facts in her application and his record) correctly interpreting the intent of Congress?

Indeed, social scientists will increasingly find, if they have not already discovered it, that software available to them, and to organizations in our contemporary society, enables them, or restricts them. How we interact with organizations, no less than how we interact with data, is becoming a product of software interfaces and functions.

Collecting Software

It's one thing to realize that software is itself a source of evidence about the functioning of society and a crucial key to understanding the execution of organized activity in our age, and quite another to do something about documenting it. Our first problem is that we are very uncertain about what we would need to keep in order to adequately document software.

Should we retain a running version of system, i.e. the panoply of licensed software (object code) that operated in an application arena? This is impossible as anyone who has been responsible for systems software in a large organization is painfully aware. Even documenting what version of an application system was running with what releases of operating system, telecommunications monitors, report writers, and hardware configurations is exceptionally difficult, and, in practice, such "configuration management" of computing facilities is found more in exhortation than in practice. Actually maintaining running software systems as they were over their active life would, of course, also involve maintaining hardware, which is at least prohibitively expensive, if not impossible.

If we don't retain a usable software system, what are the best sources of evidence about how it ran? Interestingly, the answer is not as simple as keeping the source code, although much can be learned from source code (assuming it is written in a language that can be understood by the researcher). It is extremely difficult to reconstruct what it feels like to use a system from its source code, so external functional specifications and tutorials, user documentation and even films of the system in use, are important documentation. Finally, assuming we want to understand not just what the software was, but how it came to be that way, we will want to retain design specifications and early drafts of important routines and algorithms. In other words, collecting software as documentation is really no different than collecting documentation of any other activity of an organization, and <u>does not</u>, ultimately, involve actually collecting the stuff normally thought of as software, e.g., object code.

Conclusions

Responsible social science research on large data collections requires that we understand the software in which such aggregations of information came to be collected. Understanding the way in which people interact with their society in the late twentieth century also requires that we understand the nature of the systems, run by programs, that they are interacting with. Finally, we can only understand the ways in which organizations, and even some individuals, succeed or fail in our society if we appreciate the fact that their

successes and failures are mediated by computer programming decisions, embodied in code, that reflect their expectations of the way in which the external world will, or should, behave and how they will respond to it. For all these reasons, we must prepare ourselves to collect software as documentation.

Do we know what to do with this documentation? At present, no. We'll need to learn how to read these new sources of evidence however if we are going to make sense of the political, economic, social and even cultural worlds in which we live.□