Foundation software: A significantly improved approach to the development of large application systems

by GARY A. CURTIS
American Management Systems
Chicago, Illinois

ABSTRACT

The American Management Systems (AMS) approach to the technical framework of large applications systems is based on a concept we call foundation software, an integrated environment of standard packages and custom modules that provides common services to the development and operation of applications software. This environment provides a standardized, structured, and simplified view of the outside world to applications software. Use of foundation software dramatically improves the economics of development, operation, and maintenance of large systems and reduces the risk of developing such systems. In this paper, the foundation software approach is defined and illustrated, with particular emphasis on the relationship of foundation software to the overall architecture of large-scale systems and the impact of foundation software on the application-system development life cycle.
INTRODUCTION

The AMS approach to the technical framework of large applications systems is based on foundation software. Use of foundation software dramatically improves the economics of development, operation, and maintenance of large systems and reduces the risk of developing such systems. This paper describes the foundation software concept and our experience using this approach from three perspectives:

- The relationship of foundation software to the overall architecture of large-scale systems is discussed, including foundation software functions and components and the major benefits to developing large-scale systems using the foundation software approach.
- The use of foundation software is an inherent part of the AMS GUIDE: Methodology, systems-development life cycle. Foundation software activities in each phase of the systems-development life cycle are described.
- The integration of foundation software and application software is discussed from the viewpoint of the overall system architecture.

THE DEFINITION OF FOUNDATION SOFTWARE

Foundation software is an integrated environment of standard packages and custom modules that provides common services to the development and operation of applications software. This environment provides a standardized, structured, and simplified view of the outside world to applications software.

Foundation software increases productivity throughout the development, operation, and maintenance of applications by isolating applications software from changes in the technical components of the computer system and by making those components easier to use.

Despite claims to the contrary, most operating systems, database management systems (DBMSs), teleprocessing monitors, and other technical components fail to simplify applications development and operations. Indeed, a common result is that such components serve to greatly complicate the development and operation of applications software. The foundation software approach avoids this complication and delivers the benefits offered by these technical components to application software in a simple, effective manner.

Software Architectural Levels

Large-scale systems can be divided into three major architectural levels. Figure 1 shows the relationships among these levels. Associated with each level are specific functional and organizational responsibilities. The levels are

1. Technical environment—This level includes the technical components that the system designer takes for granted and that usually cannot be modified for application purposes. It typically consists of
   - system control program (OS/VS, DOS/VSE, MPE, VMS, etc.)
   - access methods and utilities (VSAM, VTAM, ID-CAMS, etc.)
   - network architecture (SNA, DECNET, X.25, etc.)
   - telecommunications monitor (IMS DC, CICS, IDMS DC, etc.)
   - DBMS (IDMS, DL/I, Model 204, IMAGE, etc.)

Responsibility for the maintenance and support of components of the technical environment generally rests with the computing facility and its systems-software organization.
2. Application software—This level is the functional core of the application system. It contains all of the specific substantive functions that relate to the application (business) problem at hand:
   - editing, verification, cleansing of application data
   - computations, analysis, transformations of application data.
Most if not all, of the application-data-dependent processing operations occur at this level.

3. Foundation software—This level provides an interface between the application software and the detailed considerations required by each component of the technical environment. Foundation software directly uses standard programming, communication, and control services of the technical environment, such as database calls, network messages, and control blocks, to provide high-level common application services such as menu processing, security, and error handling. Foundation software may also include software packages such as inquiry software and report generators.

**Foundation-Software Functional Scope**

The functional scope of foundation software cannot be rigidly defined. For any specific application system, determining the functions to be provided by foundation software should be done in the context of the application design characteristics, constraints of the technical environment, and the organizational environment of the system-development effort.

Our experience is that functions with the following characteristics are nearly always more effectively handled as foundation software:

- common use of the function throughout some or all application subsystems
- simplification of complex technical-environment features
- interface to technical-environment features that are subject to a high rate of technological change
- expected volatility in application requirements for the function
- missing or poor technical-environment features
- performance sensitivity.

Although benefits from any one of these attributes can justify inclusion of a function in foundation software, it is usually the case that foundation software functions exhibit benefits due to several of them. Based on these attributes, a general statement of the functional scope of foundation software can be made. The functional areas that foundation software typically comprises are discussed later.

**Foundation-Software Components**

Foundation software consists of three types of components:

- Custom interface modules—Efficient, standardized utilization of the technical environment's most complex components is usually provided through custom interface modules. In some cases, such interfaces provide an entire environment for applications processing. Control relationships among these interfaces and application software vary from normal subprogram linkage to architectures in which application software processes under the control of a foundation software interface environment. Such facilities as a reference-data interface, on-line menu processor, and report distribution subsystem are usually provided in this manner.
- Packaged software—Software packages are usually integrated into foundation software with custom interfaces. This topic is considered in more depth later.
- Common modules—Processing functions required frequently throughout the application software are provided as common modules that are invoked through standard subprogram linkage. Some functions provided as common modules, such as numeric editing, free-form parsing, and data validation, are generic to most business applications systems. Others provide services that are specific to a particular system or subsystem.

**The Role of Packaged Software**

Software packages, such as report generators, inquiry packages, and data-entry packages, are usually important components of large applications systems. Packages can provide cost-effective solutions to many of the processing functions of large applications systems. A major difficulty in the effective use of packages in such systems is that packages tend to be functionally narrow in scope and are often cumbersome to use outside the context of standardized interfaces and procedures.

Through the use of foundation software front-end and back-end interfaces, software packages are integrated into the processing environment of large applications systems, such that the functions which the packages provide can be used much more effectively by applications designers and programmers than would be possible if they were used stand-alone.

For example, in a large-scale IMS DB/DC financial system, ad hoc inquiries into historical and other databases have been provided cost effectively by integrating the INQUIRY IV/IMS query package into the foundation software on-line user environment. Users access the package through the standard application on-line protocol and menus, request their queries in simplified, familiar terminology, and are essentially unaware that a package is being used. This level of integration allows application security controls to be applied to ad hoc queries and allows uncomplicated transfers among package and nonpackage transactions. In this case, functionality was also added to the package through foundation software. The foundation software interface that captures query requests for transfer to the package first scans the requests for search conditions that will result in unacceptably lengthy on-line database processing and redirects such requests to overnight handling.

For a package to be incorporated into foundation software it should be possible to use the package without any internal modifications. This restriction does not include the use of
Foundation Software: A Significantly Improved Approach

Vendor-supported exits through which the package passes control to other software. Indeed, the use of such exits is a common method for tailoring a package to specific application requirements within the foundation software.

**Foundation-Software Benefits**

The foundation software approach provides benefits during all phases of the system development process. Herein the major ones are described, by phase.

**During system design**

Foundation software results in a high level of modularization and is, in this regard, an extension of structured design methodology. Common functions are designed only once and many more functions can be provided by standard, reusable software.

The high degree of isolation from a need for detailed understanding of the technical environment enables application designers to concentrate more effectively on solving business problems.

**During system development and implementation**

Senior technical staff resources are scarce throughout the computing industry. Foundation software allows the efforts of such staff members to be concentrated in high-payoff areas. This allows technically sophisticated applications to be developed by relatively less sophisticated staff.

During the development of large-scale systems, changes in the technical environment are usually introduced independent of the application development effort. Foundation software isolates application programmers from these changes and results in fewer disruptions, less recoding and greater productivity.

**During system support**

Rapid change in technology presents the system support staff with a continuous major effort to keep application systems functioning against a moving background. With the foundation-software approach, application programs are isolated from the effects of this change. Application software can be maintained by less sophisticated technical staff.

Due to the extensive use of foundation-software common services, new application functions can be added with minimal impact on the existing system. The application can thus be adapted to changing user requirements more easily and more cost effectively.

**INTEGRATION WITH THE SYSTEM DEVELOPMENT PROCESS**

The foundation-software approach is an inherent part of the AMS GUIDE: Methodology, systems-development life cycle.

This section examines some of the key aspects of the foundation-software life cycle and the foundation-software development and support team.

**Foundation-Software Life Cycle**

Foundation-software is designed, developed, and implemented in a life cycle that is integrated with that of the overall application system. The relationship of the foundation-software and the application-development life cycles is shown in Figure 2. Some of the key aspects of the foundation-software life cycle follow.

- **Overall system architecture**—The first step in the foundation-software life cycle is the development of an integrating framework for the design of the system as a whole. The overall structure for the system, the system architecture, is designed as early as possible. The system architecture defines the user interface with the on-line components of the system, the environment in which the system will be developed, and the operational environment in which the application software will run, and provides a model for effective management use of the system. This step is critical to the successful evaluation of the total system through its design, development, and support phases.

<table>
<thead>
<tr>
<th>Application System Life Cycle</th>
<th>Foundation Software Life Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept Definition</td>
<td>Overall System Architecture</td>
</tr>
<tr>
<td>System Design</td>
<td>Foundation Software Design</td>
</tr>
<tr>
<td>System Development</td>
<td>Foundation Software Development</td>
</tr>
<tr>
<td>System Implementation</td>
<td>Foundation Software Implementation</td>
</tr>
<tr>
<td>System Support</td>
<td>Foundation Software Support</td>
</tr>
</tbody>
</table>

**Figure 2—Relationship of the foundation software and the application-system life cycles**

From the collection of the Computer History Museum (www.computerhistory.org)
functions nor an exhaustive list. They do, however, represent
The following foundation-software functions reflect AMS ex­
in contrast to most application teams, the foundation-software
The primary objective of the foundation-software team's ac­
tications and support activities begin during application sys­
organizational, and functional
permit experience gained during application develop­
to be fed back into the foundation software without

Organizational Impact

The system-development project usually has a foundation­
and support of all foundation-software components. The primary objective of the foundation-software team's ac­
press releases, and the use of previously developed foundation software, new custom founda­
Foundation-software design specifications must be com­
early in the system-design phase to support the development of program specifications for
Development of foundation­
technical environment and application
areas and are neither a set of required foundation-software
functions nor an exhaustive list. They do, however, represent

FOUNDATION-SOFTWARE FUNCTIONS
AND FEATURES

The following foundation-software functions reflect AMS ex­
various technical environments and application
areas and are neither a set of required foundation-software
functions nor an exhaustive list. They do, however, represent
areas where experience shows the foundation-software ap­
approach to be effective and the payoff to be significant.

On-Line User Management

On-line user management facilities provide a friendly,
screen-oriented environment that allows the user to exercise
all authorized application functions. The language, sequence
of operations, and features of this environment are relevant to
the application and the user, and are not constrained by jargon
and idiosyncrasies of the on-line technical environment. Features usually provided include

- menu processing
- screen handling
- security
- user assistance.

Input Management

Input management controls the processing of application
data from the point at which the data enter the system until
the data have been accepted as valid and has been stored in
application-data structures. Data can be entered, processed,
corrected, and reprocessed in batch or on-line modes, in any
combination. Standardized, application-oriented data organi­
sations such as transaction, document, and batch organi­
ations are used. Specific features include

- data entry
- error processing and suspense
- error correction
- document approval
- input workflow control.

Application Data Management

Foundation software is used to simplify and standardize ac­
access to application data structures and, where needed, to make
the access more efficient. This function generally takes the
form of data-structure (database) access interfaces that present
to the application tabular logical views of application data, and
of update-isolation facilities which ensure that application data
structures are updated in a consistent, synchronized manner.
The interfaces manipulate data structures using efficient, and
often complex, call patterns, database facilities, and custom­
developed functions. The major functions addressed by appli­
cation data management are

- reference data maintenance and control
- reference data interface
- update isolation
- application data backout.

Network Management

In distributed data-processing environments, application
software may operate on different processors connected
through a network. In such an environment, isolation from the technical complexity of the network is provided by network-control foundation software. Network-control software allows the application software to be designed and implemented without regard to where in the network architecture the application software must operate. Network control includes the functions of

- network status
- transaction routing
- distributed site support.

Output Management

Management of the varied forms and high volumes of output produced by large-scale application systems is controlled by output-management foundation software. Some of the major functions include

- report distribution
- graphics interface
- report generators
- on-line inquiry.

System Management

Overall management of the processing of a complex application system is simplified through several foundation-software functions. The objective is to present to system administrative personnel a standardized, simplified view of control facilities that makes it possible to exercise complex functions of the technical environment with minimal technical expertise. System-management foundation software provides:

- scheduling
- recovery/restart
- performance monitoring.

Office Automation Facilities

Foundation software integrates application-system data and reports with office-automation facilities in two ways. Where the user's office environment includes existing facilities (such as stand-alone word-processing systems), foundation software provides interfaces that allow application software to send data to and receive it from these facilities. Where office-automation features are required by an application system but are not available in the user environment, foundation software includes both the application interfaces and the actual document preparation and mailbox facilities themselves. The office-automation facilities supported by foundation software include:

- document preparation
- word processing
- electronic mail.

Technical-Environment Enhancements

Occasionally, technical environments do not provide some basic system support facilities that are essential to fulfillment of the application's primary objectives. In this situation, the system designer must often decide between a considerable sacrifice in application functionality and the development of significantly more complex application software owing to the incorporation of technical support features. When analysis of this tradeoff leads to a decision to support the application's required functions by developing the complex facilities missing from the technical environment, the foundation software approach minimizes the adverse impact of this additional complexity. It further ensures that the complexity of the application software is not affected. The following are examples of facilities that are normally, and preferably, provided by the technical environment, but that may be provided by foundation software when necessary.

- database locking
- transaction logging
- job control
- dataset management.

FOUNDATION SOFTWARE CASE STUDY

A description of a large-scale integrated financial system implemented under IBM's IMS DB/DC technical environment is presented below. The relationship of the foundation software and application software components are particularly noteworthy. Figure 3 shows the overall system architecture keyed for the following discussion.

The on-line user interface (1) handles user sign-on, sign-off, and security checking. It presents users with a series of menus to get to the desired system function, be it data entry, processing, or an inquiry request. As a security precaution, if a terminal has not been used for an extended period (set by the system administrator—perhaps 15 minutes), then the On-Line User Interface will automatically sign-off the terminal.

A data entry/error correction program (2) accepts input transactions (i.e., documents) and stores them on the document suspense database.

If the user wishes to process the transaction immediately, the data entry/error correction program will perform an IMS message switch to an application edit/update program. If errors are detected, the application program will signal the data entry/error-correction program (2), which will post the errors highlighted back to the user, who may then correct the erroneous data and immediately resubmit the document.

Note that data can be entered and corrected without ever interacting with an application program. The data entry/error correction program also handles the scheduling of documents for processing. The purge-accepted-documents program (3) physically deletes documents from the document suspense database and creates an audit trail log.

Reference tables are created and maintained by the foundation software reference data edit/update software (4) and accessed through the reference data interface (RDI) software (5). The RDI is a memory-buffered approach, which takes maximum advantage of the fact that in most financial systems only a few specific table values constitute the majority of the requests. The RDI approach has eliminated over 90% of the
Figure 3—IBM DB/DC integrated financial system architecture overview

Legend:
F.S.  Foundation Software
A.S.  Application-Specific Software
reference data DL/I calls in our financial system for Standard Oil of Indiana.

The foundation software provides on-line inquiry (6) into the reference data tables and also provides ad hoc inquiry through a general-purpose inquiry package. The INQUIRY IV/IMS package from Informatics has been used for this purpose.

System assurance software (7) ensures that the application database retains integrity at all times. Not only is the technical integrity of the application database verified (no broken pointer chains, for example), but the substantive integrity is also verified. This capability is incorporated into the database design with planned redundancy and summary totals. In our experience this is an essential tool to help prevent system and application errors from corrupting the quality of the application data.

Reports are produced both by custom-written COBOL report programs, and by an ad hoc report generator (8), such as EASYTRIEVE/IMS from Panasophic Software, which is integrated into the foundation software.

Large systems typically generate scores of reports on a regular basis to be distributed to many recipients. It is time-consuming and expensive to manually burst, duplicate, and decompile the output of standard report programs for distribution to individual managers and staff personnel. AMS has developed and used successfully on a wide variety of projects foundation software that generates a custom packet of report pages for each recipient. This report-distribution system (9) is table driven, and it allows each individual to receive the correct number of copies of the desired reports, all organized into a neatly bound and indexed packet.