Reverse engineering of data

by P. H. Aiken

Data reverse engineering (DRE) is a relatively new approach used to address a general category of data disintegration problems. DRE combines structured data analysis techniques with rigorous data management practices. The approach is growing in popularity as an integrative systems re-engineering method because of its ability to address multiple problem types concurrently. This paper describes a general DRE template both as an activity model and as a data model to be populated with reverse engineered data. Scenarios show how DRE has been used to (1) harness data assets to address organizational data integration problems, (2) develop organizational data migration strategies, (3) specify distributed systems architectures, and (4) implement and propagate organizational CASE-tool usage to address system maintenance problems. Selectively applied DRE can be an important first step toward eventual organization-wide data integration.

Interest in reverse engineering is growing as organizations attempt to re-engineer existing systems instead of replacing them. When a system is reverse engineered, it is examined, documented, modeled, analyzed, and understood, in order to better inform subsequent efforts. Of additional value, the reverse engineering analysis outputs can be reused as a source of enterprise architecture components. Since successful systems re-engineering (SR) depends on effective reverse engineering, reverse engineering is viewed as a critical part of SR.

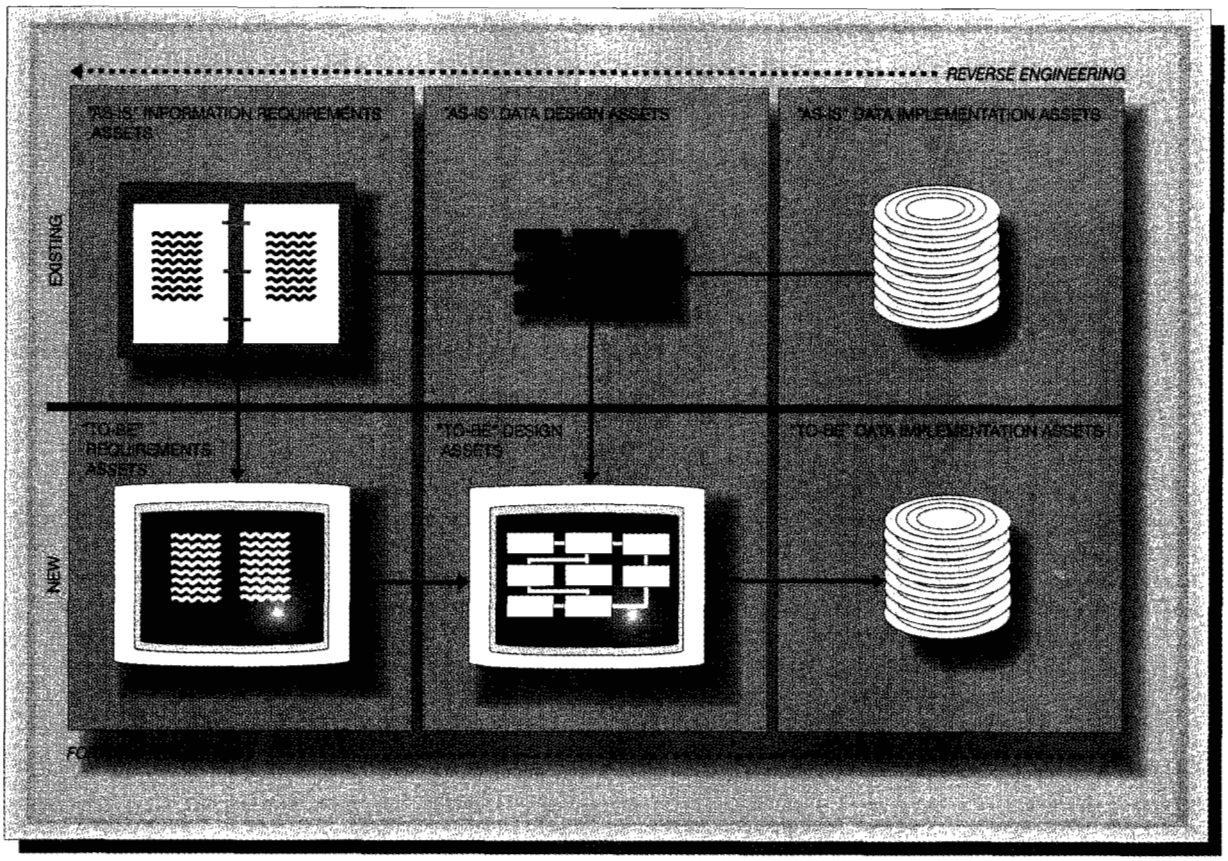
Figure 1 illustrates how SR is based on coordinated reverse- and forward-engineering activities, where forward engineering benefits from information gained by reverse engineering. A general SR goal is often stated as "delivering output meeting user requirements, using systems that currently do not." Organizations are turning to SR as a means of upgrading their existing information systems in situations where it appears to be a less expensive alternative to system replacement.[2] Three conditions can make it difficult for organizations to adapt their information systems to meet changing business needs: complex legacy systems, business re-engineering, and data access difficulties.

Complex legacy environments are frequently encountered. Many organizations have developed stand-alone, or "stovepiped" information systems (IS) that are both brittle and unintegrated. Over time, changing user and business conditions cause information integration requirements to continually evolve. Interfaces developed in response typically link the outputs of one system to the inputs of another, based on common understanding of the data. Interfaces define the requirements for periodic data exchanges among systems. Eventually brittle situations, such the example shown in Figure 2, are the result. Because these systems were not developed to easily exchange data, they do not. Changes in the payroll database, for example, might require corresponding changes to personnel and manufacturing applications. Changes to the personnel applications might require corresponding changes to the personnel database that may in turn also require still further changes to the manufacturing applications, etc.

Unintegrated and brittle information systems are also often barriers to a popular business process re-

0018-8670/98/$5.00 © 1998 IBM

Figure 1 Data re-engineering taxonomy—adapted from Chikofsky and Cross.[1] (Note: The requirements outputs are optional—it is possible to proceed directly from the "as is" design assets to the "to be" design assets, bypassing the requirements assets when they are not necessary.)

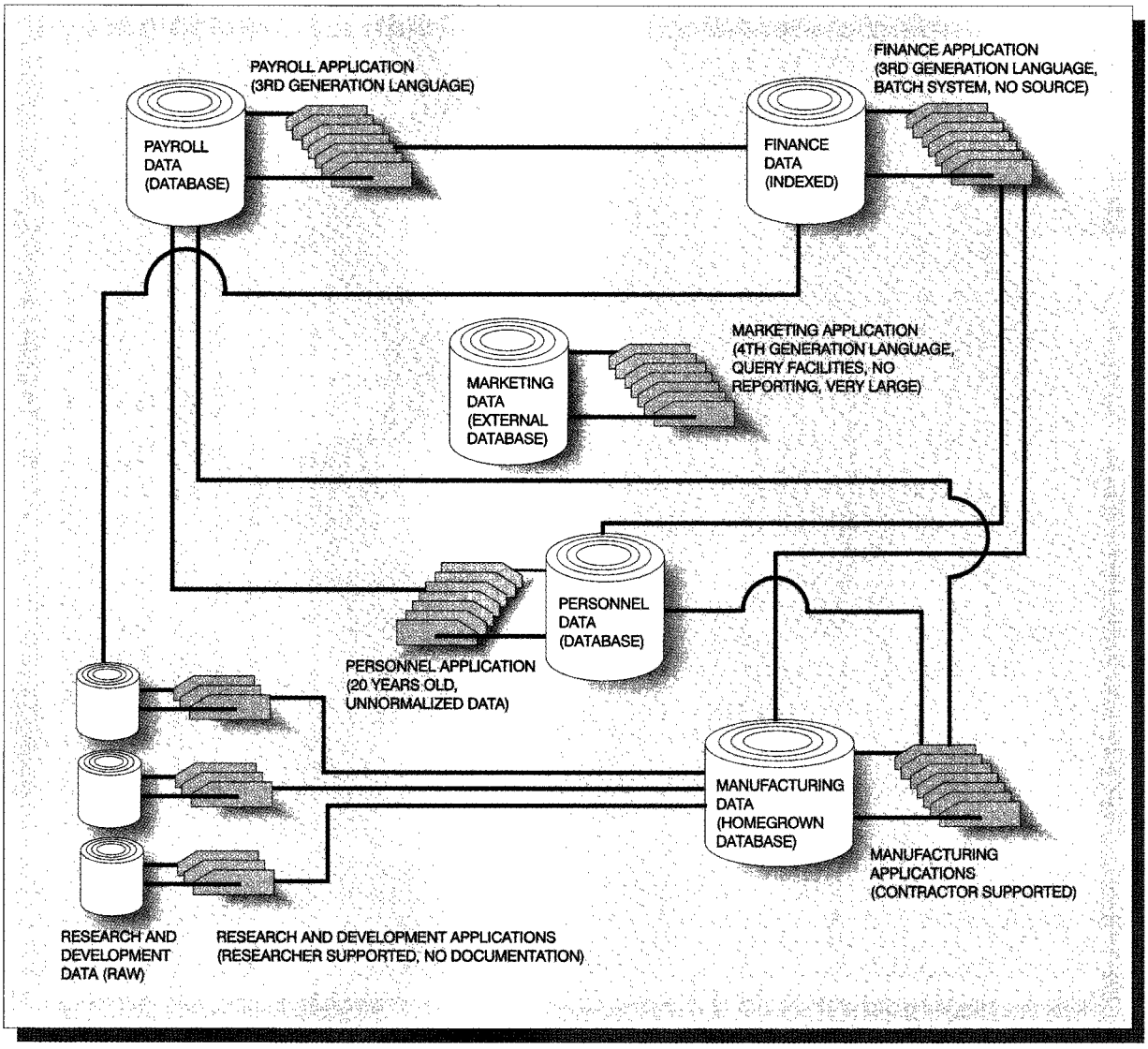


engineering (BPR) technology that uses shared data. Data sharing occurs when an organization logically integrates its data to meet multiple user requirements. Specified almost as universally as a "user-friendly interface," the concept of *shareable data* is a key technological requirement for many BPR efforts.[4–6] Data sharing is prerequisite to organizational integration. Before efficiently sharing data across the organization and with external partners, organizations must analyze and integrate their data.

Although data sharing is key to implementing many of today's business practices, data sharing difficulties within an organization can cause information to be difficult to obtain and expensive to maintain. These characteristics can effectively discourage sharing with external partners and block important growth opportunities. For example, organizations such as Wal-Mart Stores, Inc., prefer to conduct business with partners by directly exchanging data.[7,8]

Faced with these conditions, organizations are wondering where they should begin and what has worked for other organizations as they address problematic data issues. Reverse engineering of a system's data has proven to be a successful approach to reconstituting the understanding or the physical condition of organizational data systems that have deteriorated or become unclear. The remainder of this paper describes the reverse engineering of data as applied to resolving organization data problems. First the paper presents an overview of the reverse engineering of data (DRE), characterizing the current state of the practice and detailing an approach codeveloped by the author. Next it defines the reverse engineering of data using a DRE template and a DRE activity

Figure 2 A simplified version of Durell's[3] Gordian Knot with nine interfaces linking five stovepipe systems each supporting a functional area. (Note: Most functional business areas support multiple applications—increasing the actual number of interfaces proportionately.)



model. It then describes DRE guidance, analysis, and tools, followed by situations where DRE has proven successful. The paper closes with a discussion of the lessons learned.

Data reverse engineering

Reverse engineering goals are (1) to analyze a system, (2) to identify the system's components, (3) to identify the interrelationships of the system components, and (4) to create representations of the system in another form or at a higher level of abstraction.[9]

When considering re-engineering as a system enhancement methodology, the question arises as to what reverse engineering techniques should be applied. Types of reverse engineering actively being researched include system, software, database, and data.

An initial reverse engineering focus has been on software. The annual IEEE (Institute of Electrical and Electronics Engineers) reverse engineering conferences have been focused on software-oriented reverse engineering research, investigating topics such as the automation of techniques that answer questions such as: What does this program do? Why does the program do this? or How does the program perform this function?[10]

If the focus of a reverse engineering effort is on system or organizational data, the analysis should be labeled as *data reverse engineering* (DRE). DRE is defined as the use of structured techniques to reconstitute the data assets of an existing system.[11] DRE offers an effective means of addressing situations where:

- The scope of the investigation is on the system-wide use of data.
- The problem sparking the investigation is caused by problematic data exchange or interfaces.
- The re-engineering goals require a more strategic than operational analysis focus.

Consider as an example a situation (described later as Scenario 1) with more than 1400 application programs associated with the personnel system to be replaced. The functioning of just a few programs was of interest to the SR effort because the basics of personnel information management are generally understood (see Hay,[12] Figure 3.5) and because the vast majority of the programs were being replaced by new system components. With the exception of these few programs, individual program functionality was less important than understanding the system data-oriented input, output, and maintenance capabilities. These were analyzed so that potential replacement system capabilities could be assessed for their ability to satisfy the current and future organizational requirements.

A further distinction can be drawn between the reverse engineering of data and the reverse engineering of databases. In a series of publications, Blaha[13–15] and others have described many aspects of database reverse engineering. Because, by definition, databases possess certain homogeneous characteristics,[16] database reverse engineering is often a more structured version of data reverse engineering. Often the database schema, the meta-data, the directory structure, or other system descriptions can be reported automatically, leading to reverse engineering activities that are more tightly focused with respect to the project duration, reverse engineering technique, and tool set. On the other hand, because of greater likelihood of encountering nonstandard systems, DRE tends to be potentially more involved, broader in scope, and less well supported from a tool perspective.

Another situation that required DRE analysis was characterized by a "homegrown, one-of-a-kind" data management system utilizing fixed-length 5000 character records maintained by a system that serviced more than 100 different federal agencies. The data structure of the individual records was translated at run time using a series of conceptual schema overlays. Each record's layout was dependent on both its data content type and its agency affiliation, determining which overlay would correctly read the data. There was no chance of locating CASE (computer-assisted software engineering) tool support, and the data engineers had none of the traditional database structure rules to rely on when performing the analysis. Because of a low degree of automated support, DRE was accomplished manually by the data engineering team. For this situation, DRE was a cost-effective, data-centered approach to systems re-engineering, although automated techniques were not available or not materially useful.

DRE provides a structure permitting data engineers to reconstitute specific organizational data requirements and then implement processes guiding their resolution. Because it is a relatively new formulation of systems re-engineering technologies, most organizations are unaware of DRE as a technique and practice less structured approaches in response to data challenges.

The variation of DRE described here was developed as an outcome of the United States Department of Defense (DoD) Corporate Information Management Initiative, where the author's position as a reverse engineering program manager was to oversee the requirements engineering and formalization of thousands of management information systems requirements supporting DoD operations.[17]

In this and the following section, DRE is described in more detail using a DRE template, a DRE activity model, and a model of the data to be captured during DRE analysis—a DRE meta-data model.

A DRE analysis template. Table 1 illustrates a DRE analysis template providing a system of ideas for guiding DRE analysis, an overall meta-data-gather-

Table 1 DRE analysis template

No.	Activity	Outputs	Measures
	INITIATION PHASE		
1	Target system identification	Candidate target systems	"Size and shape" of the re-engineering challenges
2	Preliminary coordination	Target system points of contact	Measures of nontechnological complexity
3	Evidence identification and access	Target system evidence	Indications of system re-engineering feasibility
4	Analysis team initiation	Team directory	Determination of "return-on-investment" component derivations and starting date
5	Preliminary system survey	Analysis estimate data	Analysis baseline and an estimation process evaluation
6	Analysis planning	Analysis plan	Target measures are established and initial gathering begins
7	Analysis kickoff	Analysis charter and authorization	Articulation of initial system, financial, and organizational expectations
	IMPLEMENTATION PHASE		
8	Target system analysis	MR/V cycles (repeat until complete)	Cycle measures, team productivity data
8.1	Cycle planning	Focused plan for next cycle	Specific entities and attributes to be modeled
8.2	Evidence acquisition	Structured evidence	Indications of team cohesion and domain knowledge
8.3	Evidence analysis	Candidate entities and attributes	Time to develop data bank component
8.4	Straw model development	Data entities organized into models	Time to develop straw model
8.5	Model refinement and validation (MR/V)	Clearer, more accurate, validated models	Time to develop model validation, resources consumed to date and per session
8.6	Model storage and organization	Accessible models and associated information	Time to develop model storage, resources consumed to accessibility, data engineer productivity data
	WRAP-UP PHASE		
9	Data asset packaging	CASE tool-based data assets	Time stamp data, time to develop, and resources consumed per modeling cycle
10	Data asset integration	Integrated data assets	Time to develop and resources consumed to integration
11	Data asset transfer	Shared and shareable data assets	Time to develop and resources consumed to transfer
12	Analysis measures assessment	Additions to the analysis measures database	Measures assessment
13	Template refinement	Continually improving template and implementation capabilities	Analysis measures

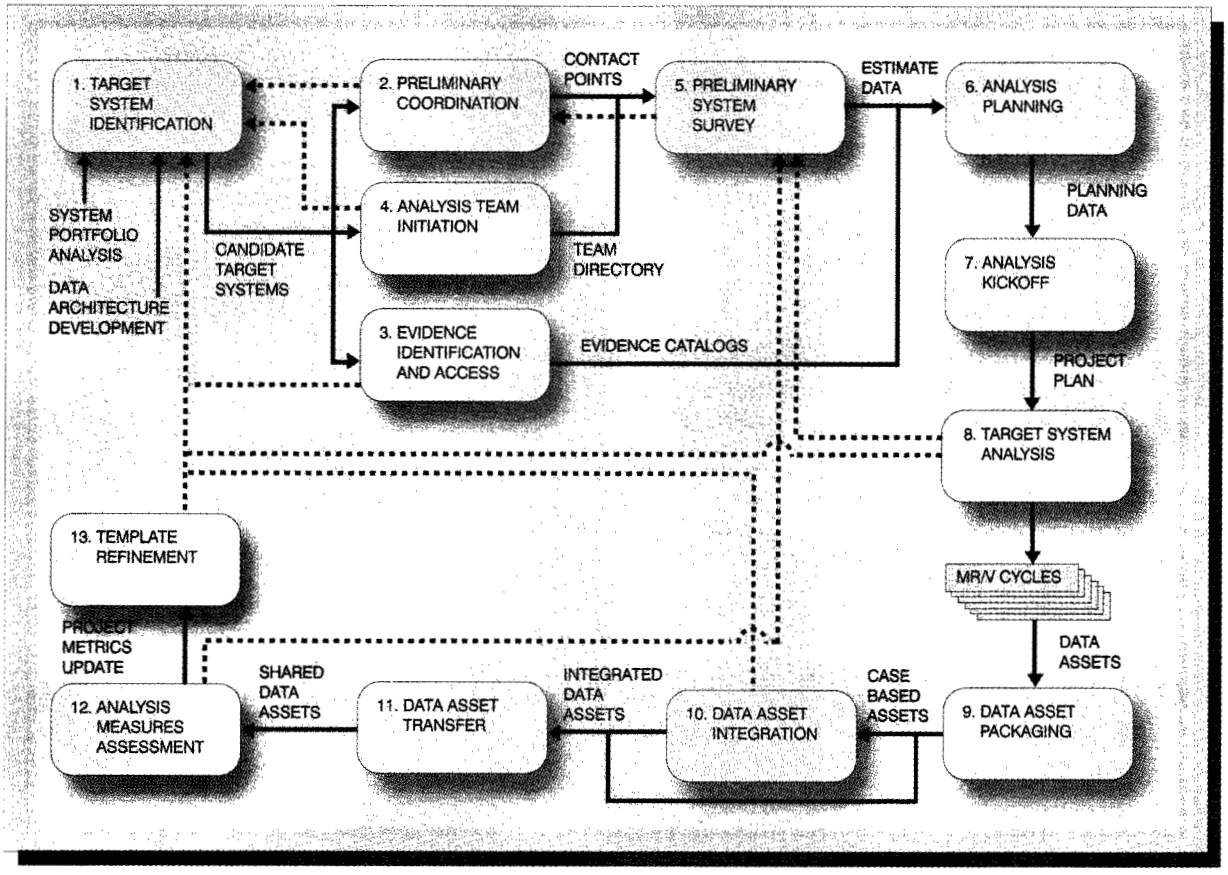
ing strategy, a collection of measures, and an activity, or phase structure that can be used to assess progress toward specific re-engineering goals.

The template has been used to facilitate project knowledge development on a number of data re-engineering projects. It consists of 13 activities comprising three analysis phases: initiation, implementation, and wrap-up. Outputs are used to leverage subsequent system enhancement efforts. Each DRE activity produces a specific output and associated activity measures. Production and acceptance of output delivery signals activity completion. For example, Activity 5, "preliminary system survey," results in the data contributing to the development of an analysis estimate. Estimate data establish the analysis baseline and also produce an initial assessment of the analysis estimation process that can be periodically reexamined.

In the next subsection, each template activity is described in the context of a DRE activity model.

DRE activity model. DRE analysis begins with typical problem-solving activities, first identifying, understanding, and addressing any administrative, technical, and operational complexities. Initiation activities are designed to ensure that only feasible

Figure 3 DRE activity model showing template inputs, activities, outputs, and feedback



analyses are attempted. Figuring prominently in the initiation phase is the development of baseline measures describing the reverse engineering analysis, at a conceptual level, in size and complexity. These measures are used to develop an analysis plan. Figure 3 shows the template activities configured into a DRE activity model. The dashed lines illustrate potentially useful feedback loops among activities.

Activity 1: Target system identification. The first DRE template activity is target system identification. The identification activity is required when organizational understanding of data systems has degraded or become confused. Data architecture development activities guide, and systems performance characteristics motivate and inform, target system identification. Activity 1 has two primary inputs—system performance data and, in particular, data on problematic system performance to help to identify specific data problems. Data architecture development needs can also influence the target system identification, providing a second incentive to reverse engineer. This occurs when a DRE output contributes to the correction of a system problem and at the same time produces a lasting organizational data asset that assists in the development of an organizational data architecture component.

Activity 2: Preliminary coordination. Since some systems are shared among organizational components with differing needs, the possibility exists for coordination difficulties. Preliminary coordination is required when systems serve multiple clients or when the reverse engineering can conflict with forward-engineering demands. In order to form the reverse-engineering analysis team, it is crucial to secure management approval to access the skills and knowledge of available key system and functional specialists

(sometimes called "key specialists"). The cross-functional nature of DRE leads to three "rules of thumb" for coordination:

1. Identified and prioritized system "stakeholder" objectives must be synchronized with the DRE objectives and priorities.
2. DRE analysis cannot be successful without coordinated system management commitment. High-level management approval is necessary but not sufficient. Other management and systems personnel must also understand and support the DRE analysis objectives; otherwise organizational politics may jeopardize analysis success.
3. Negotiation, planning, and "buy-in" processes must be complete before attempting analysis.

Activity 3: Evidence identification and access. Evidence identification and access has a broad definition. Obtaining access to evidence can range from explicitly obtaining key specialist participation, to getting CASE tool-readable versions of system dictionary data, to getting access to the proper versions of the system documentation. Data engineers assess the state of the evidence to estimate the effort required to develop a validated system model. Individual pieces of evidence can be classified as being in one of three possible states:

- *Synchronized.* Synchronized evidence accurately represents the current state of the system. Synchronized is the most desirable evidence classification state. System documentation that is produced and maintained using CASE technology is most likely to be synchronized. It has been also, unfortunately, the rarest.
- *Dated or otherwise of imperfect quality.* If documentation exists, it can be outdated or of poor quality. Dated system evidence reflects the system as it existed at a point in time. Changes made to the system since the evidence was created are not reflected. Other types of imperfection in data evidence could include corruption errors, technical errors, and value errors affecting completeness, correctness, and currency.[18] This is the category that describes most evidence available in DRE analysis.
- *Not useful or not available.* The worst possible situation occurs when documentation was never created, or has become subsequently not useful, or is unavailable.

Activity 4: Analysis team initiation. Initiation involves forming the analysis team, defining participation levels, and planning target system analysis. Team selection is important—members influence the articulation of business requirements. Once constituted, beginning with the preliminary system survey, team members collectively perform the remainder of the DRE analysis. To function effectively as a team, they need to understand the analysis goals in the context of an overall enterprise integration strategy.

Activity 5: Preliminary system survey. The preliminary system survey (PSS) is a scoping exercise designed to help assess the analysis characteristics for re-engineering planning purposes. Survey data are used to develop activity estimates. The purpose of the PSS is to determine how long and how many resources will be required to reverse engineer the selected system components. The PSS is concerned with assessing system dimensions according to several types of criteria, including:

- The condition of the evidence
- The data-handling system, operating environment, and languages used
- Participation levels of key systems and functional personnel
- The organization's previous experience with reverse engineering

Completed PSS results provide system characteristics used to develop a sound cost-benefit analysis and a useful analysis plan. Two structured techniques are applied during the PSS: functional decomposition and initial data model decomposition. Each results in a validated model that serves a specific role (described in the next subsection). Model development produces data useful for estimating the remainder of the analysis. The models then guide subsequent target-system analysis activities.

Activity 6: Analysis planning. Analysis planning involves determining (1) key specialist availability, (2) the number of analysis team members, and (3) the number of weeks of analysis team effort. Core system business functions are evaluated for overall complexity, described using model components that are combined with a functional-analysis-rate per hour. The activity output is an estimate of the number of weeks required to accomplish the analysis. The team derives the analysis characteristics as a function of three components that are instantiated using organization-specific data. The three components that determine analysis characteristics are: the relative condition and amount of evidence, the combined data handling, operating environment, and language fac-

tor, and the combined key-specialist participation and net-automation-impact component.

In general, the value of the term describing the combined data handling, operating environment, and language factor is greater than one. It serves as a confounding DRE characteristic, representing increased resources required to reverse engineer systems with obscure or unknown data handling, operating environment, or programming languages. This component typically increases the set of baseline characteristics established by the relative condition and amount of evidence component.

In contrast, the availability of key specialists and automation can significantly increase reverse-engineering effectiveness. Thus this component value typically ranges between zero and one, reducing the analysis characteristics represented by the combination of the first two components. The overall analysis estimate is determined as a function of the analysis characteristics and the historical organizational reverse engineering performance data.[19]

Activity 7: Analysis kickoff. Analysis kickoff marks the transition to implementation and the start of target system analysis. At this point it is useful to have achieved a number of setup milestones including:

- Identification and implementation of solutions to required coordination issues
- Education of colleagues and project team members
- Confirmation of participation commitments
- Participant consensus as to the nature of the investment in this enterprise integration activity

Activity 8: Target system analysis. Target system analysis is evolutionary in nature—modeling cycles are repeated until the analysis has achieved the desired results or (in some cases) the analysis has become infeasible. Modeling cycles use evidence analysis techniques to derive validated system models. This is the activity most often associated with DRE analysis. It is focused primarily on correctly specifying (at the same or at a higher level of abstraction) information capable of describing:

- *System information connecting requirements.* These are driven by the number of information sources and destinations; connecting in this context is defined as the ability to access data maintained elsewhere.
- *System information sharing requirements.* These are driven by the volume and complexity of the organizational information sharing and integration requirements; sharing is defined as the ability to integrate and exchange information across systems using a common basis for understanding of the data.
- *System information structuring requirements.* These are driven by the number and types of relationships between coordination elements; understanding system structures results in defined descriptions of user ability to extract meaning from data structures.

Target system analysis cycles are described in the next section.

Activity 9: Data asset packaging. Figure 4 illustrates multiple uses of packaged data assets. Activity 9 is generally completed by the data engineers who supervise data asset validation, documentation, and packaging in usable and accessible formats. Data asset packaging ensures that data assets are correctly packaged for delivery to other enterprise integration activities.

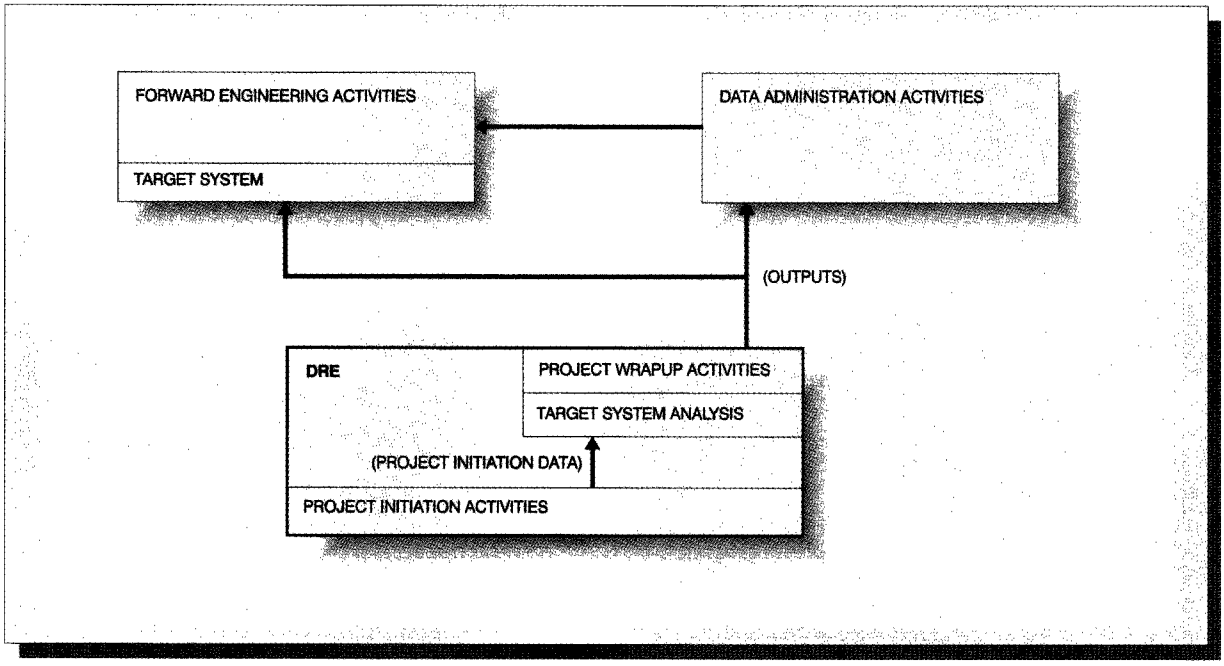
Two output formats are particularly useful:

1. A usually paper-based format that the analysis team can point to and say something like "The data assets created by this analysis are documented in this binder, and data administration can help you obtain electronic access to them." While printed versions are largely symbolic, the value of packaged data assets is in the representation of the largely intangible analysis required to produce them.
2. An electronic, CASE tool-based format managed by the functional community and maintained by data administrators. In organizations that have implemented CASE on an organization-wide basis, this information is readily accessible for other uses.

Because DRE analyses are made economically feasible by CASE tools, data-asset packaging often occurs continuously as the validated data assets are developed and added to the data bank. When models are "published" in the organizational data bank, they will be treated as organizational data assets facilitating and guiding future systems development.

Activity 10: Data asset integration. Because of the cumulative nature of DRE analysis outputs, the data assets developed during DRE analyses can be made

Figure 4 Multiple uses of packaged DRE analysis outputs



more valuable by integrating them with other data assets developed during other enterprise integration activities. Data asset integration involves, for example, explicitly addressing redundant data entities, data synonyms (where different terms have similar meanings), and data homonyms (same pronunciation but different meanings). This activity's goal is to resolve instances of data confusion and place the target system models in accurate perspective relative to other data assets. Outputs from Activity 10 are integrated data assets made more useful to the remainder of the organization through data administration programs.

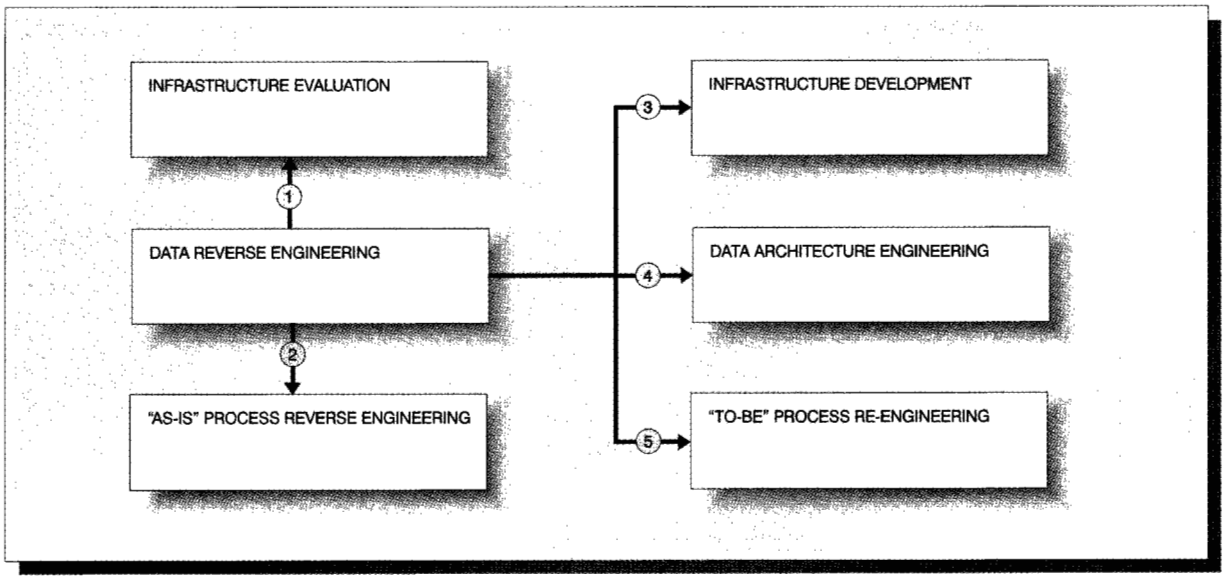
Activity 11: Data asset transfer. Template Activity 11 is formal recognition and enforcement of the fact that most DRE analyses produce outputs that are required by other enterprise integration activities. Making these assets available is the most tangible output of DRE analyses. Data asset transfer enforces the notion that DRE activities are designed to provide specific information useful to other enterprise integration activities.

Figure 5 illustrates how a single DRE analysis can produce five different types of assets useful to other enterprise integration activities. Potential data-asset transfers include:

1. Regular information exchanges, with concurrent infrastructure evaluation activities, help the organization to identify unmet gaps.
2. Data assets exchanged with "as-is" process reverse-engineering efforts concisely illustrate the existing organizational data capabilities.
3. System-related technology constraints and opportunities identified during DRE analysis often provide specific infrastructure requirements information to subsequent development activities.
4. Validated data assets are developed with the presumption that they will be integrated into the organizational data architecture.
5. An inventory of existing data assets, containing the type and form of current data, can provide information about existing but unrealized data opportunities (such as mining). These can be quickly turned into "low-hanging fruit" in "to-be" business process re-engineering activities.

Making data assets available can involve changing the media, location, and format of data assets to match requirements of other enterprise integration

Figure 5 Outputs of a single DRE analysis can provide inputs useful to multiple enterprise integration activities



activities. For example, situations may arise where organizations are changing CASE tools. In these instances, the data assets may be translatable from one tool format to another via various import/export utilities and exchange formats. Other asset transfer requirements may occur when the enterprise-level models need to be extended to link to operational concepts or additional data assets. The outputs of Activity 11 are data assets delivered on time, within budget, and meeting their intended purpose of proving useful as inputs to other enterprise integration activities.

Activity 12: Analysis measures assessment. After the analysis is complete, the team summarizes and assesses the measurement data gathered during the analysis. The assessment is used to establish and refine organizational DRE productivity data used in both planning DRE and strategically assessing enterprise integration efforts. Examples of summary measures collected include:

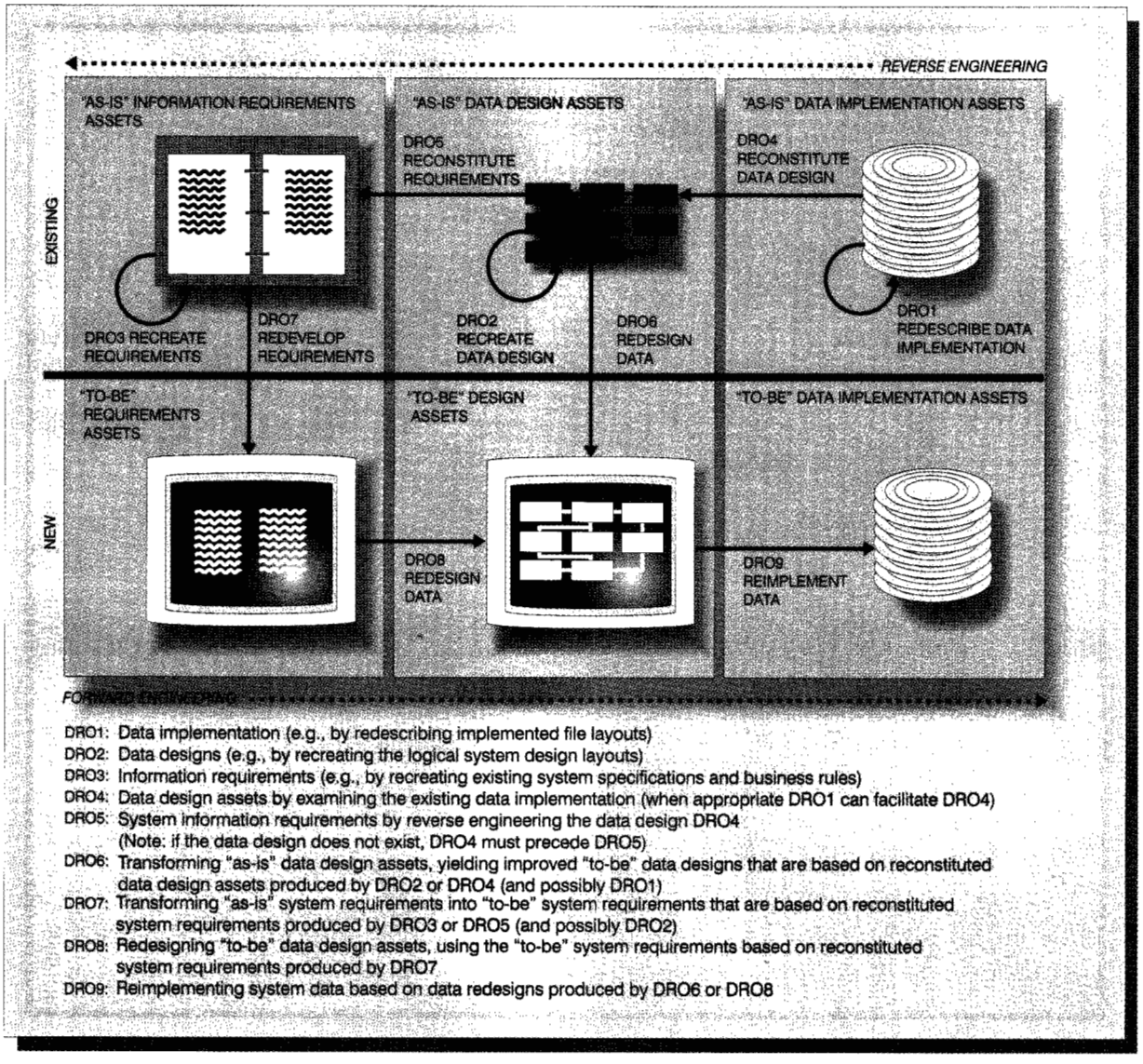
- The number of data entities analyzed
- The number of duplicate data entities eliminated
- The number of shared data entities identified
- The project rationale
- The expected financial benefit
- Information describing the overall analysis throughput
- Assessment of the key specialist participation
- Reactions of systems management to the analysis

The outputs of Activity 12 become another set of measures in the overall enterprise integration analysis data collection.

Activity 13: Template and implementation refinement. One of the most important analysis items is collecting and recording implementation measures, any refined procedures, tool and model usage data, and operational concepts. The outputs from Activity 13 in the template are focused on assessing and improving both the template and the subsequent implementation. The results and changes are archived to permit subsequent analysis.

The net worth of the analysis outputs often cannot be accurately evaluated immediately after the analysis. This is because the overall contribution of these outputs toward data administration goals and enterprise integration activities often becomes apparent only in the context of longer-term re-engineering activities. The nature of DRE analyses and all enterprise integration activities is such that the benefits increase in value as the results are integrated. DRE analysis should be periodically reviewed with hindsight to learn from the successes as well as the unexpected occurrences. Results of this activity are im-

Figure 6 Data re-engineering taxonomy illustrating possible data reverse engineering outputs



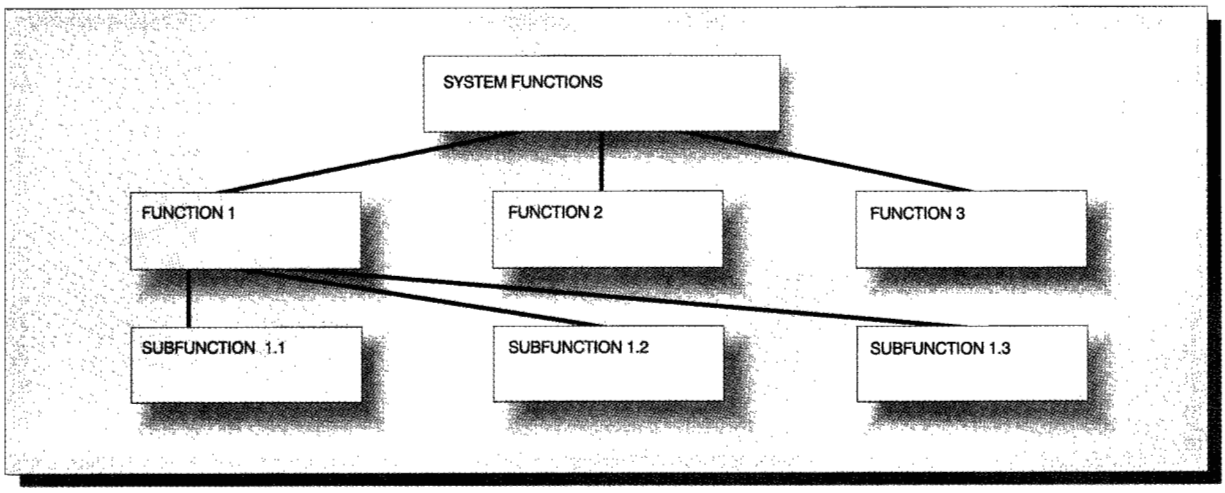
proved procedures, data on tool and model usage, and the template implementation assessments, giving closure to the analysis.

DRE guidance, analysis, and tools

As a structured technique, DRE analysis has three components: functional decomposition, data model decomposition, and target system analysis. Each should be developed using the following guidance:

- *Leverage of data management principles.* This involves understanding a relatively large amount of information by modeling and managing a relatively small amount of meta-data. When scoping DRE projects, it is useful to understand the possible types of data re-engineering outputs (DRO) illustrated in Figure 6. Optimizing DRE projects involves identifying and developing requisite subsets of DRO.
- *Modeling from integration points.* Unlike a jigsaw

Figure 7 A sample DRE functional decomposition



puzzle where it is important to begin at the edges, the structure of systems can often be understood most effectively by beginning with existing system interfaces and working into the system.

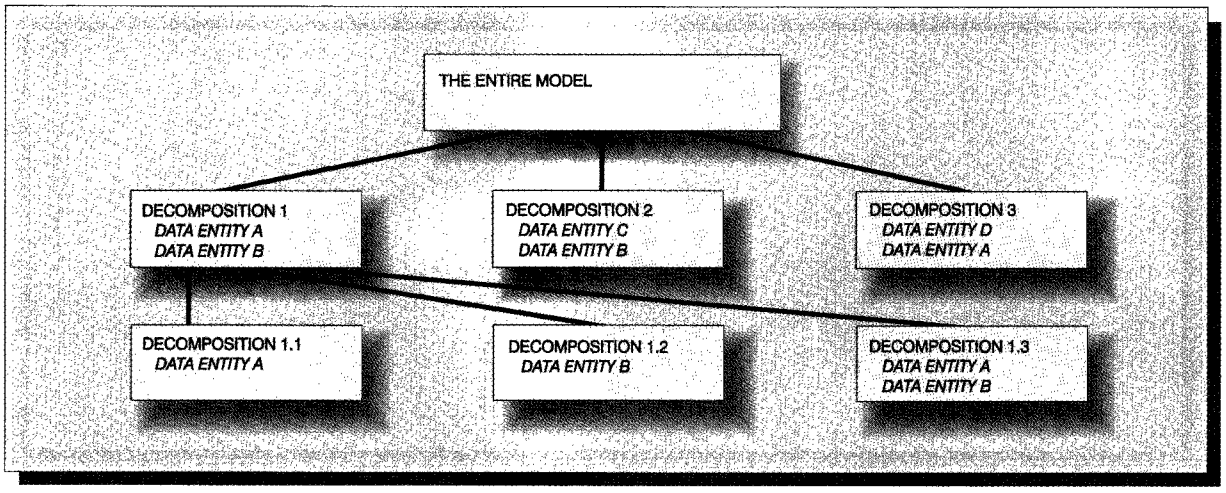
- *Immediate rapid development.* Candidate (or straw) versions of the models can be developed early. These quickly establish a common dialog among the analysis team and other involved personnel, such as the customer. Because it is often easier to critique than to create, it is better to confront a key specialist with an imperfect model than with a blank screen.
- *Living documents as imperfect models.* By acknowledging that the models can be currently imperfect, the organization treats the models as living documents that will evolve into more accurate versions throughout the analysis; this encourages constructive criticism from the collaborators and quickly draws newcomers into the process.
- *Critical mass.* The cumulative value of data assets increases at a more rapid rate as the degree of asset integration increases. The data assets produced are worth much more to an organization after they have been integrated with other data assets than by remaining as isolated groups describing individual systems or components. A key DRE goal is to, over time, expand the organizational knowledge structure with these data assets. As such, the relative value of the first data assets produced (or any single group of data assets) will be less than the value resulting from the integration of two or more data asset groups.

Functional decomposition. Figure 7 shows a sample functional decomposition. DRE analysis develops an accurate functional decomposition of the target system. In some instances this already exists because it is a basic form of system documentation. When a valid system decomposition must be reconstituted, the analysis goal is to describe the system according to classes of related functions instead of attempting to deal with numerous, individual functions. Functional decompositions are usually maintained in the form of structure charts, following standard diagramming conventions (e.g., Yourdon, DeMarco, Gane and Sarson).

In a functional decomposition, the system is described in terms of the functions performed within a single function (labeled "System Functions" in Figure 7). When accessing the electronic version of this chart, "double-clicking" on the system functions box reveals that it is comprised of three primary functions. Each function can be further decomposed into subfunctions that can be further decomposed—down to the smallest useful description.

Unlike forward engineering, where analysts decompose problems from the top down, in reverse engineering, the structure chart is often constructed from the bottom up by examining the system evidence. The answers are in the evidence; the question is "*What sort of functions are performed by the existing system*?" If analysis resources permit, it can be cost-effective at this point to specifically identify subfunction data

Figure 8 A sample data model decomposition



inputs and outputs, permitting development of data flow diagrams for system representations. Collectively this information is used during analysis planning to establish milestones and to assess system size and complexity.

Data model decomposition. Figure 8 is an example of a data model decomposition. While similar in appearance, this model is used to maintain information associating groups of related entities—to each other and to categories of access (i.e., create, read, update, and delete) by certain groups of users. Using the functional decomposition as a basis, each unit of decomposition is examined to determine whether it constitutes a work group or collaboration focal point. The goal is to develop candidate, then validated, arrangements of data entity groupings.

Data model decomposition is accomplished by key specialists helping to model data relevant to each functional area represented by the data model components. Once validated, the data entity groupings are used to reassess the functional decomposition validity and as the basis for developing further project milestones. For some systems there will be high correspondence between the data model decomposition and the functional decomposition. For others, the data model decomposition will reveal different underlying data structures. In these instances the differences can be examined for possible process reengineering opportunities.[20]

Target system analysis: Data reverse engineering meta-data. Table 2 details the implementation phase of the template. Target system analysis consists of modeling cycles. Modeling cycle activities can occur in various formats ranging from contemplative solitude, phone consultation, and structured interviews to evidence analysis and joint application development (JAD) or model refinement/validation (MR/V) sessions.

The goal is to develop validated models of aspects of the target system. Candidate models are developed using system data entities, the relationships between those entities, and organizational business rules. Candidate model development can be greatly aided by the use of available data model pattern templates (such as those cataloged by Hay[12]).

The models are developed using system evidence, as shown in Table 3. The models are analyzed, reviewed, and improved by both functional and technical analysis team members. Revisions and refinements are made to the models as new or clarified information comes to light during these sessions. Data assets produced during DRE are stored in the organizational data bank along with other relevant analysis information. Model components are integrated with other components as required. When a critical mass or sufficient quantity of models has been integrated, the information in the data bank becomes capable of providing useful, consistent, and coherent information to all levels of organizational deci-

Table 2 DRE template implementation phase is comprised of modeling cycles.

IMPLEMENTATION PHASE No.	Name	Activity	Output
8	Target system analysis (repeated until completed)		
8.1	Cycle planning	• Evaluating and incorporating previous cycle results • Identifying area of highest risk of lack of knowledge • Specifying analysis targets and a plan for the current modeling cycle	Focused plan for obtaining desired results from the next cycle
8.2	Evidence acquisition	• Collecting evidence • Cataloging evidence • Structuring evidence • Looking for missing evidence	Organized evidence
8.3	Evidence analysis	Analyzing evidence for appropriateness and model development potential	Candidate data entities
8.4	Straw model development	Creating candidate models	Data entities organized into models
8.5	Model validation/refinement	• Identifying changes in the model as a result of errors, new knowledge, and normalization • Documenting changes and further refining models • Validating models using appropriate techniques	Clearer, more comprehensive, more accurate, validated models
8.6	Model storage and organization	Collecting, cataloging, and structuring models for archival and configuration management purposes	Accessible models

sion making, creating conditions for better organizational functioning.

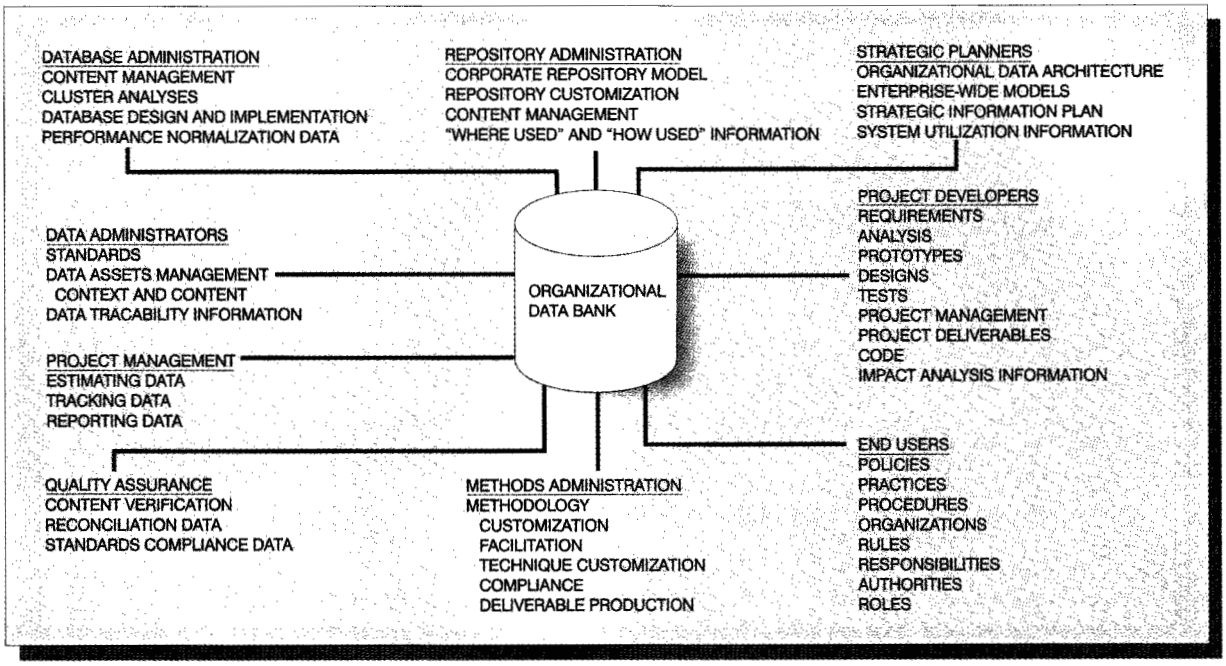
Figure 9 shows possible uses of DRE analysis information. DRE, and target system analysis in particular, focus on creating representations of the target system using appropriate entity-relationship and other data modeling techniques. Figure 10 represents the data required for DRE as a meta-data model—a model of the information capable of being captured during DRE (shown unnormalized to facilitate understanding).

The DRE meta-data model contains precise information, required to understand the target system, that was unavailable or disorganized before the analysis. Populating the DRE meta-data model is the primary focus of target system analysis. The analysis goal is to produce validated data for the meta-data model. For example, the goal of the functional decomposition (described previously) is to populate the Pro-

Table 3 System evidence categories

Evidence Category	Examples
Key specialists	Domain knowledge from the specialists, business rules
Processes	Functional descriptions, process models, code, user manuals
External data	Screen, report, interface specifications, interfaces to other systems
Conceptual data	Logical data models
Internal data	Program variables, data element lists, tables, file layout structures
Policies	Directives, guidelines, planning statements
System	Program source, object code, job procedures, libraries, directories, test cases, schemas, copylibs, make files, link maps, I/Os and other documentation, data

Figure 9 Possible data bank users (adapted from Selkow[21])



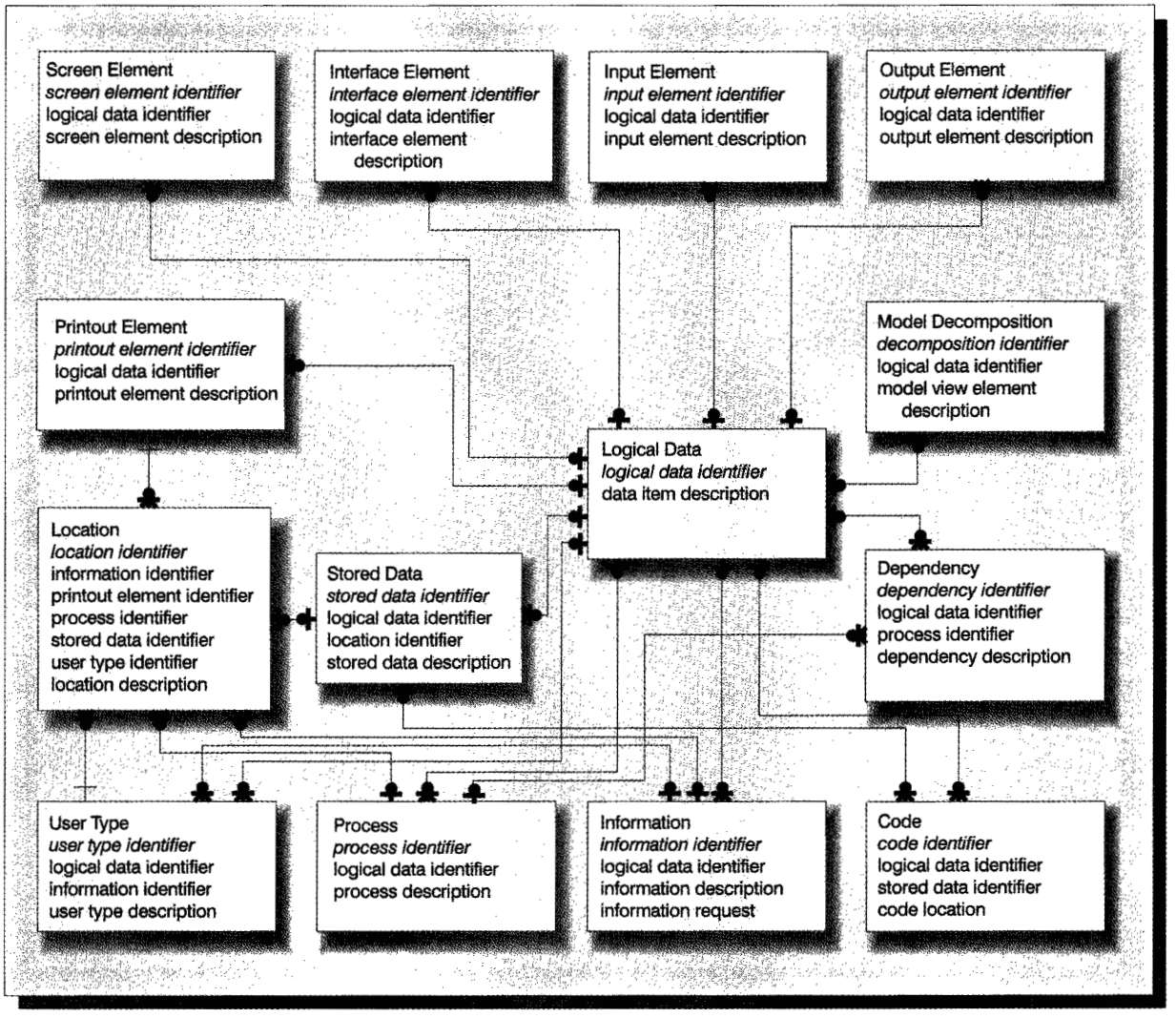
cess and Dependency entities. The goal of the data model decomposition (also described previously) is to populate the initial version of the data stored in the Logical Data and Model Decomposition entities of the meta-data model. Key to successful analysis planning is identifying just how many of the data are required in light of the analysis objectives. A description of the DRE meta-data model follows.

Visually, the model is centered around the data entities Logical Data and Stored Data. Logical Data entities are the conceptual things tracked by a system. Following standard definitions, Logical Data entities are facts about persons, places, or things about which the target system maintains information. Attributes are facts, grouped as they uniquely describe Logical Data entities. Additional understanding is obtained from the way each entity is related, or not related, to each other entity.

Stored Data entities are instances where a Logical Data entity is physically implemented. The Logical Data entity is populated with entity type descriptions. An association linking each Stored Data entity to one Logical Data entity indicates that eventually one Logical Data entity should be related to one or more Stored Data entities and each Stored Data entity should be linked to at most one Logical Data entity. This structure indicates a requirement to define every Physical Data entity by associating it with one Logical Data entity. Organization-wide data sharing can begin when Stored Data entities are commonly defined using Logical Data entity descriptions and applications process those data using the standard definitions. This mapping also permits programmatic control over the physical data using logical data manipulation.

At the top of Figure 10, four entities—Screen Element, Interface Element, Input Element, and Output Element—also have many-to-one associations with the Logical Data entity. Information describing each displayable field is maintained using the Screen Element entity. Each Logical Data entity is linked to every instance where system code causes the item to be displayed as a Screen Element field. When populated, a database described by the model will maintain information as specific as *screen element attribute W of screen X is a display of attribute Y of logical data entity Z*. (Each attribute can be displayed in multiple places in the system.) The many-to-one pattern of association with the Logical Data

Figure 10 **A DRE meta-data model showing *key* and other information that can be captured during DRE analysis. (Note: Many different types of attributes can be implemented for each entity—all represented with a single, nonkey entity description attribute.)**



entity is repeated for the Printout, Model Decomposition, Dependency, Code, Process, and Location entities. The analysis goal is to be able to link each system Input, Output, Interface, and Screen Element entity, with one, and only one, specific Logical Data entity, thus defining common use throughout the system.

The Model Decomposition entity association with the Logical Data entity indicates that each Logical Data entity exists on one or more model decompositions. (Recall that model decompositions are used to manage data model complexity by grouping data entities common to subsets of the overall model.) On the right-hand side of the model, the Dependency entity is used to manage interdependencies for data entities that are derived from within the system and other functional or structural representations. The Code entity contains references to each of the system code locations that access each Logical Data en-

tity. For example, *data entity W is generated by a code location X of job-stream Y that is maintained at location Z.*

The model indicates that the Information entity has the same association with the Logical Data entity, but the interpretation here is definitional. The Information entity is defined in terms of specific Logical Data entities provided in response to a request. Following Appleton,[22] data are a stored combination of a fact and a meaning. Information is at least one datum provided in response to a specific request. The request and any data provided in response are identified using the Information entity. The Information entity is also associated with one or more User Type entities that generate specific information requests. In addition, information is also associated with one or more specific locations where the data need to be delivered in order to be of maximum value. Similarly, the Printout Element entity accounts for printed fields. Printout is also associated with the location requiring the printout. A location has links to user types at a specific location, to the functions performed at that location, to the information requested by that location, and to any system code stored at that location. A function is defined as the process of spending resources to deliver specific information requested by a specific user type at a specific location.

Since target system analysis is cyclical in nature, the focus is on evolving solutions from rapidly developed candidate or straw models that are refined with subsequent analysis. Three primary types of data are produced as a result: traceability information, the data entity definitions, and the data map of the existing system. While these three are useful individually, they are made most useful when maintained in an integrated, CASE-based organizational data bank.

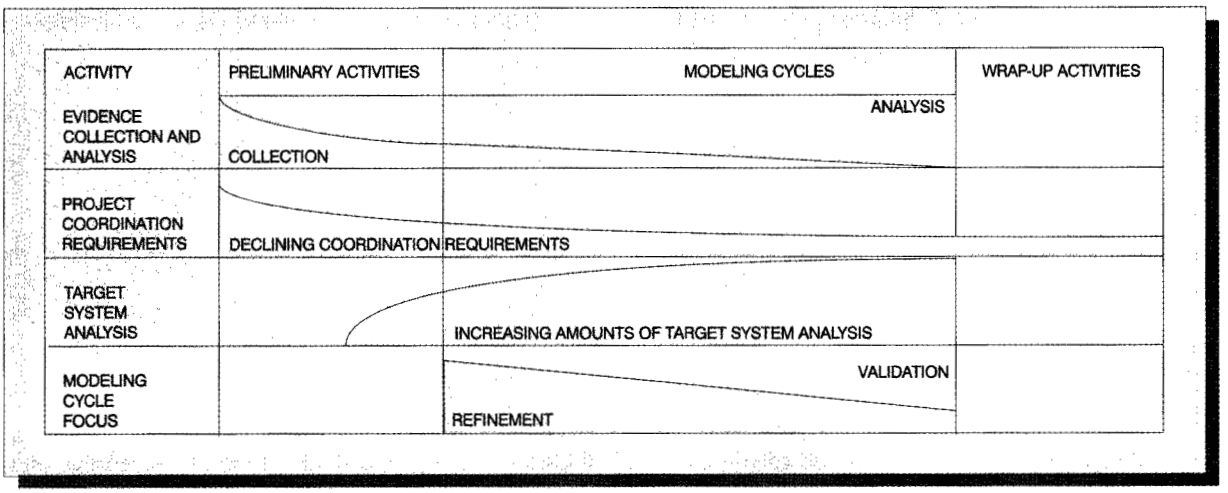
It is possible to accomplish target system analysis as a comprehensive examination of the system, beginning at one starting point and proceeding through the entire system. Usually this approach is unnecessarily cumbersome. Experience indicates that 80 percent of the time, DRE information requirements can be captured by focused analysis of 20 percent of the system. The question arises, how does the DRE team determine which is the 20 percent that they should focus on? This is guided in part by the scope of the models produced. Discrepancies between the functional decomposition and the data model decomposition should be targeted for early analysis to determine why the discrepancy exists between the users' perception of the system functions and the data entity groupings in the system that (in theory) support the functions. It is not effective to start at one "edge" of the system and plan to work through the entire system in a comprehensive manner until the DRE meta-data model is complete. Instead, allow the DRE analysis goals to determine what information is required for the analysis, target specific system aspects, and model these within their operational context. Data from this analysis are used to populate appropriate portions of the DRE meta-data model and to develop products capable of meeting analysis goals. Consider it an exercise of knowing the answers and determining the questions.

Developing and maintaining the completeness of the traceability matrices as specified by the DRE meta-data model is an important and challenging task. Since few CASE tools are capable of maintaining all of the required associations, organizations have been developing their own meta-data management support using, for example, combinations of spreadsheet, word-processing, and database technologies. As organizations become more proficient at DRE, the utility and ease of developing and maintaining the meta-data will increase. Many CASE environments support data-definition-language (DDL) production as a modeling outcome, permitting rapid development by evolutionary prototyping of components such as database structures, views, screens, etc.

The data bank is used to maintain all of the information in the DRE meta-data model. It contains entity definitions stored as part of the corresponding data map. Key here is to map system components directly onto the meta-data. The data model components derived from the system evidence are analyzed and entered into the CASE tool. The data map is constructed by defining and associating the data entity groupings identified as part of the preliminary system survey (PSS). Each data model decomposition is populated with attributes, including key information. As these are developed, they are assessed against existing system data entities to see if they match. Aliases are also cataloged and tracked.

Four specific changes in the modeling cycle activities should be observed during DRE analysis. Figure 11 shows how the relative amounts of time allocated to each task during the modeling cycle change over time. It also illustrates how the preliminary activities occur prior to the start of the first modeling cycle in order to obtain the PSS information. The modeling cycle activity changes include:

Figure 11 Relative use of time allocated among tasks during DRE analysis



- *Documentation collection and analysis*. Over time the focus shifts from evidence collection to evidence analysis.
- *Preliminary coordination requirements*. Coordination requirements can be particularly high in situations where managers are unaware of the analysis context or the target system's role in enterprise integration activities. Once target system analysis commences, coordination requirements should diminish significantly.
- *Target system analysis*. Just as the documentation and collection and preliminary coordination activities decrease, the amount of effort that can be devoted to target system analysis should increase steadily—shifting away from collection activities and toward analysis activities.
- *Modeling cycle focus*. By performing a little more validation and less refinement each session, the focus of modeling cycles shifts correspondingly away from refinement and toward validation activities.

The purpose of DRE analysis is to develop models matching the existing system state. Model components should generally correspond one-for-one with the system components. Normalization and other forms of data analysis are deferred to forward-engineering activities and are performed on a copy of the models used to maintain the existing target system meta-data. Additional information collected during this activity can facilitate the development of distributed system specifications. For example, additional meta-data entities useful in planning distributed systems and obtainable as part of reverse-engineering analysis are described in the next section.

Situations where DRE has proven successful

This section presents several scenarios illustrating how DRE analysis has proven successful in solving data problems. An interesting observation is that while DRE was developed as a part of system re-engineering, it has been effectively applied outside of that context (as in Year 2000 analyses).

Scenario 1: Distributed systems architecture specification. To better meet evolving customer requirements, a system manager planned to evolve two existing legacy applications from a mainframe base, by first combining them into a single, integrated two-tiered and then to a three-tiered client/server system. The multiyear plan was guided by an evolving, integrated data re-engineering effort. DRE formed the basis for the data migration plans transforming the original systems to the two-tiered architecture. The integrated models were developed by reverse engineering the two existing systems. The functional decomposition and data model decomposition assisted in the development of specific data model views that were prototyped with the various user communities, using CASE tool-based DDL output. The integrated data model consists of 126 entities and more than 2800 attributes. The completed two-tiered implementation consisted of more than 1500 tables

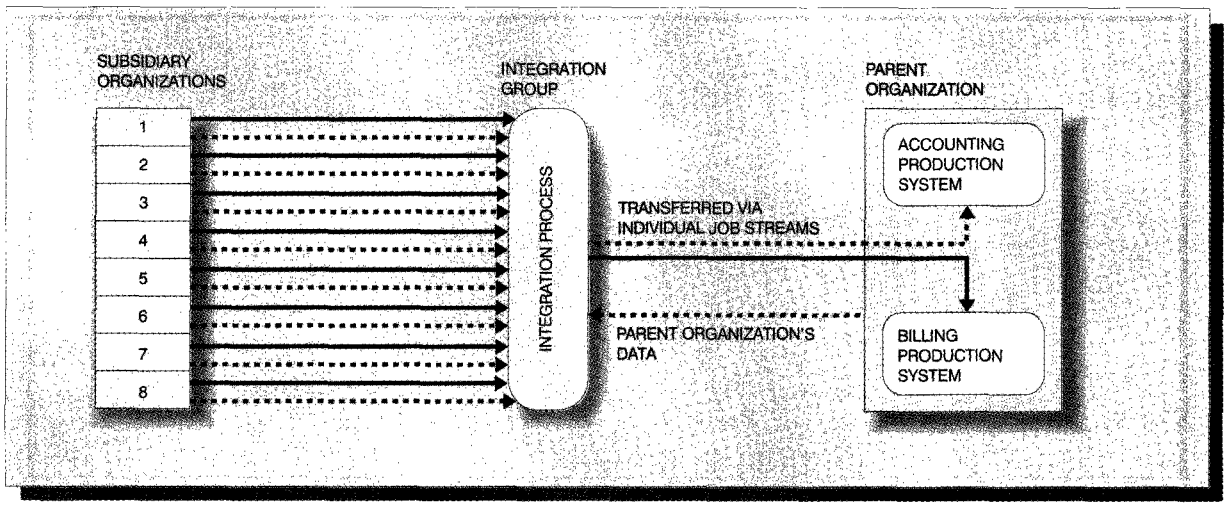
Table 4 Meta-data attributes useful in client/server application development that are obtainable as part of DRE analysis (meta-data derived from Inmon[23])

No.	Primary Use	Meta-datum	Domain Value
1	Architectural development	Data quality type	Public Private
2		Data usage type	Operational Decision Support System I: single server node Decision Support System II: multiple server nodes Decision Support System III: distributed data and processing
3		Data residency type	Functional Subject Geographic Other
4	Application development	Archival data type	Continuous Event (discrete) Periodic (discrete)
5		Data granularity type	Smallest unit of addressable data is • Attribute • Element • Some other unit of measure
6	Architectural development	Data performance issues	Performance measurement/period of measurement
7	Application development	Data access frequency	Accesses/period of measurement
8		Data update frequency	Updates/period of measurement
9		Data access probability	Likeliness that an individual data element of the total population will be accessed during a processing period
10		Data update probability	Likeliness that an individual data element of the total population will be updated during a processing period
11	Architectural development	Data integration requirements	Number and possible classes of nodes
12		Data subject area	Number and possible subject area breakdowns
13		Data grouping	Useful for cataloging user-defined clusters
14	Application development	Data location	Number and availability of possible node locations
15		Data stewardship	Range of all business units
16		Data system of record	System responsible for maintaining data element's data

in a relational database. When the planning for the two-tiered implementation was completed, the data engineers returned to modeling, this time to populate the Information, Location, and User Type entities, supplementing the meta-data with 16 additional attributes (see Table 4). These extended data models are being used as the basis to develop data architecture specifications for the three-tiered target-system architecture. Subsequent analysis described specific user classes possessing different information requirement types at hundreds of different locations. Understanding the distributed information requirements, by location, will result in the mapping of data between users and information requirements. This will result in key, strategic system-planning elements. (More information on this project is available, describing the data re-engineering,[24] business process re-engineering integration,[20] development of the meta-data,[25] and project decision analysis context.[26]

Scenario 2: Data integration problems. In a series of acquisitions, eight utility companies were merged with a parent organization. A data integration group was established to organize and produce job streams from the eight subsidiaries' data. Each of the eight subsidiaries transferred separate billing and account-

Figure 12 Data integration context diagram indicating the requirements to modify data to conform to expected system requirements, consolidate into a whole job stream, and perform a series of edit checks in preparation for transfer to the production systems



ing data to the integration group. The integration group's mission was to consolidate subsidiary with parent organization data, and to remove errors from the job streams prior to transfer to the production systems (Figure 12).

After encoding the data so they could be traced back to the originating system, the integration group performed a lengthy list of edit checks on the incoming data. When the data were thought ready, the integration group transferred the now "scrubbed" data via a job stream interface to the parent organization's production systems. The production systems responded to bad data by failing. All the data for an entire cycle had to run at once, completely and without data errors, in order to produce any output. In spite of rigorous scrubbing, correcting repeated problems has cost significant resources as both systems repeatedly failed due to bad or missing data. A puzzling characteristic was that no two problems encountered seemed the same—a unique data problem apparently produced each failure.

The solution was to focus the DRE analysis on the data at the interface to the production systems, and to work backward into the integration group processing. A PSS determined the analysis challenge and established baseline measures. The Logical Data, Stored Data, and Interface Element entities were modeled. The models became a systematized data asset, formally describing the production system data input requirements and permitting systematic analysis of each subsidiary's data. These models provided the starting point for further analysis and discussion between these organizations. Each subsidiary organization's individual data streams were systematically compared to the modeled interface data specifications. The previous practice had been to correct each data error in subsidiary data input streams.

Once populated, the DRE meta-data model permitted programmatic data protection and maintainability. Delays associated with the accounting and billing production were reduced to the point where the integration group was no longer needed. The organization chose to reuse their experience to help reengineer other systems.

Scenario 3: Developing data migration strategies. A public-sector-run mainframe-based system was to be upgraded. The custom-developed application served an entire functional area and contained program elements more than 20 years old. While the system functioned correctly and effectively, only two individuals in the organization understood the structure of its homegrown data management system. Fixed-length, 5000-character records were coded, linked, and composed using thousands of different combinations to maintain data for many different organizations. The government funded an upgrade to

replace the data management system. A question was raised as to the implementation of the new data management system. Some argued for a relational database management system for maximum data flexibility. Others claimed the anticipated query volume would be too great for a relational implementation and insisted that alternative models were more appropriate.

The solution was developed by formally modeling the existing system data as part of the data migration planning. The PSS determined that almost 100 different functions were embedded in the system—leading to the development of a corresponding model decomposition. The PSS also indicated that the analysis would require a six-person analysis team and two months to complete the model. PSS results directed the analysis effort to populate the meta-data model with information linking Logical Data to Screen Element, Printout Element, and Interface Element entities. More than 500 Stored Data entities were linked via Logical Data entities to 100 key reports, screens, and interfaces. A set of 200 Logical Data entities were documented. The completed model also documented more than 200 business rules. It was determined that the query volume could be reduced to 20 percent of the original by developing a separate data warehouse, permitting intranet access to typically requested information that would be extracted periodically from operational data. Based on this system design, a relational database management system was selected to implement the new data management system. The team used a CASE tool to maintain the required analysis data, including the system data model and analysis data dictionaries.

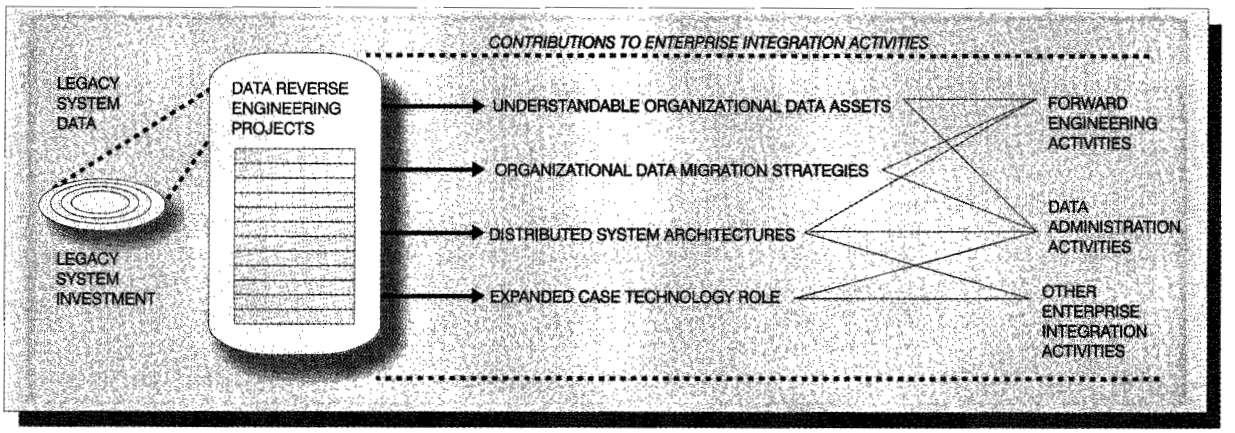
Scenario 4: Improving system maintenance with CASE. As a result of a merger, a new work group was established to perform maintenance on a 1960s-vintage application system. In the mid-1980s, a consulting partner introduced CASE technology as part of a codevelopment effort. The partnership failed; the employees who had been trained in the use of the CASE tool were downsized; and the system documentation was not kept synchronized with the maintenance application. The new work group wanted to quickly become knowledgeable about the system and was also CASE-illiterate. The team leader decided to address both issues simultaneously, and acquired the most recent version of the CASE tool. The next step was to develop a CASE training program for the work group, focusing on recovering the system data assets using the CASE tool. This tool supported the automated development of data models from existing physical data structures by importing the schemas into the tool. Much of the DRE meta-data model was quickly populated by the work group as part of the training exercise, including the Printout, Screen, Interface, Input, Output, and Stored Data entities. From these, the system functional decomposition and data model decomposition were developed, as was the system data dictionary and data map—the organization had never previously developed this form of system documentation. This information was compared to the most recent system documentation. Once it had accurately reconstituted the system documentation, the work group:

- Developed a much better system understanding as a result of the DRE-based CASE training exercises
- Increased its effectiveness in estimating proposed system changes, due to better understanding of the system models
- Became more effective in system maintenance as a result of greater familiarity with the system component interactions
- Gained more knowledge as to how the system fit into the larger organizational information-processing strategy
- Was increasingly consulted for advice on data problem correction, functioning as an organizational re-engineering resource

Under these circumstances, the solution was found in the synergy between applying system maintenance and developing work-group CASE tool experience. Using the CASE tool's ability to programmatically reverse engineer the system data, the team used their growing DRE knowledge of the system to facilitate CASE tool understanding and vice versa. By reverse engineering the data using a CASE tool, the work group became more knowledgeable of both the CASE technology and the system itself. In recognition of increased work group performance, members were asked to become first consultants and then data engineers on other analyses.

Year 2000 analyses. DRE has an obvious application in addressing Year 2000 (Y2K) data problems, easily providing structure for Y2K investigations. Thorough DRE analysis can well prepare an organization to be open for business on Monday, January 3, 2000. Target system analysis can be highly correlated with the activities performed as part of organizational Y2K analyses. If approached from a DRE perspective, during target system analysis, date-oriented or -derived data can be flagged for further, Y2K-specific anal-

Figure 13 A DRE template used to provide a basis for other enterprise integration activities



ysis. The examination of the data and confirmation of Y2K compliance can be accomplished by storing the data elements in a CASE repository. After completing DRE, an organization is in a much stronger position with respect to the Y2K problem.

Lessons learned

Data reverse engineering represents an emerging technology capable of serving multiple organizational roles. Particularly in systems re-engineering contexts, it can be used by managers interested in aligning their existing information systems assets with organizational strategies to accomplish more effective systems re-engineering. Selectively applied, DRE can also be an important first step toward increased organizational integration. Data-based success stories, such as AT&T's entry into the credit card business, MCI's Friends and Family** program, and the airline industry's systems supporting reservation and frequent-flyer programs, demonstrate the value of capitalizing on organizational data to implement successful business strategies.[27] Management is becoming aware of the true value of data as an organizational resource, ranking data second (behind "organizational architecture development" and in front of "strategic planning") in a survey of 1990s MIS (management information system) management issues.[28] Figure 13 illustrates that DRE analysis outputs describing an existing system can be used as a common source from which other enterprise integration activities result.

The 1997 re-engineering market was estimated to be $52 billion—with $40 billion to have been spent on systems re-engineering.[29] Understanding how DRE can provide a basis for other enterprise integration efforts prepares managers to recognize conditions favorable to its successful utilization. To cite an instance, the project manager in Scenario 1 realized the value of reverse engineering two existing systems and subsequently directed our research team to reverse engineer the newly delivered, widely installed, commercial software application system in the belief that the effort would also be productive.[30] In this instance, the exercise achieved four primary organizational incentives for data reverse engineering within the project context:

1. Bringing under control and directing the organizational data assets for integration and sharing
2. Identifying data migration strategies by understanding existing organizational information requirements and developing corresponding data migration plans
3. Providing an information base for use in developing distributed system architectures capable of meeting organizational needs
4. Expanding the role of CASE-based technologies within the organization beyond their traditional role in new systems development

A future data reverse engineering research agenda includes investigation into additional system meta-data uses. Leveraging meta-data can contribute to other enterprise integration activities including:

- *Integration with object modeling.* The reverse engineering meta-data can be used as the basis for organizationally evolving or transitioning to an ob-

ject orientation. The capabilities of CASE tools that can integrate object and data meta-data will be the subject of research investigations.

- *Development of "common use meta-data."* If it is possible to define common use meta-data, the research focus can shift away from understanding the meta-data contents and toward meta-data use by application developers building on current repository technology, sought after by Microsoft[31] and others with meta-data standardization projects.
- *Expert systems.* Incorporation of organizational expertise into meta-data presents an intriguing challenge. Future research plans include examining the degree to which the organizational meta-data can provide expert-system-based advice on human resource policy implementation.
- *Data warehouse engineering.* In Scenario 1, the project manager does not have the resources to rebuild the data warehouse—it is a situation where the implementation must be correct the first time. Effective meta-data use is required to correctly engineer data warehouses.[32]

Acknowledgments

Much of the content for this paper was prepared in response to an invitation to address a group of information system managers in the Netherlands in early 1996 on the subject of legacy systems re-engineering. This opportunity was organized by Volken de Jong of Origin Groningen. The paper benefited enormously from critiques by several of my colleagues within the DoD Data Administration Program and the comments of four insightful but anonymous reviewers. The DRE meta-data model was developed using Visible Advantage** from Visible Systems Corporation of Waltham, Massachusetts. The underlying research was sponsored in part by the State of Virginia Department of Personnel and Training. Bill Girling, manager of systems for the department, saw the need and developed the initiative. Many classes of data engineers have worked on DRE projects as part of their curricula in information systems in the School of Business at Virginia Commonwealth University (VCU). In particular, I would like to thank VCU's Information Systems Research Institute research associates Kim Boos, Leslie Borman, Lewis Broome, Sasipa Chankaoropkhun, Pawan Pavichitr, and Sirirat Taweewattanaprecha for their professional contributions to these projects.

**Trademark or registered trademark of MCI Communications Corporation or Visible Systems Corporation.

Cited references

1. E. Chikofsky and J. Cross, "Reverse Engineering and Design Recovery: A Taxonomy," *IEEE Software* **7**, No. 1, 13–17 (January 1990).
2. R. N. Karr, *Data Management Issues Associated with Stovepipe Systems*, General Services Administration/Information Resources Management Service/Policy Analysis Division, Report KMP-94-1-I (October 1993).
3. W. Durell, *Data Administration: A Practical Guide to Data Management*, McGraw-Hill, Inc., New York (1985).
4. D. Stoddard and C. J. Meadows, "Capital Holding Corporation—Reengineering the Direct Holding Group," in J. Cash, R. Eccles, N. Nohria, and R. Nolan, *Building the Information-Age Organization: Structure, Control, and Information Technologies*, Richard D. Irwin, Boston, MA (1994), pp. 433–452.
5. J. R. Caron, S. L. Jarvenpaa, and D. B. Stoddard, "Business Reengineering at CIGNA Corporation: Experiences and Lessons Learned from the First Five Years," *Management Information Systems Quarterly* **18**, No. 3, 233–247 (September 1994).
6. *Beyond the Basics of Reengineering: Survival Tactics for the '90s*, Quality Resources, The Kraus Organization Industrial Engineering and Management Press, Institute of Industrial Engineers, Norcross, GA (1994).
7. L. Wilson, "Cautious Change for Retailers," *Information Week*, 177–180 (October 10, 1994).
8. S. Haeckel and R. Nolan, "Managing by Wire," *Harvard Business Review* **71**, No. 5, 122–132 (September/October 1993).
9. E. J. Chikofsky, "The Database as a Business Road Map," *Database Programming and Design* **3**, No. 5, 62–67 (May 1990).
10. *Proceedings of the First Working Conference on Reverse Engineering*, May 21–23, 1993, Baltimore, MD, IEEE Computer Society Press (1993).
Proceedings of the Second Working Conference on Reverse Engineering, July 14–16, 1995, Toronto, Canada, IEEE Computer Society Press (1995).
Proceedings of the Third Working Conference on Reverse Engineering, November 8–10, 1996, Monterey, CA, IEEE Computer Society Press (1996).
Proceedings of the Fourth Working Conference on Reverse Engineering, October 6–8, 1997, Amsterdam, the Netherlands, IEEE Computer Society Press (1997).
11. P. Aiken, *Data Reverse Engineering: Slaying the Legacy Dragon*, McGraw-Hill, Inc., New York (1996).
12. D. Hay, *Data Model Patterns: Conventions of Thought*, Dorset House Publishing, New York (1995).
13. W. Premerlani and M. Blaha, "An Approach to Reverse Engineering of Relational Databases," *Communications of the ACM* **37**, No. 5, 42–49, 134 (May 1994).
14. M. Blaha, "Observed Idiosyncracies of Relational Database Designs," *Proceedings of the Second Working Conference on Reverse Engineering*, July 14–16, 1995, Toronto, Canada, IEEE Computer Society Press (1995), pp. 116–125.
15. M. Blaha, "Dimensions of Relational Database Reverse Engineering," *Proceedings of the Fourth Working Conference on Reverse Engineering*, October 6–8, 1997, Amsterdam, the Netherlands, IEEE Computer Society Press (1997), pp. 176–183.
16. E. F. Codd, "Relational Database: A Practical Foundation for Productivity," *Communications of the ACM* **25**, No. 2, 109–117 (February 1982).
17. P. Aiken, A. Muntz, and R. Richards, "DoD Legacy Systems:

Reverse Engineering Data Requirements," *Communications of the ACM* **37**, No. 5, 26–41 (May 1994).
18. Y. Yoon, P. Aiken, and T. Guimaraes, "Defining Data Quality Metadata: Toward a Life Cycle Approach to Data Quality Engineering," under review by the *Information Resources Management Journal*.
19. P. Aiken and P. Piper, "Estimating Data Reverse Engineering Projects," *Proceedings of the 5th Annual Systems Reengineering Workshop*, February 7–9, 1995, Monterey, CA (1995), pp. 133–145. Also available as Report RSI-95-001 from Applied Physics Laboratory Research Center, Johns Hopkins University, Baltimore, MD.
20. L. Hodgson and P. Aiken, "Synergistic Dependence Between Analysis Techniques," *Proceedings of the 9th Software Technology Conference*, April 27–May 2, 1997, Salt Lake City, UT; published on CD-ROM by the Air Force Software Technology Support Center, http://www.stsc.hill.af.mil.
21. W. Selkow, "Strategic Information Planning: A New Framework," *Enterprise Systems Journal* (1990).
22. D. Appleton, "Business Rules: The Missing Link," *Datamation* **30**, No. 16, 145–150 (October 1984).
23. B. Inmon, *Data Architecture: The Information Paradigm*, QED Technical Publishing Group, Boston, MA (1993).
24. P. Aiken and B. Girling, "Data Reengineering Fits the Bill," *InformationWeek*, 8A–12A (May 26, 1997).
25. P. Aiken, "Integrating the Reverse and Forward Engineering of Data Using Metadata Models," *Proceedings of the 10th Annual DAMA-NCR Symposium*, September 11–12, 1997, Washington, DC (1997), pp. 103–160; available from National Capital Region, DAMA, McLean, VA.
26. See the IHRIS Project Website at http://www.isy.vcu.edu/~paiken/dre/ihris.htm.
27. E. Chikofsky, "The Necessity of Data Reverse Engineering," *Data Reverse Engineering: Slaying the Legacy Dragon*, P. Aiken, Editor, McGraw-Hill, Inc., New York (1996), pp. 8–11.
28. F. Neiderman, J. Brancheau, and J. Wetherbe, "Information Systems Management Issues of the '90s," *MIS Quarterly* **15**, No. 4, 475–499 (December 1991).
29. B. Caldwell, "Missteps, Miscues: Business Reengineering Failures Have Cost Corporations Billions, and Spending Is Still on the Rise," *InformationWeek*, 50–60 (June 20, 1994).
30. P. Aiken and O. Negwenyana, "Reverse Engineering New Systems: An Implementation Support Perspective," under review by *IEEE Software*.
31. See http://www.microsoft.com/repository/start.htm for details.
32. C. Finkelstein and P. Aiken, *Data Warehouse Engineering with Data Quality Reengineering*, McGraw-Hill, Inc., New York (scheduled for October 1998 publication).

Accepted for publication January 15, 1998.

Peter H. Aiken *Virginia Commonwealth University, Department of Information Systems, Richmond, Virginia 23284 (electronic mail: paiken@acm.org).* Dr. Aiken's B.S. and M.S. degrees were received from Virginia Commonwealth University. He was awarded the Ph.D. degree in information technology by George Mason University (GMU) in 1989. Subsequently, he joined the GMU faculty as a visiting assistant professor of information systems and systems engineering. In 1992 he was recruited by the Department of Defense to work in the Center for Information Management's (CIM) Information Engineering Directorate. At CIM, he directed a multimillion-dollar DoD-wide reverse engineering program aimed at recovering data architectures from existing information systems. From 1992 to 1997 he held the position of Computer Scientist, most recently with the Office of the Chief Information Officer. In 1993 Dr. Aiken joined the faculty of the Department of Information Systems at Virginia Commonwealth University. He has assisted a number of organizations worldwide with their data management strategies and systems implementation initiatives. His research is reported in several scholarly journals, and he has been invited to speak at a number of foreign as well as domestic conferences and events.

Reprint Order No. G321-5676.