CHAPTER

6 Immutability and Immortality

OUTLINE

You live your life forward, but you understand it backward. **Soren Kierkegaard**

BACKGROUND

Immutability is one of those issues, like identifiers and introspection, that seems unimportant until something goes terribly wrong. Then, in the midst of the problem, you realize that your entire information system was designed incorrectly, and there really is nothing you can do to cope.

Here is an example of an immutability problem. You are a pathologist working in a university hospital that has just installed a new, $600 million information system. On Tuesday, you released a report on a surgical biopsy, indicating that it contained cancer. On Friday morning, you showed the same biopsy to your colleagues, who all agreed that the biopsy was not malignant and contained a benign condition that simulated malignancy (looked a little like a cancer, but was not). Your original diagnosis was wrong, and now you must rectify the error. You return to the computer and access the prior report, changing the wording of the diagnosis to indicate that the biopsy is benign. You can do this because pathologists are granted "edit" access for pathology reports. Now, everything seems to have been set right. The report has been corrected, and the final report in the computer is the official diagnosis.

Unknown to you, the patient's doctor read the incorrect report on Wednesday, the day after the incorrect report was issued, and 2 days before the correct report replaced the incorrect report. Major surgery was scheduled for the following Wednesday (5 days after the corrected report was issued). Most of the patient's liver was removed. No cancer was found in the excised liver. Eventually, the surgeon and patient learned that the original report had been altered. The patient sued the surgeon, the pathologist, and the hospital.

You, the pathologist, argued in court that the computer held one report issued by the pathologist (following the deletion of the earlier, incorrect report) and that report was correct. Therefore, you said, you made no error. The patient's lawyer had access to a medical chart in which paper versions of the diagnosis had been kept. The lawyer produced, for the edification of the jury, two reports from the same pathologist, on the same biopsy: one positive for cancer, the other benign. The hospital, conceding that they had no credible defense, settled out of court for a very large quantity of money. Meanwhile, back in the hospital, a fastidious intern is deleting an erroneous diagnosis and substituting his improved rendition.

One of the most important features of serious Big Data resources (such as the data collected in hospital information systems) is immutability (see Glossary item, Serious Big Data). The rule is simple. Data is immortal and cannot change. You can add data to the system, but you can never alter data and you can never erase data. Immutability is counterintuitive to most people, including most data analysts. If a patient has a glucose level of 100 on Monday, and the same patient has a glucose level of 115 on Tuesday, then it would seem obvious that his glucose level changed between Monday and Tuesday. Not so. Monday's glucose level remains at 100. Until the end of time, Monday's glucose level will always be 100. On Tuesday, another glucose level was added to the record for the patient. Nothing that existed prior to Tuesday was changed.

IMMUTABILITY AND IDENTIFIERS

People change and forget to tell each other. ***Lillian Hellman***

Immutability applies to identifiers. In a serious Big Data resource, data objects never change their identity (i.e., their identifier sequences). Individuals never change their names. A person might add a married name, but the married name does not change the maiden name. The addition of a married name might occur as follows:

18843056488 is_a	patient
18843056488 has_a	maiden_name
18843056488 has_a	married_name
9937564783 is_a	maiden_name
4401835284 is_a	married_name
18843056488 maiden_name	Karen Sally Smith
18843056488 married_name	Karen Sally Smythe

Here, we have a woman named Karen Sally Smith. She has a unique, immutable identifier, "18843056488." Her patient record has various metadata/data pairs associated with her unique identifier. Karen is a patient, Karen has a maiden name, and Karen has a married name. The metadata tags that describe the data that is associated with Karen include "maiden_name" and "married_name." These metadata tags are themselves data objects. Hence, they must be provided with unique, immutable identifiers. Though metadata tags are themselves unique data objects, each metadata tag can be applied to many other data objects. In the following example, the unique maiden_name and married_name tags are associated with two different patients.

9937564783 is_a	maiden_name
4401835284 is_a	married_name
18843056488 is_a	patient
18843056488 has_a	maiden_name
18843056488 has_a	married_name
18843056488 maiden_name	Karen Sally Smith
18843056488 married_name	Karen Sally Smythe
73994611839 is_a	patient
73994611839 has_a	maiden_name
73994611839 has_a	married_name
73994611839 maiden_name	Barbara Hay Wire
73994611839 married_name	Barbara Haywire

The point here is that patients may acquire any number of names over the course of their lives, but the Big Data resource must have a method for storing and describing each of those names and associating them with the same unique patient identifier. Everyone who uses a Big Data resource must be confident that all the data objects in the resource are unique, identified, and immutable.

By now, you should be comfortable with the problem confronted by the pathologist who changed his mind. Rather than simply replacing one report with another, the pathologist might have issued a modification report, indicating that the new report supersedes the earlier report. In this case, the information system does not destroy or replace the earlier report, but creates a companion report. As a further precaution, the information system might flag the early report with a link to the ensuant entry. Alternately, the information system might allow the pathologist to issue an addendum (i.e., add-on text) to the original report. The addendum could have clarified that the original diagnosis is incorrect, stating the final diagnosis is the diagnosis in the addendum. Another addendum might indicate that the staff involved in the patient's care was notified of the updated diagnosis. The parts of the report (including any addenda) could be dated and authenticated with the electronic signature of the pathologist. Not one byte in the original report is ever changed. Had these procedures been implemented, the unnecessary surgery, the harm inflicted on the patient, the lawsuit, and the settlement might have all been avoided.

The problem of updating diagnoses may seem like a problem that is specific for the health care industry. It is not. The content of Big Data resources is constantly changing; the trick is accommodating all changes by the addition of data, not by the deletion or modification of data. For example, suppose a resource uses an industry standard for catalog order numbers assigned to parts of an automobile. These 7-digit numbers are used whenever a part needs to be purchased. The resource may inventory millions of different parts, each with an order number annotation. What happens when the standard suddenly changes and all of the existing 7-digit numbers are replaced by 12-digit numbers? A well-managed resource will preserve all of the currently held information, including the metadata tags that describe the 7-digit standard, and the 7-digit order number for each part in the resource inventory. The new standard, containing 12-digit numbers, will have a different metadata tag from the prior standard, and the new metadata/data pair will be attached to the internal identifier for the part. This operation will work if the resource maintains its own unique identifiers for every data object held in the resource and if the data objects in the resource are associated with metadata/data pairs. All of these actions involve adding information to data objects, not deleting information.

In the days of small data, this was not much of a problem. The typical small data scenario would involve creating a set of data, all at once, followed soon thereafter by a sweeping analytic procedure applied against the set of data, culminating in a report that summarized the conclusions. If there was some problem with the study, a correction would be made, and everything would be repeated. A second analysis would be performed in the new and improved data set. It was all so simple.

A procedure for replicative annotations to accommodate the introduction of new standards and nomenclatures, as well as new versions of old standards and nomenclatures, would be one of the more onerous jobs of the Big Data curator (see Glossary item, Curator). Over the years, dozens of new or additional annotations could be required. It should be stressed that replicative annotations for nomenclatures and standards can be avoided if the data objects in the resource are not tied to any specific standard. If the data objects are well specified (i.e., providing adequate and uniform descriptions), queries can be matched against any standard or nomenclature on the fly (i.e., as needed, in response to queries). Such a method was discussed in Chapter 1.

Why is it always bad to change the data objects held in a Big Data resource? Though there are many possible negative repercussions to deleting data, most of the problems come down to data verification and time stamping (see Glossary items: Time stamp, Verification, and Validation). All Big Data resources must be able to verify that the data held in the resource conforms to a set of protocols for preparing data objects and measuring data values. The data in the resource is validated if it can be shown that the data fulfills its intended purposes, repeatably. When you change preexisting data, all of your efforts at resource verification are wasted because the resource that you once verified no longer exists. Aside from producing an unverifiable resource, you put the resource user into the untenable position of deciding which data to believe—the old data or the new data. Time stamping is another component of data objects. Events (e.g., a part purchased, a report issued, a file opened) have no meaning unless you know when they occurred. Time stamps must be unique and immutable for data objects. A single event cannot occur at two different times.

Big Data is forever. **Big Data managers must do what seems to be impossible; they must learn how to modify data without altering the original content.**

DATA OBJECTS

How do modern Big Data managers do the impossible? How do they modify data without altering the original values? This is accomplished with the miracle of data objects.

Small data deals with data values. Typically, the values are stored in a spreadsheet or a simple database in which a data item (the row in a spreadsheet) is assigned a list of values. The values of the spreadsheet or the database are numbers or character sequences; the value descriptors are the row and column designators. This cannot work with Big Data; the collections of data are too complex, too diverse, and everything is subject to change.

For Big Data, you need to think in terms of data objects, in which everything known about the object is encapsulated within the object. For example, a data object will contain one or more values, and each value will be annotated with descriptors (i.e., attached to metadata). The object will be assigned a unique identifier that distinguishes the object from every other object. The object will be assigned to one or more object classes of objects and will inherit the properties of each assigned class. Most importantly, the data object may contain assertions about the included data values, including addenda to the data (e.g., updates, transformations, mappings to nomenclatures, etc.) and the times that these additional values were created.

Introspection was reviewed in Chapter 4. Aside from providing a way to understand the organization and content held by a Big Data resource, introspection provides a way to fully understand each data object residing in the resource. With introspection, a user can review all of the data values held in the object, along with the times that these data objects were created. If the values were fully described with metadata, it would be possible to determine which values held in the data object are obsolete and which values have the preferred and current representation of the data.

In practical terms, how are data objects created and queried? There are many ways to implement data objects, but the easiest way to see the unifying principles behind data objects starts with an understanding of references, also known as pointers.

Consider the computer instruction:

$x = 5$

In most computer languages, this is equivalent to saying, in English, put the integer 5 into the variable x (i.e., assign 5 to x). For the computer, there is no x. The computer creates an address in memory to hold the integer 5. The variable x is a pointer to the address where the variable is stored. Every computation involves following a pointer to an address and accessing the data held within.

All object-oriented languages are based on using pointers to data. Once you have a pointer you can load all sorts of accessible information. Object-oriented languages associate complex objects with pointers and provide a variety of methods whereby the various data values associated with a data object can be updated, and all of the updates to the data object can be fully described and documented to produce a data object history.

In object-oriented languages, the methods available to a data object are determined by the class assignment for the object. The languages come with a library of class methods that can be applied to every data object that has been assigned to the class. The low-order mechanics of object methods (assigning pointers, finding pointers, accessing various parts of the data associated with specific pointers, performing basic operations on data values, etc.) are provided by the language and hidden from the programmer.[65]

Big Data resources need not be built as though they were object-oriented languages, but a few of the object-oriented tricks are worth preserving. Because it is impractical for a database to keep tabs on every modification on every data object, of every type, we create data objects that document and track their own content.

How these features of immutability will be implemented by Big Data resources is left to the data managers. In theory, all of these features can be simulated using a simple table, in which every row is a unique data object, and for which all of the necessary encapsulated data is splayed out along the length of the row. If the table were accessible to a program whose methods could search through each record, performing any and all operations needed to access the correct data values, then that would suffice. Likewise, a store of RDF triples, if well designed, could provide every requisite feature of data objects, as well as an opportunity for data object introspection (see Chapter 4). There is no reason to confine Big Data resources to any particular methodology.

LEGACY DATA

We need above all to know about changes; no one wants or needs to be reminded 16 hours a day that his shoes are on. ***David Hubel***

Every child believes, for a time, that the universe began with his own birth. Anything preceding his birth is unreal and of no consequence. Many Big Data resources have a similar disregard for events that preceded their birth. If events occurred prior to the creation of the Big Data resource, they have no consequence and can be safely ignored.

The problem here is that, for many domains, it is counterproductive to pretend that history starts with the creation of the Big Data resource. Take, for example, patents. Anyone seeking a patent must determine whether his claims are, in fact, novel, and this determination requires a search over existing patents. If the U.S. patent office decided one day to begin computerizing all applications, it would need to have some way of adding all of the patents dating back to 1790, when the first patent act of the U.S. Congress was passed. Some might argue that a good resource for U.S. patent data should include patents awarded within the original colonies; such patents date back to 1641.

Big Data creators often neglect legacy data—old data that may exist in obsolete formats, on obsolete media, without proper annotation, and collected under dubious circumstances. Old data often provides the only clues to time-dependent events. Furthermore, Big Data resources typically absorb smaller resources or merge into larger resources over time. If the added data is to have any practical value, then the managers of the resource must find a way to incorporate legacy data into the aggregated resource.

The health care industry is a prime example of Big Data in search of a legacy. President Barack Obama has set a goal for every American to have a secure medical record by 2014. If, in the year 2014, every American had a secure medical record, what might that record include? Let us consider the medical record for a hypothetical patient named Lily Livesy, age 92. None of her medical records, from birth to middle age, have survived the passage of time. Not only has Mary outlived her doctors; she has outlived most of her hospitals. Though she lived in one city all her life, several of the hospitals that administered her medical care have been closed, and the records destroyed. In the past 30 years, she has received medical care at various doctors' offices and in various departments in various hospitals. Some of these hospitals kept paper records; some had electronic records. Only one of the hospitals had anything that might be likened to an integrated hospital information system that aggregated transaction records produced by the various hospital departments (pharmacy, pathology, radiology, surgery, medicine, and so on). This hospital initiated a new electronic health record (EHR) system in the year 2013. Unfortunately, the new system is not compatible with the same hospital's prior information system, and the old records did not transfer to the new system. Consequently, in the year 2014, Lily Livesy, age 92, has one EHR, residing in one hospital's information system, with no contribution from any other medical facility, and this EHR contains a secure identifier, but no actual medical information. Her 92-year-long medical history is virtually blank.

Often, the utility of legacy data comes as an afterthought inspired by a preliminary analysis of contemporary data. If a cancer researcher notices that the incidence of a certain tumor is high, he or she would naturally want to know whether the incidence of the tumor has been increasing over the past 5 years, 10 years, 15 years, and so on. A forensic criminologist who collects a Combined DNA Index System (CODIS) signature on a sample of DNA might desperately need to check his sample against CODIS signatures collected over the past five decades. The most useful Big Data resources reach back through time.

History is replete with examples of old data driving new discoveries. A recent headline story explains how century-old tidal data plausibly explained the appearance of the iceberg that sank the Titanic on April 15, 1912.[66] Records show that several months earlier, in 1912, the moon, earth, and sun aligned to produce a strong tidal pull, and this happened when the moon was the closest to the earth in 1400 years. The resulting tidal surge was sufficient to break the January Labrador ice sheet, sending an unusual number of icebergs towards the open North Atlantic waters. The Labrador icebergs arrived in the commercial shipping lanes 4 months later, in time for a fateful rendezvous with the Titanic. Back in January 1912, when tidal measurements were being collected, nobody foresaw that the data would be examined a century later.

Legacy data plays a crucial role in correcting the current record. It is not unusual for people to rely on flawed data. If we knew the full history of the data, including how it was originally collected and how it was modified over time, we might avoid reaching erroneous conclusions. Several years ago, newspaper headlines drew attention to a modern manual for prisoner interrogation used by U.S. forces stationed in Guantanamo. It turned out that the manual was a republication of a discredited Chinese operations manual used during the Korean War. The chain of documentation linking the current manual back to the original source had been broken.[67] In another example of lost legacy data, a Supreme Court decision was based, in part, on flawed information; an earlier statute had been overlooked.[68] Had the legacy data been raised during deliberations, an alternate Supreme Court verdict may have prevailed. To know the history of a data source, we need access to the legacy data that documents the original sources of our data and permits us to trace the modifications of the data, over time.

DATA BORN FROM DATA

The chief problem in historical honesty isn't outright lying. It is omission or de-emphasis of important data. ***Howard Zinn***

Imagine this scenario. A data analyst extracts a large set of data from a Big Data resource. After subjecting the data to several cycles of the usual operations (data cleaning, data reduction, data filtering, data transformation, and the creation of customized data metrics), the data analyst is left with a new set of data, derived from the original set. The data analyst has imbued this new set of data with some added value, not apparent in the original set of data.

The question is, "How does the data analyst insert his new set of derived data back into the original Big Data resource, without violating immutability?" The answer is simplicity itself—reinserting the derived data is impossible and should not be attempted. The transformed data set is not a collection of original measurements; it cannot be sensibly verified by the data manager of the Big Data Resource. The modifications guarantee that the data values will not fit into the data object model upon which the resource was created (see Glossary item, Data object model). There simply is no substitute for the original and primary data.

The data analyst should make her methods and her transformed data available for review by others. Every step involved in creating the new data set needs to be carefully recorded and explained, but the transformed set of data cannot be absorbed into the data model for the resource. The original Big Data resource from which the raw data was extracted can include a document containing the details of the analysis and the modified data set produced by the analyst. Alternately, the Big Data resource may provide a link to sources holding the modified data sets. These steps provide the public with an information trail leading from the original data to the transformed data prepared by the data analyst.

RECONCILING IDENTIFIERS ACROSS INSTITUTIONS

In many cases, the biggest obstacle to achieving Big Data immutability is data record reconciliation. When different institutions merge their data systems, it is crucial that no data be lost and all identifiers be sensibly preserved. Cross-institutional identifier reconciliation is the process whereby institutions determine which data objects, held in different resources, are identical (i.e., the same data object). The data held in reconciled identical data objects can be combined in search results, and the identical data objects themselves can be merged (i.e., all of the encapsulated data can be combined in one data object) when Big Data resources are combined or when legacy data is absorbed into a Big data resource.

In the absence of successful reconciliation, there is no way to determine the unique identity of records (i.e., duplicate data objects may exist across institutions) and data users will be unable to rationally analyze data that relates to or is dependent upon the distinctions among objects in a data set. For all practical purposes, without data object reconciliation, there is no way to understand data received from multiple sources.

Reconciliation is particularly important for health care agencies. Some countries provide citizens with a personal medical identifier that is used in every medical facility in the nation. Hospital A can send a query to Hospital B for medical records pertaining to a patient sitting in Hospital A's emergency room. The national patient identifier ensures that the cross-institutional query will yield all of Hospital B's data on the patient and will not include data on other patients.

In the United States, interhospital record reconciliations are all but impossible. The United States does not have a national patient identifier. Though the Health Insurance Portability and Accountability Act of 1996 mandated a unique individual identifier for health care purposes, the U.S. Congress has since opposed this provision. Specifically, the 1998 Omnibus Appropriations Act (Public Law 105-277) blocked funding for a final standard on unique health identifiers. It seems unlikely that a national patient identifier system will be deployed in the United States anytime soon. Reconciliation is needed whenever a patient needs to collect records from different institutions, when an institution is acquired by another institution, or when the legacy data from an abandoned information system is merged with the data in a newly deployed information system. It is obvious that reconciliation is one of the most vexing problems in the field of medical informatics.

Consider the common problem of two institutions trying to reconcile personal records (e.g., banking records, medical charts, dating service records, credit card information). When both institutions are using the same identifiers for individuals in their resources, then reconciliation is effortless. Searches on an identifier will retrieve all the information attached to the identifier, if the search query is granted access to the information systems in both institutions. However, multi-institutional or universal identifier systems are rare. If either of the institutions lacks an adequate identifier system, the data from the systems cannot be sensibly reconciled. Data pertaining to a single individual may be unattached to any identifier, attached to one or more of several different identifiers, or mixed into the records of other individuals. The merging process would fail, at this point.

Assuming both institutions have adequate identifiers, then the two institutions must devise a method whereby a new identifier is created, for each record, that will be identical to the new identifier created for the same individual's record, in the other institution. For example, suppose each institution happens to store biometric data (e.g., retinal scans, DNA sequences, fingerprints); then the institutions might agree on a way to create a new identifier validated against these unique markers. With some testing, they could determine whether the new identifier works as specified (i.e., either institution will always create the same identifier for the same individual, and the identifier will never apply to any other individual). Once testing is finished, the new identifiers can be used for cross-institutional searches.

Lacking a unique biometric for individuals, reconciliation between institutions is feasible, but difficult. Some combination of identifiers (e.g., date of birth, social security number, name) might be developed. Producing an identifier from a combination of imperfect attributes has its limitations (as discussed in detail in Chapter 2), but it has the advantage that if all the preconditions of the identifier are met, misidentification will be uncommon (i.e., two records with the same identifier will belong to the same person). In this case, both institutions will need to decide how they will handle the set of records for which there is no identifier match in the other institution. They may assume that some individuals will have records in both institutions, but their records were not successfully reconciled by the new identifier. They may also assume that the unmatched group contains individuals that actually have no records in the other institution. Dealing with unreconciled records is a nasty problem. In most cases, it requires a curator to slog through individual records, using additional data from records or new data supplied by individuals, to make adjustments, as needed.

ZERO-KNOWLEDGE RECONCILIATION

Though record reconciliation across institutions is always difficult, the task becomes truly Herculean when it must be done blindly, without directly comparing records. This awkward situation occurs quite commonly whenever confidential data records from different institutions must be checked to see if they belong to the same person. In this case, neither institution is permitted to learn anything about the contents of records in the other institution. Reconciliation, if it is to occur, must implement a zero-knowledge protocol—a protocol that does not convey knowledge in or about records.

In a prior publication, I proposed a zero-knowledge protocol for reconciling patient records across institutions.[69] Because the protocol itself is somewhat abstract and unintuitive, a physical analogy may clarify the methodology. Imagine two people each holding a box containing an item. Neither person knows the contents of the box that they are holding or of the box that the other person is holding. They want to determine whether they are holding identical items, but they don't want to know anything about the items. They work together to create two identically prepared imprint stamps. With eyes closed, each one pushes his deformable stamp against his item. By doing so, the stamp is imprinted with markings characteristic of the object's surface. The stamps are next examined to determine if the compression marks on the ridges are distributed identically in both stamps. If so, the items in the two boxes, whatever they may be, are considered to be identical. Not all of the markings need to be compared—just enough of them to reach a high level of certainty. It is theoretically possible for two different items to produce the same pattern of compression marks, but it is highly unlikely. After the comparison is made, the stamps are discarded.

This physical analogy demonstrates the power of a zero-knowledge protocol. Neither party knows the identity of his own item. Neither party learns anything about his item or the other party's item during the transaction. Yet, somehow, the parties can determine whether the two items are identical.

Here is how the zero-knowledge protocol is able to reconcile confidential records across institutions.

1. Both institutions generate a random number of a predetermined length, and each institution sends the random number to the other institution.
2. Each institution sums its own random number with the random number provided by the other institution. We will refer to this number as Random_A. In this way, both institutions have the same final random number, and neither institution has actually transmitted this final random number. The splitting of the random number was arranged as a security precaution.
3. Both institutions agree to create a composite representation of information contained in the record that could establish the human subject of the record. The composite might be a concatenation of the social security number, the date of birth, the first initial of the surname.
4. Both institutions create a program that automatically creates the composite representation for the record and immediately sums the signature with Random_A, the random number that was negotiated between the two institutions (steps 1 and 2). The sum of the composite representation of the record plus Random_A is a random number that we will call Random_B.

5. If the two records being compared across institutions belong to the same human subject, then Random_B will be identical in both institutions. At this point, the two institutions must compare their respective versions of Random_B in such a way that they do not actually transmit Random_B to the other institution. If they were to transmit Random_B to the other institution, then the receiving institution could subtract Random_A from Random_B and produce the signature string for a confidential record contained in the other institution. This would be a violation of the requirement to share zero knowledge during the transaction.
6. The institutions take turns sending consecutive characters of their versions of Random_B. For example, the first institution sends the first character to the second institution. The second institution sends the second character to the first institution. The first institution sends the third character to the second institution. The exchange of characters proceeds until the first discrepancy occurs or until the first eight characters of the string match successfully. If any of the characters do not match, both institutions can assume that the records belong to different human subjects (i.e., reconciliation failed). If the first eight characters match, then it is assumed that both institutions are holding the same Random_B string and that the records are reconciled.

Data record reconciliation is seldom impossible. In the somewhat extreme protocol described above, reconciliation occurs with neither institution providing the other institution with any of the information contained in confidential data records. As you might expect, a practical implementation of the protocol might carry a retinue of complexities, as discussed in the original article.[69] The point here is that it is possible to reconcile records from different institutions, from legacy collections, and from merged data sets within an institution, if you set your mind to the task.

THE CURATOR'S BURDEN

Who is responsible for all this immutability that haunts every Big Data resource? Most of the burden falls upon the data curator. The word "curator" derives from the Latin "curatus," the same root for "curative," and conveys that curators "take care of" things. In a Big Data resource, the curator must oversee the accrual of sufficiently annotated legacy and prospective data into the resource; must choose appropriate nomenclatures for annotating the data; must annotate the data; and must supplement records with updated annotations as appropriate, when new versions of nomenclatures, specifications, and standards are applied.

The curator is saddled with the almost impossible task of keeping current the annotative scaffold of the Big Data resource, without actually changing any of the existing content of records. In the absence of curation, all resources degenerate over time. We will return to the important tasks of the modern curator in Chapter 15.

Intentionally left as blank