Solving the Data Deluge Problem

Jens Pohl, Ph.D.

Executive Director, Collaborative Agent Design Research Center (CADRC)
California Polytechnic State University (Cal Poly)
San Luis Obispo, California, USA

Abstract

The paper postulates that the information technology revolution that is commonly referred to as the Information Age is currently in a transition stage between data-processing and knowledge management that should be more aptly referred to as the Data Age. Symptoms of this transition stage are a data deluge problem that is evidenced by the inability of human computer-users to effectively analyze and draw useful conclusions from the overwhelming volume of data that is being collected, the increasing complexity of networked systems, and the acknowledged vulnerability of virtually all existing digital systems to cyber security threats.

The author suggests that the core cause of the data deluge problem is that existing computer software systems are largely confined to the processing of atomic data elements rather than meaningful information. With the incorporation of a virtual model of the relevant real world knowledge domain it is possible for computer software to interpret the meaning of data within the context provided by the model. Such models can be constructed in the form of an ontology that is machine processable and accessible to inferencing modules referred to as agents. Context-based information-centric software provides a level of artificial intelligence that can be effectively used to mitigate the current data bottleneck, to shield the human user from the technical complexities of distributed systems, and to maintain an acceptable level of cyber security.

Keywords

Agents, autonomic computing, context, cyber security, data, Data Age, data-centric, information, Information Age, information assurance, information-centric, intelligence, networks, ontology, representation, security,

1. Introduction

When we use search engines to find some information on the Internet, we typically receive more links to potential information sources (i.e., hits) than we care to look at. We have learned from experience that many of the hits will be disappointing because they do not lead to the information that we are seeking. Soon after the terrorist attacks on the United States in September 2001, much evidence was found that several warnings of a planned attack were contained in the routinely collected intelligence data, but had been overlooked. The military are so inundated with sensor data from satellites, unmanned aerial vehicles, and land-based sources that they cannot possibly analyze in near-real time. These are all symptoms of a rapidly escalating data deluge problem.

The amount of data that is being collected by our global digital infrastructure far exceeds our human ability to interpret, analyze, draw conclusions, and act upon under even less than time-critical circumstances. This is not a problem that occurs only under special circumstances, such as the relief operations after a major national disaster. Rather, the data deluge problem is clearly becoming more pronounced. The reason is quite simple; - while the volume of data is increasing exponentially our human ability to interpret the data is increasing linearly. Clearly, if we continue to apply the same methods to this problem then we are destined to fall further and further behind.

So, what is at the core of this problem? Is it that we are collecting more data than we need? Or, perhaps, faster computers and better collection methods will eventually overcome the problem. No, we must not limit our data collection capabilities and even faster computers are not going to solve the problem because the volume of data is increasing at an even faster rate. Instead we must find a way of automating the interpretation and analysis of data. Although we have proclaimed for some time to have entered the Information Age we are still largely immersed in what might be more appropriately referred to as the Data Age.

2. Difference between Data and Information

It is a common misconception that with the entry into the Information Age we are overwhelmed by an overabundance of information. A more accurate characterization would be that while we are being subjected to a deluge of data we are in fact typically faced with a scarcity of information. There is a major difference between data and information. Data are simply numbers and words such as *rain*, *traffic*, *4*, *mph*, *car*, *police*, *intersection*, *184*, *Grand Junction*, *100*, *bridge*, *crossing*, and so on (Figure 1). These are all symbols that we understand and are able to reason about when they appear in some *context* such as the following sentence: "*… police car 64 crossing Grand Junction bridge at 100 mph.*"

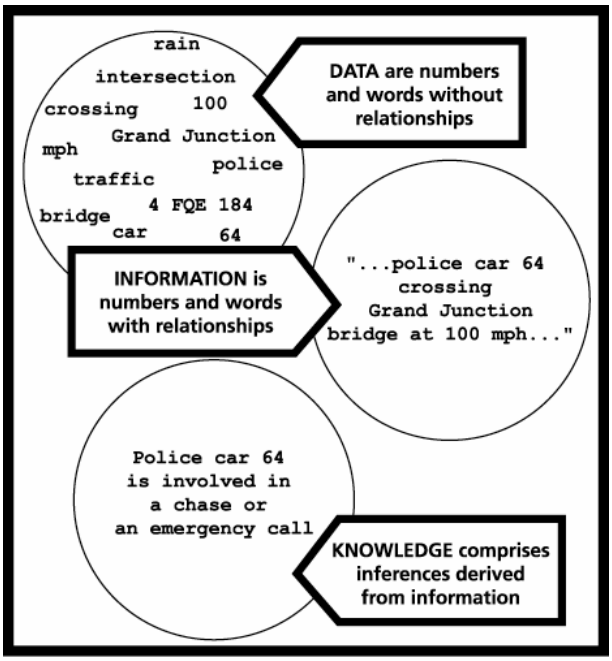

Figure 1: Definition of terms

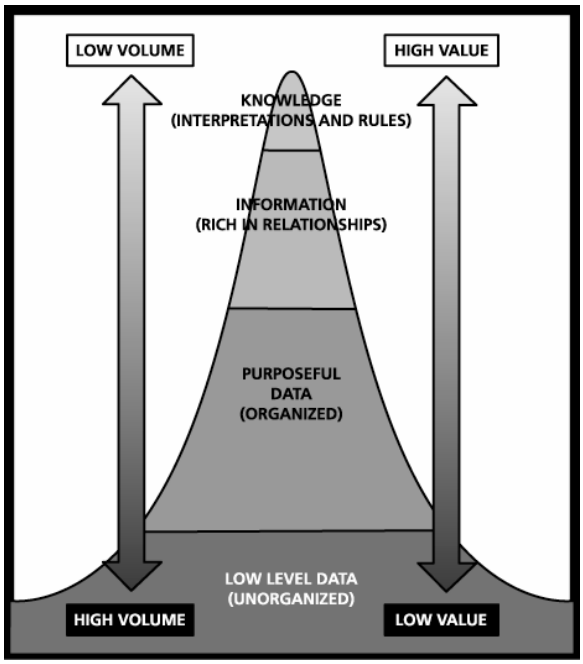

Figure 2: Transition from data to knowledge

It is not difficult for us to infer from the *context* of this sentence that if a *police car* is traveling at *100 mph* over a *bridge* in what is presumably a populated *Grand Junction* neighborhood then it is likely to be involved in a chase or emergency call. We are able to interpret data into information by utilizing the *context* that we have accumulated in our brain over time (i.e., our experience). At the lowest structural level this *context* consists of the implied relationships that bind the words and numbers together into a meaningful piece of information. For example, in our brain the symbol *bridge* is associated with characteristics that relate to the purpose of a bridge and how such a purpose can be achieved to the best of our engineering knowledge, and how this engineering problem has been solved in the past based on our life experience. Based on structural engineering principles a bridge has to span over some horizontal distance and is therefore likely to be limited in width due to structural constraints and cost. This relates well to our past experience of driving cautiously over bridges that are typically narrow, elevated, and subject to a speed limit.

Computers were invented in the 1940s because there was an urgent need for a device that could compute numbers much faster than a human mathematician. Applications that required this superhuman computational speed were principally related to transportation (i.e., navigational charts) and warfare (i.e., artillery tables). Much later in the 1970s it became apparent that the ability of computers to process large volumes of data was a very useful and potentially even more important capability. With the miniaturization of electronic components both the power and data storage capacity of computers has increased over the past two decades by factors of six and four every three years, respectively. Today (2010) an electronic mass storage device (i.e., disk drive) with a storage capacity of two terabytes (i.e., two thousand billion bytes) can be purchased for less than $250 at a local retail store. The enormity of that data storage capability becomes clear when we consider that the millions of books and documents in the Library of Congress collection will require a data storage capacity of less than 12 terabytes.

The principal reason for storing data is for analysis, for monitoring trends, and for planning purposes. However, these data-processing tasks must be performed in consideration of the *context* in which the data have been generated. While the quantity of data stored in computers was still relatively small these tasks could be performed by the human users who provided the necessary *context* by drawing on the experience and knowledge stored in their brain. Today, the volume of data that is stored in computers far outstrips the ability of the human user to interpret, analyze, and detect any but the most obvious trends.

3. The Need for Software Intelligence

Could the computer become an extension of the *context*-based data interpretation and analysis capabilities of the human user? This would require some representation of *context* to be embedded in the software components of a computer-based environment. Data in *context* is analogous to information, which by definition represents meaning. Therefore, software that is capable of processing information, commonly referred to as information-centric software to distinguish it from data-centric software, has by implication some degree of *understanding* of the data that it is designed to process. With even a relatively limited capacity to *understand* the meaning of data it becomes possible to incorporate in the software automated data interpretation, analysis and trend detection capabilities that may be characterized as *intelligent* features.

There are at least three compelling reasons why software that is enabled with the ability to process data within the *context* that the data are relevant is an essential prerequisites for exploiting the opportunities and combating the security risks posed by the Information Age.

Reason (1) – Increasing Data Volume: The first reason relates to the current data-processing bottleneck. As mentioned previously, advancements in computer hardware technology over the past several decades have made it possible to store vast amounts of data in electronic form. Based on past manual information handling practices and implicit acceptance of the principle that the interpretation of data into information and knowledge is the responsibility of the human operators of the computer-based data storage devices, emphasis was placed on storage efficiency rather than processing effectiveness. Typically, data processing methodologies focused on the storage, retrieval and manipulation of data transactions, rather than the *context* within which the collected data would later become useful in planning, monitoring, assessment, and decision-making tasks.

The larger an organization the more data it generates itself and captures from external sources. With the availability of powerful computer hardware and database management systems the ability of organizations to store and order these data in some purposeful manner has dramatically increased. However, at the same time, the expectations and need to utilize the stored data in monitoring, planning and time-critical decision-making tasks has become a major human resource intensive preoccupation. In many respects this data-centric focus has become a bottleneck that inhibits the ability of the organization to efficiently and effectively accomplish its mission.

The reasons for this bottleneck are twofold. First, large organizations are forced to focus their attention and efforts on the almost overwhelming tasks involved in converting unordered data into purposefully ordered data (Figure 2). This involves, in particular, the establishment of gateways to a large number of heterogeneous data sources, the validation and integration of these sources, the standardization of nomenclatures, and the collection of data elements into logical data models. Second, with the almost exclusive emphasis on the slicing and dicing of data, rather than the capture and preservation of relationships, the interpretation of the massive and continuously increasing volume of data is left to the users of the data. The experience and knowledge stored in the human cognitive system serves as the necessary *context* for the interpretation and utilization of the ordered data in monitoring, planning and decision-making processes. However, the burden imposed on the human user of having to interpret large amounts of data at the lowest levels of *context* has resulted in a wasteful and often ineffective application of valuable and scarce human resources. In particular, it often leads to late or non-recognition of patterns, overlooked consequences, missed opportunities, incomplete and inaccurate assessments, inability to respond in a timely manner, marginal decisions, and unnecessary human burn-out. These are symptoms of an incomplete information management environment. An environment that relies entirely on the capture of data and the ability of its human users to add the relationships to convert the data into information and thereby provide the *context* that is required for all effective planning and decision-making endeavors.

Reason (2) – Increasing Network Complexity: The second reason is somewhat different in nature. It relates to the complexity of networked computer and communication systems, and the increased reliance of organizations on the reliability and security of such information technology environments as the key enabler of their effectiveness, profitability and continued existence. The economic impact on an organization that is required to manually coordinate and maintain

hundreds of interfaces between data-processing systems and applications that have no understanding of the data that they are required to exchange is enormous. Ensuing costs are not only related to the requirement for human resources and technical maintenance, but also to the indirect consequences of an information systems environment that has hundreds of potential failure points.

Recent studies conducted by IBM Corporation and others have highlighted the need for autonomic computing as the organizational expectations and dependence on information services leads to more and more complex networked computer solutions (Ganek and Corbi 2003). In the commercial sector "…it is now estimated that at least one-third of an organization's IT (Information Technology) budget is spent on preventing or recovering from crashes" (Patterson et al. 2002). Simply stated, autonomic computing utilizes the understanding that can be represented within an information-centric software environment to allow systems to automatically: (1) reconfigure themselves under dynamically changing conditions; (2) discover, diagnose, and react to disruptions; (3) maximize resource utilization to meet end-user needs and system loads; and, (4) anticipate, detect, identify, and protect themselves from external and internal attacks.

Clearly, the increased reliance on computer-based information systems mandates a level of reliability and security that cannot be achieved through manual means alone. The alternative, an autonomic computing capability, requires the software that controls the operation of the system to have some understanding of system components and their interaction. In other words, autonomic computing software demands a similar internal information-centric representation of *context* that is required in support of the knowledge management activities in an organization. In both cases the availability of data in context is a prerequisite for the reasoning capabilities of the software (i.e., the automatic interpretation of information by the computer).

Reason (3) – Increasing Cyber Security Vulnerability: The third reason is related to cyber security and, in particular the prevention of unauthorized network intrusions and the protection of the data that are transmitted and stored within networks. Traditionally, the security of computer-based systems has relied on the physical separation of classified and non-classified systems and data encryption. Certainly, physical separation is neither desirable nor viable in a society that increasingly demands and depends on near-instant global connectivity. Yet, at the same time, we must ensure that the cross-domain transfer of data will be accomplished transparently, seamlessly, and securely.

Under these circumstances it can be assumed that information assurance must increasingly rely on measures that focus on securing the data within the network, on the assumption that the network can be penetrated by a skillful intruder (CBS 2009, CNN 2009). This should not at all imply that network access protection is considered to be ineffective and therefore unimportant. However, past experience has shown that even our apparently most secure military, intelligence, and commercial networks can be and have been penetrated. It is unlikely that implementation of the most sophisticated multi-biometric authentication measures cannot be *spoofed* (i.e., fooled through impersonation). Therefore, protection of the data that reside in our networks represents the final and most critical level of cyber defense.

Clearly, absolute information assurance cannot be achieved. There will always be someone who will devise an ingenious method for circumventing whatever security measures have been implemented. This may involve the development of new technology, such as the Colossus

machine that successfully deciphered the German Enigma code during the Second World War, or an isolated human lapse that can jeopardize the most stringently controlled security precautions. The best that we can aim for is an *acceptable* level of security that includes multiple gateways and, in particular, can detect network intrusion and compromised data at the earliest instance.

While the encryption of data is an essential security precaution, it is by itself not sufficient to provide an *acceptable* level of security. There is a need for software that is able to categorize that data that are processed and stored within a network into degrees of sensitivity and apply multiple levels of security depending on the nature of the data. Technology that is able to identify and extract secret data elements from a data stream, encrypt them, and store them in a dispersed manner across multiple networks, is already available even though it may not yet be employed in our networks. At the very least such data security software should be immediately evaluated with a view to accelerated implementation.

However, even this level of security will not provide an *acceptable* level of information assurance in the near future, nor does it exploit the full capabilities of our current IT knowledge. There is an urgent need to apply information-centric concepts and software design principles to protect the data in our networks. In this respect information-centric refers to the implementation of data security methodologies that are based on an understanding of the content (i.e., meaning) of data and the *context* within which the data are intended to be used, or could be misused by a rogue party. The ability to apply information assurance technologies that are capable of automatically imposing multi-level data security measures based on the ability of software to have some understanding of the meaning and *context* of a data stream is in the opinion of this author a fundamental prerequisite for the achievement of an *acceptable* level of information assurance.

4. Information-Centric Software: *An Urgent Need and Unique Opportunity*

Few will argue that computer-based systems will become increasingly more intelligent in the near future. The signs of this have already become apparent in fraud detection software employed on a daily basis by the insurance, banking and communication industries, by the adaptive decision-support systems used for military planning and re-planning, and by the findings that are being published in the research community. The ability of computer software to automatically extract meaning from unstructured text through the automated interpretation of data within the applicable *context* represents a paradigm shift in human endeavors. It will become a principal differentiator between competitiveness and non-competitiveness in the global marketplace and between *acceptable* and unacceptable levels of information assurance in the homeland security and national defense arenas. Our Government has an urgent need to accelerate this paradigm shift through leadership and, at the same time, take advantage of a unique opportunity to decisively enable our industrial complex in an increasingly competitive world-wide marketplace.

So far the US Government appears to have taken a decidedly reactive stance in respect to both cyber security and advances in information management. It has relied largely on industry to lead the way and seen neither the necessity nor the opportunity of implementing a concerted effort of appropriate proportions to ensure that the US will regain the preeminent position that it held in IT some 20 years ago. Over the past two decades our leadership has eroded to the point where

our networks are subjected to daily intrusion (much of it probably unknown to us) and our most secret data assets are not secure.

5. Context Representation and Intelligent Tools

The ability to represent *context* in computer software has been available for at least the past 30 years (Winston 1970, Biermann and Feldman 1972, Cohen and Sammut 1978). Hampered initially by a lack of hardware power and later by the absence of any compelling need to involve the computer in the direct interpretation of data, these information modeling techniques were not applied in the mainstream of computer software development until fairly recently. The compelling reasons that have suddenly brought them to the foreground are the increasing volume of computer-based data that is beginning to overwhelm human users, and the homeland security concerns that emerged after the tragic September 11, 2001 terrorist incidents in the United States.

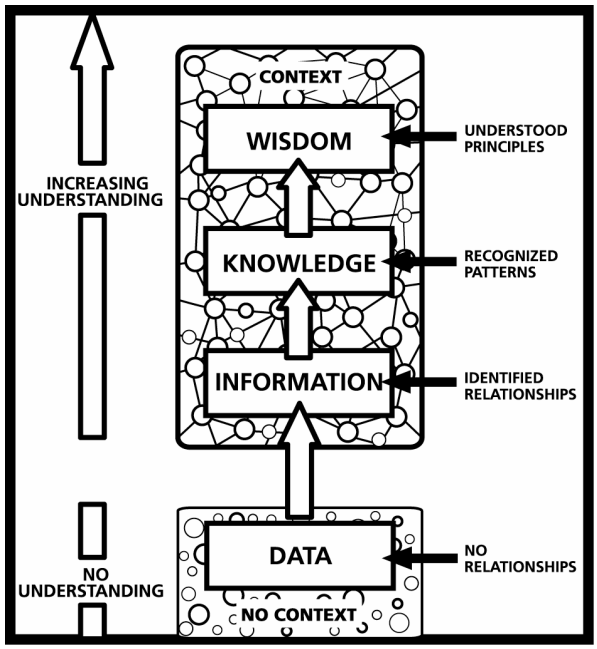

Figure 3: The paradigm shift from data to information with relationships

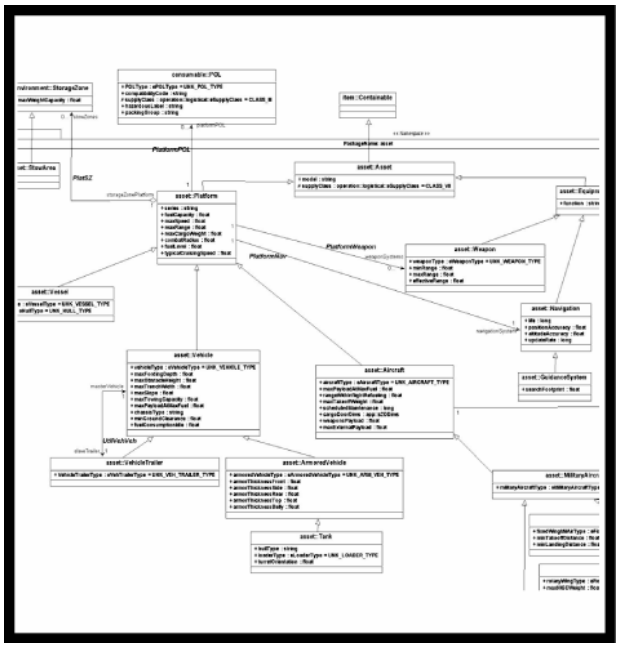

Figure 4: Portion of a typical information model (ontology) in the logistic domain

The physical gap that is shown schematically between the realms of the data environment without context and no understanding, and the information environment with context and ascending levels of greater understanding in Figure 3, underscores the fundamental difference between the two realms. The transition from data-processing software to information-centric software requires a paradigm shift in the human perception of the role of computers. By incorporating an internal information model (i.e., ontology) that represents portions of real world context as a virtual environment of objects their characteristics and the associations that relate these objects, information-centric software is capable of performing a useful level of automatic reasoning. A number of software agents with relatively simple reasoning capabilities are able to collaborate and through their collective efforts come to more sophisticated conclusions.

For the computer to be able to support automatic reasoning capabilities we have to create a software environment that incorporates context. This can be achieved fairly easily by

constructing an information model as a virtual representation of the real world context within which software agents are expected to apply their reasoning capabilities. Such an internal information model is referred to as an ontology. A small part of a typical example of such an ontology is shown in Figure 4. It describes the real world context in terms of objects with characteristics and relationships. For example, in a military command and control context such objects would include different kinds of weapons, a wide range of infrastructure objects, weather forecasts, friendly and enemy units, and even conceptual objects such as the notions of threat, planning, mobility, and readiness. Generally speaking, the more relationships among objects that are included in the ontology the more context is provided by the ontology, and the more powerful (i.e., intelligent) the reasoning capabilities of the software agents are likely to be.

Without the context provided by an internal information model (i.e., ontology) there can be no meaningful, automatic reasoning by software agents. Therefore, the essential prerequisite for intelligent agents is the existence of an internal information model that provides the necessary context for the symbolic reasoning activities of the agents. We human beings do not have to consciously invoke any action to relate what we see, hear and feel to the context held in our brain. The need for this context to be created in the computer is therefore not intuitively obvious to us. This is no doubt the principal reason why such a fundamental aspect of intelligent computer-based agents is still largely overlooked.

Software Agents – Automated Reasoning: There are several types of software agents, ranging from those that emulate symbolic reasoning by processing rules, to highly mathematical pattern matching neural networks, genetic algorithms, and particle swarm optimization techniques. The focus here is on ontology-based software that utilizes agents with symbolic reasoning capabilities. These agents may be described as software modules that are capable of reasoning about events (i.e., changes in data received from external sources or as the result of internal activities) within the context of the information contained in the internal information model (i.e., ontology). Since such agents are also often loosely referred to as intelligent agents the question arises whether computer intelligence is really possible? From a commonsense point of view it would appear that humans have intelligence and computers are just very fast but unintelligent machines. Looking at this question from an entirely human point of view we may well come to such a conclusion.

Human intelligence is only one kind of intelligence that is strongly influenced by the biochemical nature of the human body and its cognitive facilities, while the artificial intelligence that can be embedded in computer software is another kind of intelligence altogether. To differentiate human intelligence from the seemingly intelligent capabilities of computers we need to entertain the notion that there are levels of intelligent behavior. Remembering is probably the lowest level of intelligence. Certainly computers can store vast amounts of data and can retrieve these data quickly and accurately. However, from our human point of view remembering is more than just retrieving data. Remembering also involves relationships and context, which makes data meaningful and relevant. Therefore, by including an information model (i.e., ontology) in a software application or service we are able to represent information rather than data in the computer. This allows us to include modules in the software (i.e., software agents) that are able to automatically reason and communicate the results of their reasoning activities to other agents, including human users. We are creating in this way a virtual copy of the context of a problem situation in the computer-based environment. The players (i.e., the agents) in this virtual environment can assume many different roles and can contribute and collaborate at many levels,

ranging from the categorization of data to the identification of patterns and the recognition of relationships that may have been overlooked by human users.

In this way, if we store not only data in the computer but also the relationships that convert such numbers and words into information then we can also embed in the software rule sequences that are capable of reasoning about this information. Such sequences may be as simple as condition-action statements. For example, *if* an enemy tank unit is sighted *then* place a call-for-fire on the enemy tank unit *and* commence the process of weapon selection. However, the same software might also incorporate agents that perform more sophisticated tasks. For example, selecting the best mix of lift assets (e.g., helicopters, hovercraft, vertical take-off aircraft, etc.) to transport a wide range of supplies to multiple landing zones within requested time windows, and within constraints such as weather conditions, enemy actions, and so on. The latter agents consider results received from other agents, and utilize a wide range of heuristic and algorithmic methods to arrive at a possible solution.

Ontology – Context Representation: How can we embed in software a virtual version of a real world context using an ontology? Let us assume that we wish to represent a component of a building such as a conference room in the computer. Until recently, in a *data-centric* software environment, we would have treated the conference room as a three-dimensional geometric entity that can be described in terms of points (i.e., x-y-z coordinates), lines, or surfaces. While this may be satisfactory for displaying different internal views of the building space and even generating animated walk-through sequences, it does not provide a basis for the computer to reason about any aspect of the space, such as that a conference room must have a door for it to be usable. To provide the computer with such a reasoning capability the particular entity, in this case the conference room, must be represented in the computer as an information structure that describes conference room as an integral component of a building. This can be achieved by storing in the computer the word building and associating this word with some characteristics such as: a building is a physical object; it is made of material; it has height, width and length; consists of one or more floors; has spaces on floors; and so on. Then further defining spaces with characteristics such as: enclosed by walls, floor and ceiling; with walls having at least one opening referred to as a door; and so on.

In such an *information-centric* software environment the same conference room would be stored in the computer as part of the building ontology, allowing the following more intelligent user-computer collaboration:

Computer user: I would like to represent a component of a *building*.

Computer software: Loads its stored *building* ontology into memory.
Asks user "What kind of a *building* component?"

Computer user: A *space* of type *conference room*.

Computer software: For how many persons?

Computer user: Up to 16 persons.

Computer software: Suggested space size is: 16 ft (length), 14 ft (width), 8 ft (height).
Suggested furniture: 6 ft by 3 ft table, 16 chairs, screen, white board.
Other features: There must be at least one door.

As can be seen from this user-computer interaction, the computer software by virtue of its internal information structure has some understanding of the meaning of a *building* within the context of its characteristics and the relationships of its components (i.e., floors, spaces, walls, openings, and furniture). This endows the computer software with the ability to collaborate and assist the user by reasoning about the relationships between the data entered by the user and the context contained in the relatively simple information representation provided by the *building* ontology.

6. Connecting the Dots

The potential impact of this kind of computer-based reasoning capability on current data-based intelligence collection and analysis practices is profound, to say the least. For example, let us assume that during a typical four-week period the following four messages (Figure 5) have appeared among the hundreds of thousands of data items that were collected by the various intelligence agencies: (message A) *terrorist warning from Egyptian counterintelligence agency*; (message B) *suspected Middle-East terrorist apprehended on entry into US*; (message C) *meeting between suspected terrorist operatives at O'Hare Airport in Chicago*; and, (message D) *explosives theft in Chicago*. Although the messages come from different sources and are collected by different intelligence agencies, they are automatically time-stamped and fed into a message network (Figure 6) that is presumably accessible to all such agencies.

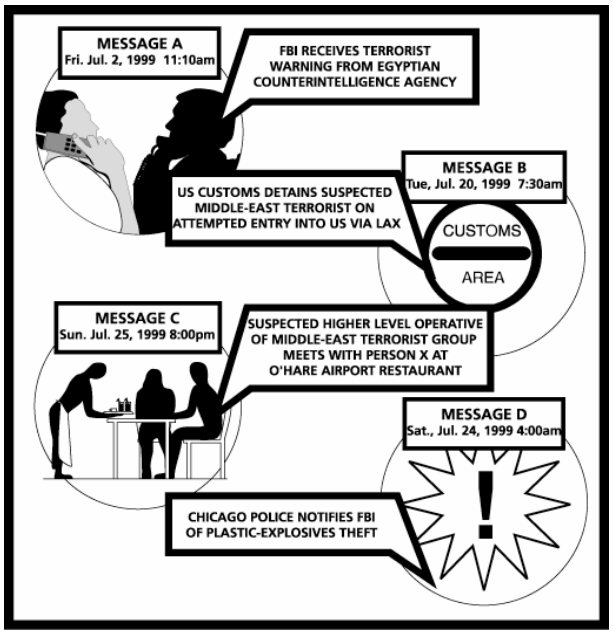

Figure 5: Four typical intelligence messages (***data-centric*** software environment)

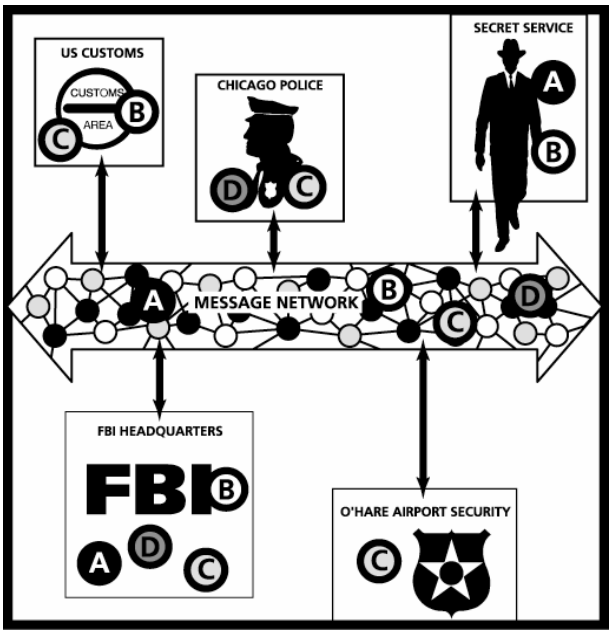

Figure 6: Manual processing of messages (***data-centric*** software environment)

There are several serious problems with this method of collecting and subsequently analyzing intelligence data in the current *data-centric* software environment. First, virtually all of these raw data items have to be analyzed and evaluated by human operators. The absence of relationships (to provide context) prevents any really useful automatic filtering to be implemented. Only limited data-processing methods such as keyword searches and indexing, can be employed. This

places an enormous burden on limited human resources. As a result valuable human intelligence personnel are literally burned out at the lowest level of intelligence analysis, leading to a tendency for the data collection activity to be restricted.

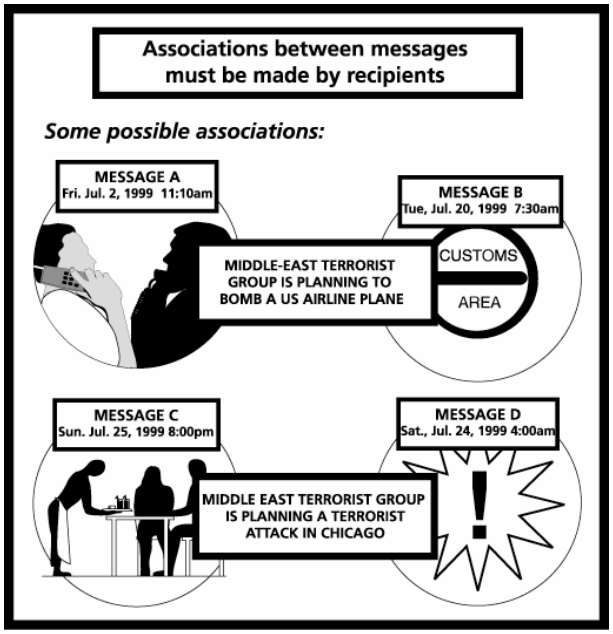

Figure 7: Manual processing of associations (***data-centric*** software environment)

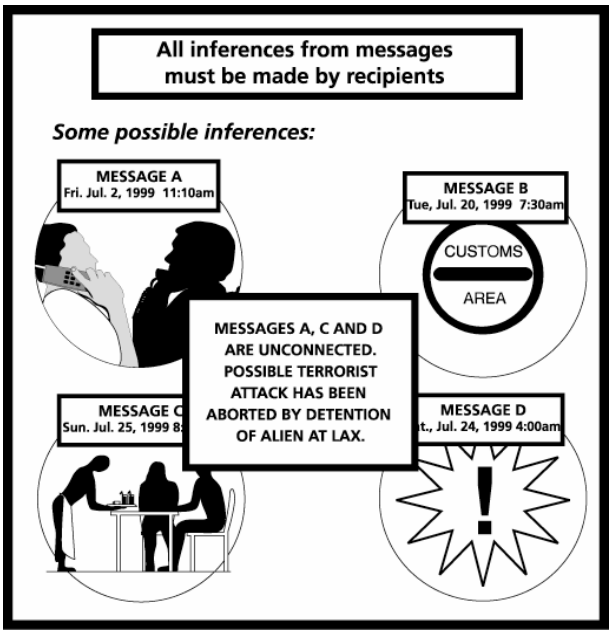

Figure 8: Manual processing of inferences (***data-centric*** software environment)

Second, there is no guarantee that all agencies will receive all four of the data items. Since any associations between these and other messages have to be made by human analysts (Figures 7 and 8) there is a good chance that some of the possible associations will be either overlooked, or not pursued for lack of available manpower. Yet these associations are critical for the transformation of data into information and the detection of patterns through appropriate inferences. Even if the available human resources allow first level associations to be established, it is unlikely that many second and higher level associations will be developed due to human labor limitations and time constraints.

Nowhere are the shortcomings of a data-centric software environment, in which the computer-based systems have no understanding of what is being processed and virtually all interpretation tasks must be performed by human operators, more apparent than in the intelligence community. While the *information-centric* building blocks that would allow computer software to play a far more powerful role in intelligence analysis and evaluation processes have been available for at least two decades, the intelligence community like most other computer users has been reluctant to take advantage of these potential capabilities. Clearly, this reluctance is not based on technical obstacles but on the innately human resistance to change. Unfortunately, only a compelling reason such as the post-September 11 (2001) terrorist threat will provide the necessary incentive for the timely adoption of *information-centric* software design and implementation principles.

The benefits of such a paradigm shift will be immediate with startling results, even in the initial relatively primitive implementations. Returning to the previous example, the four typical intelligence messages would be processed very differently by ontology-based software that provides sufficient context for some degree of automatic reasoning by software agents. What is

immediately obvious in Figure 9 is that key elements of the messages are treated as objects with attendant characteristics and relationships.

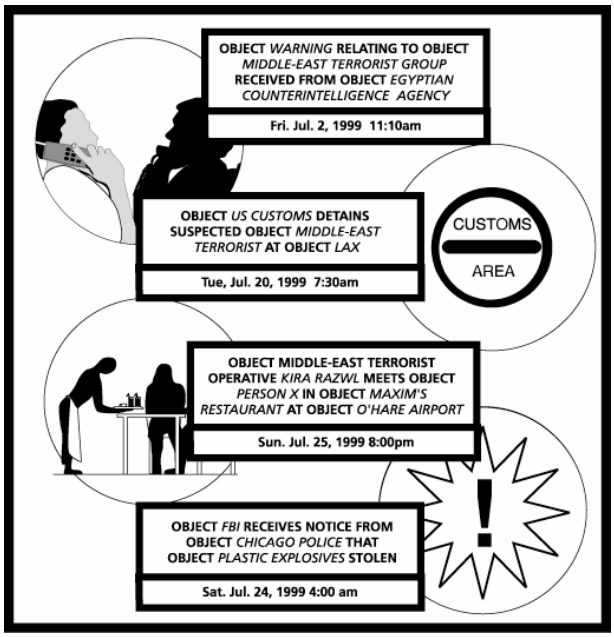

Figure 9: Four typical intelligence messages (***information-centric*** software environment)

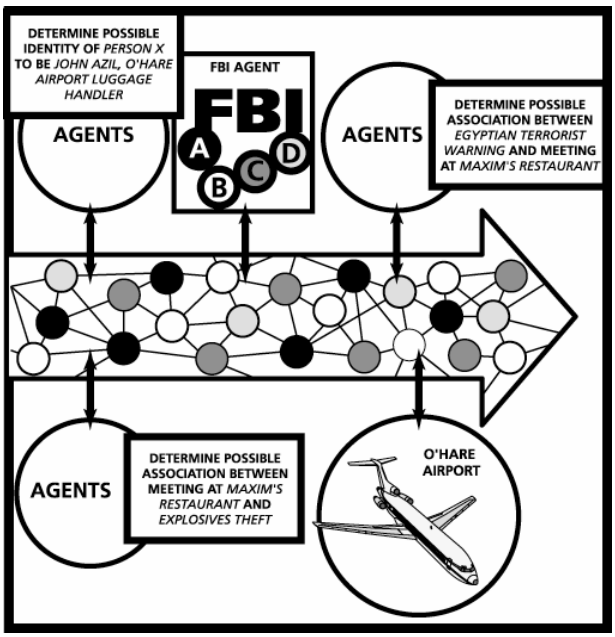

Figure 10: Automatic processing of messages (***information-centric*** software environment)

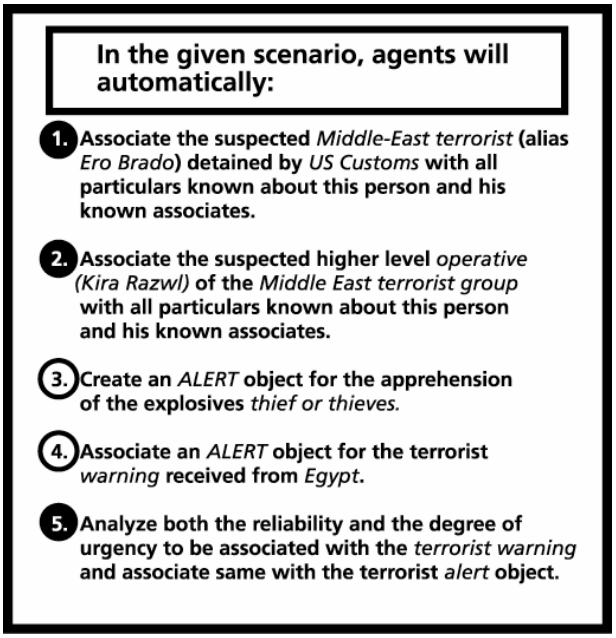

Figure 11: Automatic processing of associations (***information-centric*** software environment)

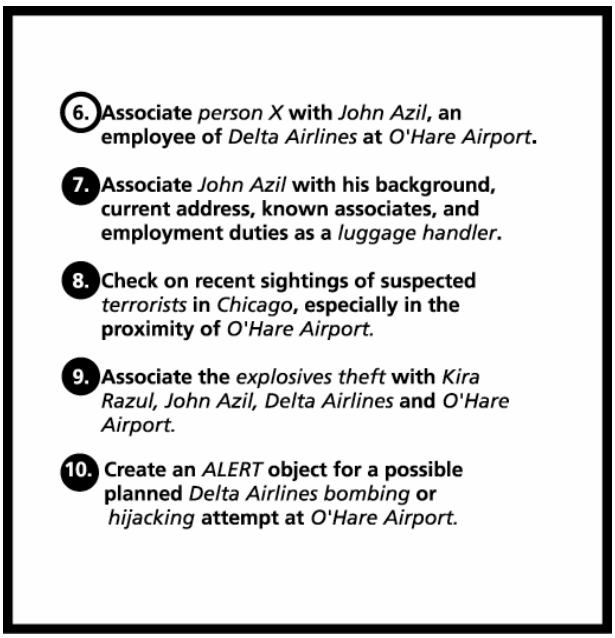

Figure 12: Automatic processing of inferences (***information-centric*** software environment)

For example in the first message, by being represented as objects within an information model the Egyptian Counterintelligence Agency, the Middle-East Terrorist Group, and the Warning itself, provide a rich context from which inferences can be drawn. This context includes not only

the information that is already available about the Middle-East Terrorist Group and the Egyptian Counterintelligence Agency, but also the relationships among these entities and other entities (i.e., objects) represented in the information framework. This allows software agents (Figure 10) to automatically explore numerous associations and possible conclusions in parallel and without initial human assistance. As a direct consequence scarce human resources can be employed at higher levels of intelligence analysis and evaluation, after most of the tedious low level filtering and data correlation tasks have been accomplished.

A typical sample of the kinds of automatic analysis and reasoning that could be performed by software agents in the given example is shown in Figures 11 and 12. Not only would the agents be able to automatically access relevant databases to enrich the information environment, but they would also be able to correlate data sources, form associations, detect patterns, and explore possible conclusions. The number of software agents performing these tasks in parallel would vary depending on current needs and hardware capabilities. In other words, agents could be cloned by other agents, or even themselves, as demanded by the workload.

7. Concluding Remarks

Endowing computer software with a virtual representation of the real world context that is required for the automated interpretation of data will not only allow us to apply far superior security and information assurance methodologies, but also provide the basis for a major leap in the exploitation of computer-based capabilities.

Moving from data-processing to *information-centric* software constitutes a paradigm shift in capabilities and human attitudes. It is not a question of whether this paradigm shift will take place, but how long will it take. While the miniaturization of electronic components has advanced in a proactive mode at an astounding rate over the past two decades, the exploitation of these newfound hardware capabilities with commensurate advances in software has progressed more slowly in a reactive fashion. The principal reason for this reactive inertia is the natural human resistance to change. We typically change only in response to a serious threat or to take advantage of an obvious and greatly enticing opportunity. Both are present at this time. We are facing a compelling and dangerous cyber threat and we are beckoned by an opportunity of great economic gain.

References

Biermann A. and J. Feldman (1972); 'A Survey of Results in Grammatical Inference'; Academic Press, New York.

CBS (2009); 'Cyber War: Sabotaging the System'; CBS 60 Minutes Series, 8 November.

CNN (2009); 'Hackers stole data on Pentagon's newest fighter jet'; Mike Mount, CNN.com/US, 21 April.

Cohen B. L. and C. A. Sammut (1978); 'Pattern Recognition and Learning With a Structural Description Language'; Proceedings Fourth International Joint Conference on Artificial Intelligence, IJCPR, Kyoto, Japan (pp.394).

Ganek A. and T. Corbi (2003); 'The Dawning of the Autonomic Computing Era'; IBM Systems Journal, 42(1) (pp.5-18).

Patterson D., A. Brown, P. Broadwell, G. Candea, M. Chen, J. Cutler, P. Enriquez, A. Fox, E. Kiziman, M. Merzbacher, D. Oppenheimer, N. Sastry, W. Tetzlaff, J. Traupman and N. Treuhaft (2002); 'Recovery-Oriented Computing (ROC): Motivation, Definition, Techniques, and Case Studies'; UC Berkeley, Computer Science Technical Report (UCB//CSD-02-1175), University of California, Berkeley, California, March 15.

Winston P. (1970); 'Learning Structural Descriptions from Examples'; Technical report AI-TR-231, MIT, Cambridge, Massachusetts, September.

Zetter K. (2010); 'Hackers Targeted Oil Companies for Oil-Location Data'; 26 January (www.wired.com/threatlevel/2010/01/hack-for-oil).