AvesTerra

A Reference Architecture for Global-Scale Knowledge Orchestration

The FOUR-Color Framework

April 2016

GEORGETOWN UNIVERSITY

Lawrence Livermore National Laboratory

OAK RIDGE National Laboratory

Pacific Northwest NATIONAL LABORATORY

Table of Contents

Preface

This document is the first in a series of companion documents that collectively describe the AvesTerra system for global-scale information sharing, knowledge representation, and analytic interoperability. The key AvesTerra documents include:

- AvesTerra: FOUR-Color Framework

> **The *FOUR-Color Framework* document (presented here) provides an overview of the basic principles and architectural aspects of AvesTerra's approach to information sharing and analytic interoperability, including a description of the end-to-end system structure and each of the main four layers of the design. AvesTerra is the first major effort to make use of the full spectrum of FOUR-Color architectural principles.**

- AvesTerra: Hypergraph Transaction Protocol (HGTP)

> The *HGTP docucment* provides a specification of the communication protocol used to communicate between AvesTerra clients and AvesTerra servers, and between AvesTerra servers when configured and operating in peer-to-peer fashion.

- AvesTerra: Application Programming Interface (API)

> The *API* document describes the key application programming interface constructs, illustrating how they can be used in support of real use-case driven applications. Examples are provided from ongoing AvesTerra research and development program areas.

- AvesTerra: Integration and Application Library (AVIAL)

> AVIAL presents the definition and technical details of the layer of standardized tools and services build directly atop the AvesTerra API. This document is intended for implementers interested in integrating with AvesTerra on new IT infrastructures and diverse computational platforms and analytic engines, as well as interfacing new data sources, analytics, and visualization components into the AvesTerra ecosystem.

Acknowledgement

This paper is a result of enumerable lengthy technical conversations with an extraordinary community of researchers, analysts, and computational science talent spanning the Nation over the past decade. Any glimmer of vision contained within clearly results from standing on their collective tall shoulders. Special thanks to members of the Lawrence Livermore National Laboratory, Oak Ridge National Laboratory, Pacific Northwest National Laboratory, Sandia National Laboratory, National Security Agency, the Defense Intelligence Agency, the National Geospatial Agency, Johns Hopkins University-Applied Physics Laboratory, University of Texas-Austin, University of Maryland-College Park (UMIACS), Dartmouth University, Southern Methodist University, Carnegie Mellon University, Raytheon Company, EMC, Cisco Systems, Apple, BBN, Sun Microsystems, SGI, Metron, ParAccel, Akamai Technologies, Cray, Objectivity, Composite Software, AGI, HighFleet Inc, and a vast list of American small businesses, too lengthy unfortunately to list here, for contributing heavily and advancing this important discussion. Specific contributors to the framework's formulation include Bob Adolf, Bill Anderson, Sid Berkowitz, Robert Burleson, David Besemer, David Bridgeland, Travis Breaux, Fred Chang, Michelle Crawford, Jeffrey Collmann, Spiros Dimolitsas, George Economou, Jim Ferry, Matt Gaston, Charles Granderson, Mark Guiton, Dan Hall, Michael Liggett, Kimbry McClure, Natalio Pincever, Shari Pfleeger, Stacy Prowell, William Ryder, J. C. Smart, Brian Urch, Everett Wheelock, Brian Worely, and Barry Zane.

For additional information, contact:

J. C. Smart, Ph.D.
Chief Scientist, AvesTerra Program
Research Professor
Department of Computer Science
Georgetown University
37th and O Street, NW
Washington, D.C. 20057-1241
smart@georgetown.edu
(443) 745-5876

Introduction

Our world is a very complex place. It is faced with many difficult, interdependent problems; some caused by nature, others caused by man. As individuals, families, communities, and nations, we face an ever changing and compounding series of complex challenges and threats. Defense, health, climate, food, cyber, energy, transportation, education, weather, the economy. Problems in each of these areas threaten our health, our safety, our security, our livelihood, and our sustainability. We seek improved capabilities to help detect, understand, mitigate, and prevent this brave new world of threats. To address these problems, we invariably resort to science, our systematic enterprise for building and organizing knowledge that helps us understand, explain, and predict our world around us. At the core of science is inquiry. We formulate questions. We generate hypotheses. We predict consequences. We experiment. We analyze. We evaluate. We repeat.

Fueling the scientific process are the observations we make and the data we collect. With the advent of the 21st Century telecommunications explosion, data is now flowing and evolving all around us, in massive volumes with countless new streams, mixing and shifting each minute. Cyberspace is enormous and continuously changing. And by many accounts, its expansion and movement has only just begun. Analyzing and understanding this vast new ocean of data is now of paramount importance to addressing many of the growing problems facing our world.

Today's data analytic industry is vibrant with a continuous supply of new and innovative products, services, and techniques that thrive and prosper based on their relative merits in the respective marketplaces. Unfortunately, these components are rarely interoperable at appreciable scale. Moreover, the rapid proliferation of analytic tools has further compounded the problem. With only loose coordination, these partial solutions are ineffective at combating the broad spectrum of problems. Attempting to impose a "one-size-fits-all" analytic solution, however, across today's tremendous data expanse poses significant scientific, technical, social, political, and economic concerns. Consequently, an enormous amount of resources must regularly be expended addressing isolated issues and mitigating specific threats. Thus, the analytic community faces considerable challenges dealing with major classes of problems, particularly those at national and international levels.

Advocated here is an approach that leverages the uniqueness of local data sources and local analytic techniques, but weaves all such point solutions together into a powerful collaborative fabric. Great care, however, must be taken with this stitching process. The further data flows from its source, and the more information that is aggregated by an increasing number of parties, the greater the privacy concerns along with an accompanying loss of autonomy. In addition, the movement of data itself between systems risks information disclosure and/or compromise. This is of particular concern in collaborative settings where it may be difficult or impossible to ascertain or maintain trust among the participants.

This paper presents a new architectural approach for organizing the Brave New World of data, sharing information derived from that data, and performing analysis at the never before possible global scale. This approach employs a powerful integration of proven capabilities selected from across the government, commercial, laboratory, and academic sectors. The resulting framework offers participants a pragmatic means to think globally, leveraging the aggregate of available knowledge to make the best well-informed regional and local decisions. This new approach is referred to here as the FOUR-Color Framework (4CF).

The 4CF identifies four fundamental layers of technology, depicted in Figure 1. These four layers each serve a critical core function. In unison, these functions work together to provide an end-to-end solution for extreme-scale information sharing and analysis. These four specific layers were very carefully formulated from fundamental computational science principles to ensure rapid and open innovation and leverage capabilities proven within other domains. For simplicity, each layer is assigned a color with which it is identified:

- **Red Layer "Source Data Systems"** –This layer is composed of the diverse, complex collection of non-shared data, shared raw data, and shared correlated data spanning public, private, commercial, and government data systems including databases, repositories, files, web services, archives, live data-feeds, real-time control systems, etc., highly distributed throughout an enterprise, a region, the nation, or across the planet.

- **Blue Layer "Integration & Isolation"** – This layer provides a very robust privacy and security isolation boundary that is extremely difficult to physically or electronically defeat. This boundary enables highly distributed **Red** Layer edge systems to interoperate at true global scale without resorting to data aggregation or centralized information management. This is accomplished via a unique data transformation process that enables the **Black** Layer above to exist and operate in true distributed fashion.

- **Black Layer "Information Sharing and Analysis"** – This layer provides a unification of the many **Red** Layer systems beneath, providing a distributive, privacy-assured information sharing and knowledge representation fabric for collaborative global-scale analysis. This layer was formulated with the rigorous protection personal information and strict privacy enforcement as fundamental requirements

- **Green Layer "Visualization and Decision Support"** – This layer interacts directly with users, researchers, scientists, and operators, providing the visualization applications, decision support functions, situational awareness services, prediction and mitigation tools, large-scale simulation, alerting and notification, and command and control capabilities.

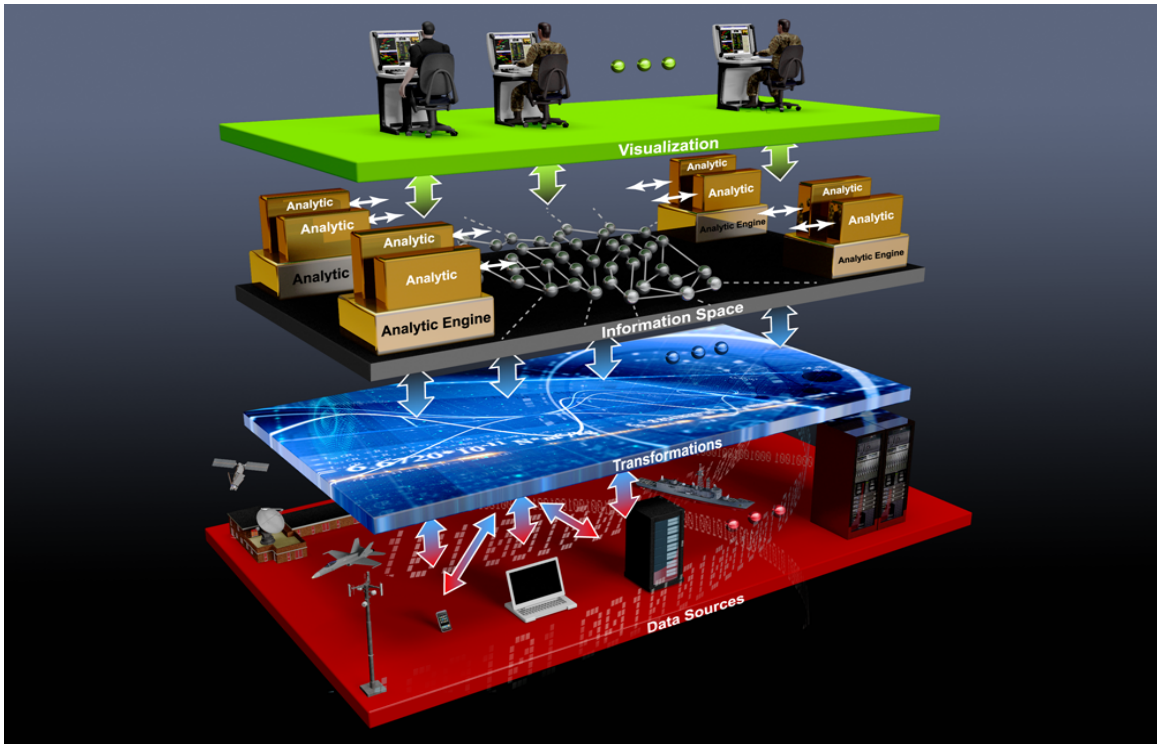

Figure 1 – The FOUR-Color Framework

The resulting 4CF approach requires no new science, but is poised to enable a new spectrum of integrative research, insight, and discovery. It embraces civil liberties and the protection of personal information as fundamental system requirements. It accomplishes information sharing without loss of individual privacy, and imposes no unnecessary or unwelcome government controls. The approach is fundamentally open, requiring no single centralized authority, and welcomes broad participation from across commercial industry, academia, government, the defense industrial base, and the non-profit sector. It implies not a regulatory environment, but an incentivized "opt-in/opt-out" setting tailored to an individual participant's preference and legal authorities.

Information sharing arguably ranks as one of the more complex technological challenges in our history, with an intricate web of difficult and unresolved social and political issues. Indeed, it might easily be characterized as a "wicked problem", belonging to the class of extremely difficult contemporary challenges with complex, interdependent requirements that are incomplete, contradictory, and ever changing. Thus, instantiation of the 4CF is understandably not for the timid. It implies the construction of some unique regional, national, and/or international class assets, integrated in very special ways. This integration, however, promises an unprecedented global situational awareness and analysis capability to help address our most pressing problems. While attempting to make neither false promises, nor offer false hopes, the authors of this work are both confident and passionate of its effectiveness.

Foundational Principles

Analysis thrives on data. In our 21st century globally connected world, the supply of data is now seemingly boundless. What was once a scare commodity is now an overly abundant feedstock. When plotted over time, the volume of available data flowing throughout the world invariably follows a trend similar to that in Figure 2. That is, data exhibits near exponential growth ultimately slowing, at best, to extremely steep linear growth. If the world's entire telecommunications infrastructure was suddenly completely built out and all available bandwidth fully utilized, the volume of new data generated and transmitted would continue at an enormous rate, barring some disaster of cataclysmic proportions. This long-term, sustainable data supply is generally good news for the analytic community. Unfortunately, the very fuel that propels analysis can also set it ablaze. The ability to collect, move, store, and process data is bounded by the availability of adequate computing resources and telecommunications infrastructure. These resources are similarly limited by economics and more mundane realities such as available space, power, and cooling to support them. What was once a dire analytic thirst has now been replaced by a veritable analytic drowning: too much data without sufficient processing, communication, storage, electrical power, and cooling to handle it.

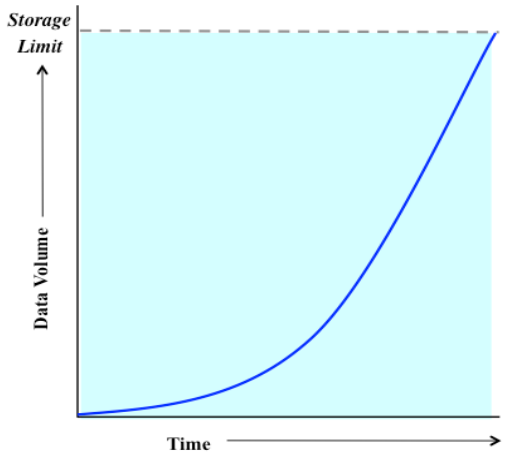

Figure 2 – The nature of *Data* discovery over time

During the past decade, numerous technical solutions have been advanced to address this problem. Included on the list are large-scale databases, federated query systems, data brokers, data mining, data warehousing, distributed data storage systems, distributed databases, cluster computing, grid computing, cloud computing, and cloud storage. All of these techniques have made substantial progress, exploiting parallelization and computing resource utilization. Unfortunately, with the sudden "slowing" of Moore's law behavior and the current limits of software parallelization technology, these techniques can promise only modest performance gains for the foreseeable horizon. The exponential behavior of data growth, however, continues to risk rendering them all inadequate in the not too distant future. Furthermore, the real-

world costs and schedules required to deal with exponentially increasing data volumes can be quite sobering, measuring in the hundreds of millions to billions of dollars. Since budgets are not inexhaustible, this unfortunately equates to lack of capabilities, and/or poor utilization of valuable analytic resources. As it turns out, there is a silver lining to the data explosion problem, with the analytic community as the primary benefactor.

While the flow of data is seemingly endless, the amount of "information" conveyed by that data is much more modest. For this to become apparent, the notion of information is carefully constrained as in some historical contexts it has been used interchangeably with the term "data". While *Data* is regarded as the raw 1's and 0's that move across cables, transmitted through the air, stored on disks, etc., *Information* is characterized here as a representation of the real world entities that the data describes. For example, while a continuous stream of e-mails may flow between various nodes in a network, the set of communicants associated with these messages are relatively finite by comparison, with an upper bound limited by the size of the user base. In fact, the observation rate of *new* communicants exchanging messages in a network invariably follows a curve that resembles Figure 3. This is a critical distinction. While Figure 2 grows exponentially (or very steep linearly), Figure 3 grows asymptotically to some upper bound. This upper bound is simply the maximum number of entities of a particular type that can exist (i.e. population limit). While this may be quite large (e.g. number of people on the planet), this number is relatively finite by comparison, growing only slowing (i.e. ~1%) when contrasted with current data growth (e.g. ~45%). By carefully formulating an appropriate set of Information entities (i.e. those entities types that follow Figure 3 versus Figure 2), a significant analytic paradigm shift is possible. The key enabler is that analysis tasks are often focused on a relatively limited set of real world entity types (e.g. people, organizations, computers, etc.) and that there exists only a finite number of instances of the these object types. This is in sharp contrast to the endless nature of data that describes a multitude of different aspects of these entities. This is the single most fundamental principle of the analytic approach discussed here. The 4CF exploits this notion heavily.

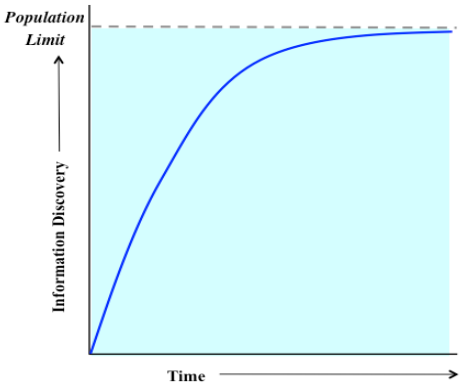

Figure 3 – The nature of *Information* discovery over time

While data growth has been proceeding on an exponential path, the preserved volume of data over time $d(t)$ must ultimately be linearly limited by the maximum physical data collection rate at the sources multiplied by the filtering rate at which data must be discarded due to finite storage requirements, or

$$d(t) \rightarrow c*(t - t_0) * f$$

where c = the collection rate and f = the filtering rate. To keep up, current data processing practice must either increase the storage limit or increase the filtering rate. Unfortunately, neither option is attractive. A continuous increase in storage implies a continue increase in cost (and associated power, space, and cooling); discarding data implies an irretrievable loss of an important asset. The asymptotic nature of information discovery holds the key. That is, for each data source, the discovery of "information" over time can be calculated via

$$i(t) = (1\text{-}e^{-t/T})$$

where n_0 is represents the entity population limit and T is the "time constant" of the source with $T = n_0/r$ where r = the information entity collection rate. Interestingly, each of these quantities can be measured and/or calculated for every data source, providing key architectural metrics.

A large segment of analysis is dedicated to understanding real-world (or virtual world) entities and their complex interrelationships and dependencies. In computer science jargon, a representation of this understanding is frequently referred to as computational *Knowledge*. The 4CF was specifically formulated to aid the representation and capture of such knowledge. This is most invaluable, although perhaps not immediately obvious. Referring back to Figure 3, recall the asymptotic behavior of Information discovery. Although the time 'constants' may actually vary over extended periods, observations of specific types of entity relationships exhibit this same asymptotic behavior. Of keen analytic interest is when the entities and their relationships are aggregated into a single construct. The power of this knowledge aggregation process is now a fairly widely known and frequently applied, leveraging basic graph theory (i.e., loosely speaking, the mathematical study of node/link structures). The 4CF applies this process at global scale to aid the creation of a knowledge base with world coverage. The estimated upper bound of such a knowledge base equates roughly to a graph of approximately one trillion nodes and one quadrillion links.

The utility of a knowledge representation of our world would be extraordinary. Knowledge aggregation (in contrast to data aggregation) is an extremely powerful method for enabling a significant boost to analytic yield. In simple terms, analytic yield can be loosely described as the amount of analytic "work" that can be accomplished during a fixed duration of time. While analytic work performed by a person is generally considerably richer than that of a machine, a computer is particularly adept at examining exceptionally large volumes of entities and relationships for specific patterns.

When knowledge is aggregated in this fashion, analytic yield increases significantly once a "critical mass" is reached, as shown in Figure 4. Achieving this high analytic yield is the primary goal of the 4CF.

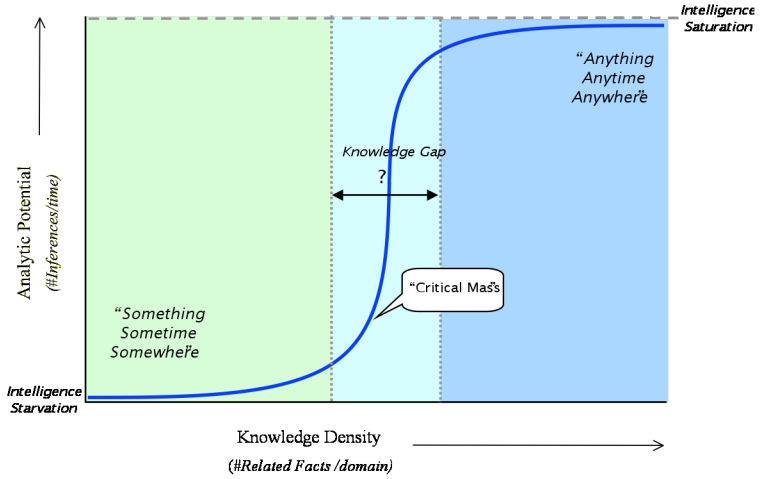

Figure 4. The nature of *Knowledge* aggregation

These notions are calculable, providing two important quantitative 4CF metrics. The *knowledge density* δ of an information space characterizes its connectedness and can be determined by the formula

$$\delta = \frac{|E|}{|V|^{\lambda}}$$

where $|V|$ is the number nodes (entities), $|E|$ is the number of links (relationships). The exponent λ in the denominator is used for normalization purposes depending upon the domain since $|E|$ is potentially $O(n^2)$ larger than $|V|$. An information space with $\delta = 1$ implies a fully connected knowledge base; $\delta = 0$ is fully disconnected.

The notion of *analytic yield* characterizes the number of computational inferences that an analytic engine could potentially perform given a specific knowledge base. Analytic yield is thus likened to potential energy (or work) in physics and can be computed via the formula

$$w \approx J(G) = \sum_{i=1}^{m} \frac{|H_i|^2}{|V|^2}$$

where H_i is the i'th connected component of the m-connected graph $G = (V,E)$ that represents the knowledge base.

The key milestone of a 4CF instantiation is to achieve a "critical mass" density for a given set of data sources such that the analytic yield is at least, say, 10X over conventional data-centric processing approaches. Used in combination, these metrics can be

incorporated into an overall integration *analytic power* rating p to compare performance and effectiveness of 4CF instantiations and competing architectural approaches:

$$p = \frac{dw}{dt}$$

Technical Approach

The main components of the 4CF are shown in Figure 5. The 4CF is a four-layer system design, with each layer identified by a color, **Red**, **Blue**, **Black**, and **Green**, respectively. At the heart of this architecture is the **Black** Layer that contains the FOUR-Color information space (i.e. the knowledge graph), a very large distributed set of objects that collectively represents the world's physical (and virtual) entities and their relationships. The structure of this space is that of an extremely large (hyper) graph, up to ~one trillion nodes and ~one quadrillion links. At the bottom of Figure 5 are the data sources, S_1, S_2, ..., S_s (i.e. the **Red** Layer). It is assumed that the number of such sources is substantial (i.e. thousands) and that each source can be very diverse in size and format, ranging from fixed flat files to massive real-time streaming flows, both structured and unstructured. Upon the **Red** Layer is a set of transformations T_1, T_2, ..., T_t that comprise the **Blue** Layer. The role of each **Blue** Layer transformation is to continuously extract entity and relationship state and attribute information from each data source and update the information space managed by the **Black** Layer above. Access to this information space is provided by a single open, standardized Application Programming Interface (API).

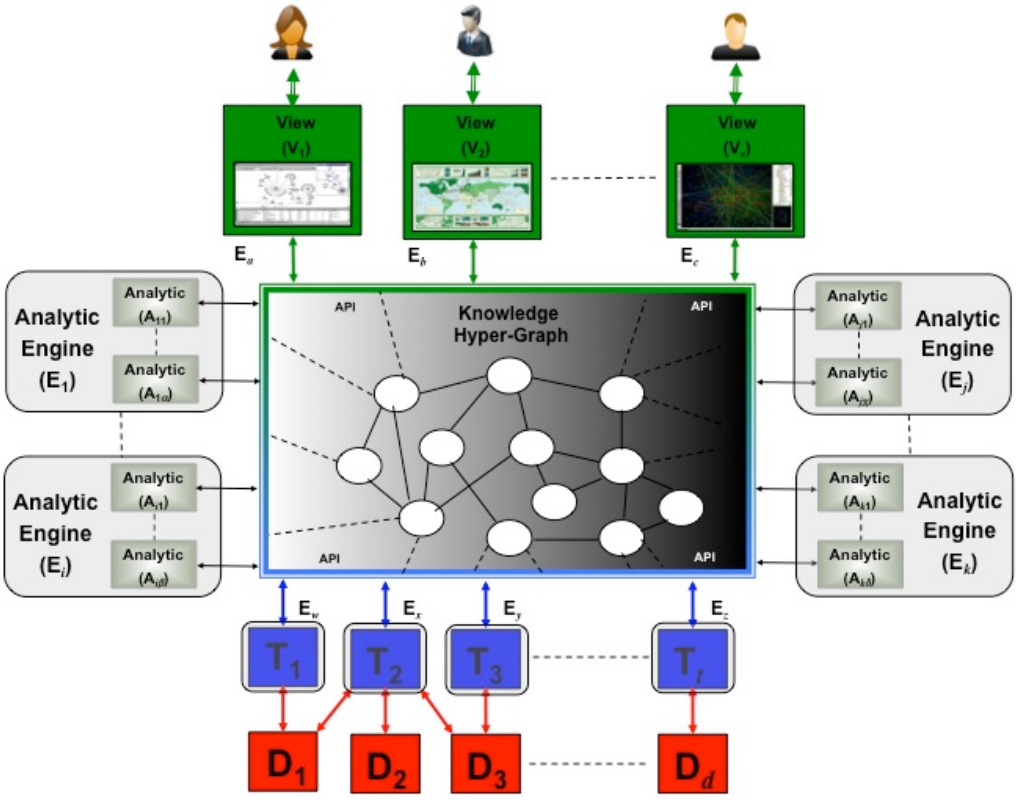

Figure 5. FOUR-Color **Framework**

Surrounding the information space are a set of analytic engines, E_1, E_2, ..., E_e. An analytic engine is a compute platform specifically designed to accelerate certain classes of analytic algorithms (e.g. hydrodynamic-based, graph-based, relational-based, table-based, statistical-based, etc.). Each analytic engine E_i hosts a set of analytics A_{i1}, A_{i2}, A_{i3}, ..., A_{ia} offered by the community for community use. These analytics interact with the information space via the common API. The **Blue** Layer transformations are actually, symmetrically, considered simply a special case of analytic that interfaces directly to one or more data sources. Each transformation T_i is hosted too on an analytic engine E_j which is sharable with other analytics and/or transformations, perhaps tuned for its underlying computational platform.

At the top of Figure 5 are a collection of **Green** Layer views V_1, V_2, ..., V_v. Views are end-user applications that provide visualization and decision support capabilities. All **Green** Layer applications access the **Black** Layer information space using the same exact API as the transformations and the analytics. Hence, the power of the 4CF hinges very heavily on this API. The 4CF API is motivated by the rapid information overlay technology concept, a technique for quickly constructing extremely large distributed object overlays. This overlay technology comprises a simple set of general, core primitives, carefully selected to enable global scalability and proof-of-correctness. These primitives allow a client (e.g. an analytic) to connect to the information space, create and delete objects, remotely invoke object methods (e.g. "get" and "set" data), and asynchronously publish and subscribe to object events. Using this set of primitives, remarkable scalability can be achieved. In addition, these primitives enable an extremely robust form of information space isolation in support of cyber security and privacy enforcement.

While the 4CF prescribes only one core API that is identical for all 4CF instantiations, the implementation of this API can vary considerably across different platforms (i.e. "E_i"). This is quite intentional, and both one of the 4CF architecture's most attractive features, and one of its primary technical challenges. By enabling the implementation internals to change from platform to platform, a 4CF implementation is able to better exploit the specific performance features of the local platform. The 4CF importantly recognizes that no single platform is ideal for all classes of analytics and sources, and that performance can vary greatly by as much as six orders of magnitude or more depending upon the specific algorithm, data types, and the platform choices. By maintaining a common API, a high degree of analytic interoperability and commonality is achieved. But by varying the API implementations, significant analytic performance can be obtained by tuning its implementation to best leverage the computation features of the underlying platform. This tuning often involves the degree at which information space caching and pre-caching is performed. For example, on a very large shared memory machine, the entire global knowledge graph might be loaded into main memory for near constant time traversals. For global search applications, however, a MAP-REDUCE infrastructure would be considerably more effective. Similarly, for transaction oriented analytic operations, a columnar or conventional database platform would be more ideal.

The 4CF enables these platforms to be mixed and matched to better balance costs and performance while preserving the more important investment made in the design and implementation of the analytics themselves. This separation of the API from its implementation reduces the dependency and coupling of upper layer components on the lower-layer infrastructure, thus considerably extending the end-to-end system architecture lifecycle.

Privacy Assurance

Systems that aggregate information must carefully balance security goals with the protection of civil liberties, often by limiting the scope of information allowed into the system. The 4CF challenges this paradigm, allowing the aggregation of very sensitive information to uncover deeply rooted, insidious problems or threats without exposing personal or private information to inspection. This is made possible through the creation of an unsearchable, formally verifiable information storage system and automated matching technique that strictly prohibits human inspection of any private information.

Within the 4CF **Black** layer, information must be analyzed in accordance with policy and law. Thus, it is within the **Black** layer that automated implementation of privacy is maintained. This is shown notionally in Figure 6. In its most robust protected form, the **Black** layer is viewed as a strict "black box" that makes absolutely no allowance for *any* access to the information contained inside. This security concept allows information to be aggregated but prevents it from ever being released outside of its confines. Rather, policy/law compliant patterns approved by an appropriate policy body are provided to the black box through a very strictly controlled interface. The black box operates by outputting only the identifiers of patterns that are detected, and any associated information that the policy body unanimously pre-approved for reporting. Participating organizations use these automated pattern detection reports to initiate legally appropriate actions to further investigate within their currently existing policy/legal authorities. As privacy concerns and authorities vary widely from organization to organization, the 4CF recognizes that a spectrum of black box containers, each with a varying degree of data/privacy protections is needed. Thus the 4CF enables black boxes of varying shades (i.e. grey boxes) each with differing levels of privacy restrictions, such as "Public/Open", "Sensitive", "Restricted", "Classified", "Compartmented", "Strictly Private", etc.

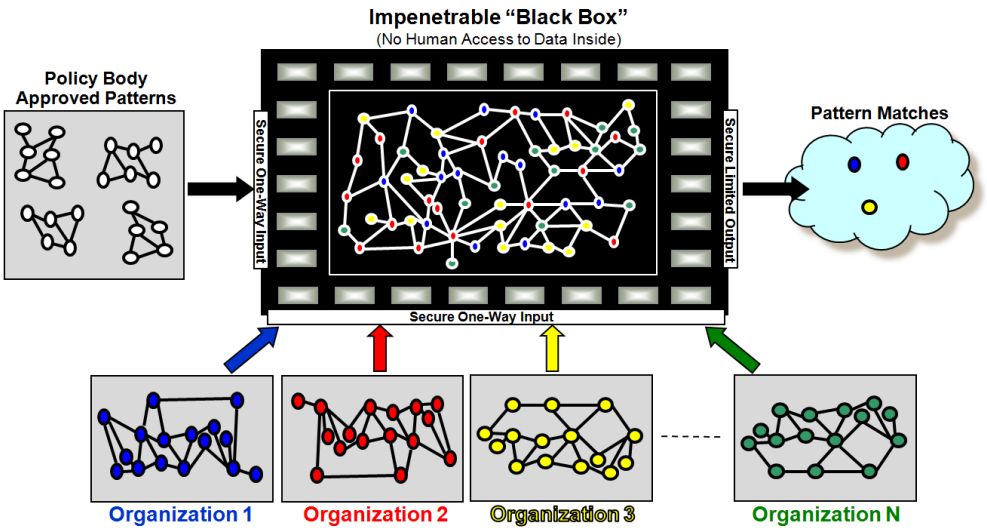

Figure 6. The "**Black** Layer" Privacy Assurance Construct

Layer 1 (Red) - Source Data Systems

The **Red** Layer provides the foundational data with which to build, share, and analyze information. The **Red** Layer is comprised of potentially multitudes of **Red** systems throughout the vast data landscape. The **Red** Layer represents this massive heterogeneous collection of data, databases, archives, real-time feed, networks, systems, and solutions spanning the global enterprise. Figure 7 contains an overview of the key **Red** Layer components. Within the 4CF, the primary requirement imposed on source **Red** systems is that each provide some sort of interfacing mechanism or API to enable access to the allowable functions they choose to offer and support within a 4CF environment. It is via this local API that a **Red** system is connected into the larger 4CF enterprise. This connection is accomplished via a Composite Adapter. A Composite Adapter is a software (typically, but not necessarily) subsystem that interfaces a **Red** system via its API to the **Blue** Layer above.

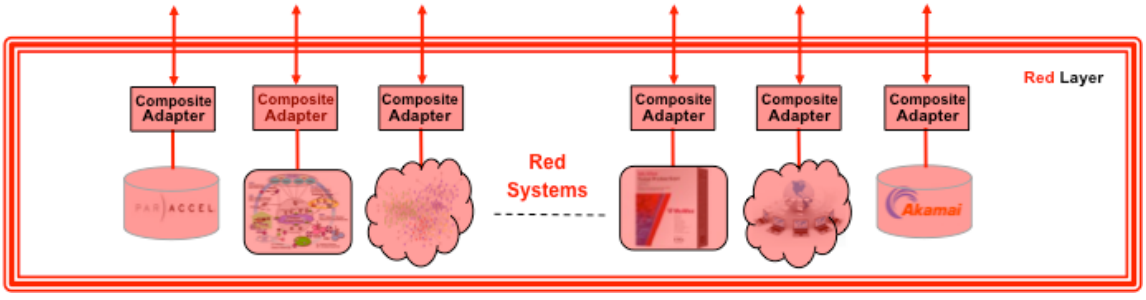

Figure 7. The **Red** Layer

The Composite Adapter is important for several reasons. It is assumed that **Red** systems are enormously diverse in their capabilities, functions, and implementations. The design of this adapter structure is formulated to dramatically decrease the cost and

time associated with integrating such diverse systems at such large scale. The role of the Composite Adapter is to isolate all unique, custom interfacing work for interacting with **Red** system source data into exactly one location in the **Red** Layer so that the other components across the 4CF may be standardized and generic. This is an extremely powerful and discriminating offering. This structure promotes an open "culture" for large-scale, highly secure distributed systems integration. That is, all data systems desiring 4CF integration can contribute regardless of their capabilities, but without fear of compromise. Systems with a very rich API would be able to leverage all features and benefits of the 4CF while systems with limited APIs would only participate up to their limits of capability. The Composite Adapter handles this often complex, troublesome interfacing burden, transforming the **Red** system's custom API into a 4CF compliant component. This transformation is aided via a standardized, non-proprietary protocol to the **Blue** Layer above.

Working broadly across all systems and leveraging what is possible within those systems allows the 4CF to be as effective as practically possible. To assist new developers with maximizing interoperability, the **Red** to **Blue** Layer interface specification is intended to be open source, freely available allowing the widest possible set of solutions. Reference implementations of Composite Adapters are publishable to ease the development of solutions by local entities and commercial suppliers. Over time it is envisioned and encouraged that **Red** system developers will expose services in a **Blue** Layer compliant fashion reducing or precluding the need for the custom Composite Adapter plumbing. Indeed, it is conceivable that **Red** system developers may desire to interact directly with the **Blue** Layer interface to better optimize their own internal organization and performance characteristics. These alternative implementations and gradual evolutions are welcome and encouraged.

Layer 2 (Blue) – Integration and Isolation

Beneath the **Blue** Layer, the **Red** Layer embodies the vast expanse of cyberspace where the data systems throughout are designed and operated with an enormous range of trust, classification, and privacy constraints. To address this complex integration challenge, the **Blue** Layer's primary purpose is to provide a common, robust, scalable access mechanism to **Red** Layer information, but without sacrificing local autonomy, jeopardizing system security and integrity, or violating privacy. Essentially, the **Blue** Layer performs an integration service across the entire spectrum of **Red** Layer products, services and systems, providing a fabric that weaves together each into a single cohesive, unified framework. The **Blue** Layer's accomplishes this "plumbing" challenge through a series of powerful information transformations specifically formulated for large-scale distributed systems operation. These transformations effectively map **Red** layer data exhibiting exponential growth (Figure 2) into **Black** Layer information entities and relationships with limited asymptotic behavior (Figure 3).

As a result of these transformations, the **Blue** Layer enables the creation of an extremely strong security isolation barrier to prevent unauthorized data breaches, vulnerability-inducing data or cyber contamination (e.g. malware transmission), or usurpation of control (e.g. hacking). This barrier establishes a trust boundary above which **Black** Layer analytic processing can be performed, but without of compromise to any constituent **Red** components. Similarly, the barrier prevents individual **Red** components from compromising the integrity of **Black** Layer operations or another **Red** Layer component in an aggregate 4CF enterprise. To achieve this high degree of integrity, the 4CF **Blue** Layer boundary is designed specifically so that the interface implementation can be rigorously defined and mathematically proven. The robustness of this trust/isolation boundary is particularly important for applications requiring the highest levels of privacy protection and preservation (i.e. "Black Box" applications).

The key components of the **Blue** Layer are shown in Figure 8. Information distributed across **Red** Layer systems is pushed and pulled on demand to and from the **Black** Layer above through a series of **Red**–**Blue** adapter/converter channels. The **Red** adapters (in Figure 8) connect **Red** Layer systems to these channels via each system's respective **Red** Layer composite adapters (from Figure 7). **Red** adapters speak to upper **Blue** Layer components through a **Blue** Layer API that defines a special protocol. This protocol is based on the information overlay concept. That is, all communication to the upper layer components references 4CF entities, their relationships, attributes, and events. The **Red** adapters are responsible for transforming **Red** system data, as presented by the **Red** composite adapters, into these graph elements in conformance with the **Blue** Layer API. Stated differently, the **Red** adapters are the 4CF components that perform the transformation from the exponential-grown data streams (Figure 2) into the asymptotically limited information entity/relationship sets (Figure 3). By doing so, the **Black** Layer analytics above will need not be concerned with the specific implementation and data details of these elements.

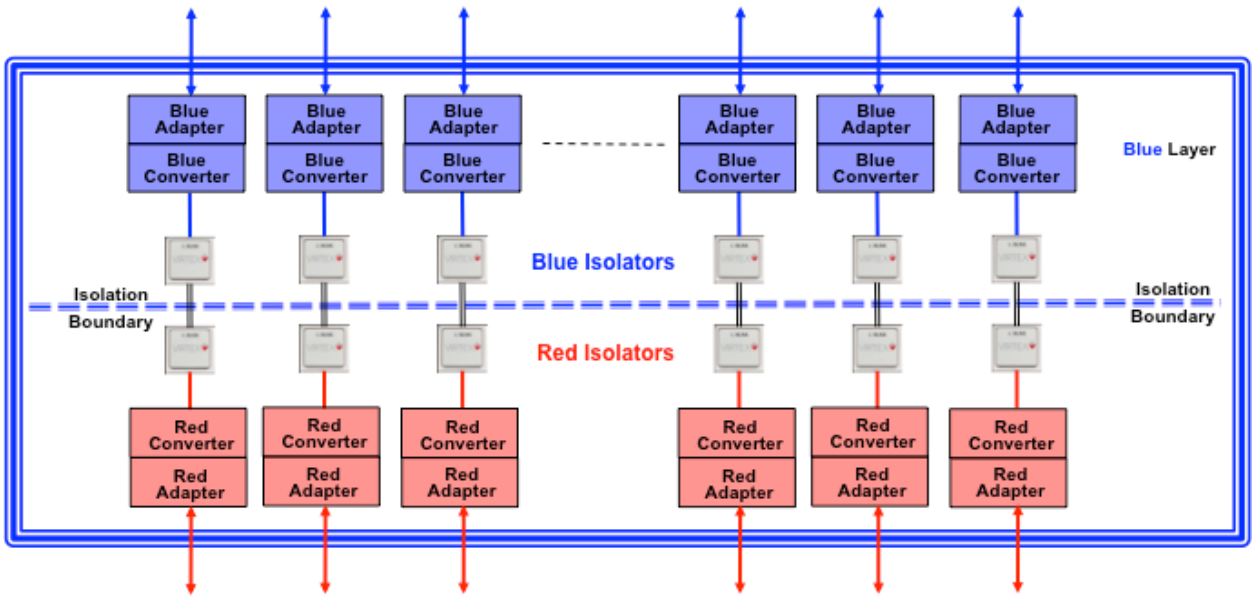

Figure 8. The **Blue** Layer

Upon the **Red** adapter's transformation of **Red** system data into graph elements, communication of these elements can now be serialized for security and privacy isolation purposes. The role of the **Red**-**Blue** converters is to do just that. That is, the **Red** converter takes incoming graph elements from the **Red** adapter and converts these elements into a set of serialized graph transactions (e.g. create node, create link, add attribute, remove attribute, etc.). These transactions are formatted using a simple, protocol consisting of well-defined operation codes and message parameters that fully encode the graph transaction. This encoded sequence is then delivered to the **Red** isolator for transmission across the **Red**/**Blue** isolation boundary to the corresponding **Blue** isolator. To ensure that only the properly structured transactions successfully cross the **Red** to **Blue** boundary, isolators are typically implemented as hardware devices such as Field Programmable Gate Arrays (FPGAs) or Application Specific Integrated Circuits (ASICs). Using formal methods based on model checking techniques for information processing circuitry, the operation of these devices can be verified mathematically to establish the correctness of their implementation. As these devices are built from combinatorial logic circuitry, verifying their correctness is a manageable process conducive to automation, whereas verifying an alternative conventional software implementation (e.g. code development in Java or C++) could quickly become unwieldy. The choice of hardware isolation makes this verification process tractable using contemporary proof techniques.

On the **Blue** side of the boundary, the **Blue** isolator and **Blue** converter perform the reverse operation of their **Red** counterparts. The **Blue** isolator device receives encoded graph transactions from the **Red** isolator, and then forwards these transactions to its corresponding **Blue** converter. The **Blue** converter converts the serialized graph transactions back into the equivalent entity/relationship representation. The **Blue**

adapter then contributes these elements to the **Black** Layer knowledge graph in conformance with the **Black** Layer API.

As mentioned, all **Blue** adapters interact with the **Black** Layer via the exact same API. That is, the specification that defines the complete set of operations and accompanying parameters that a **Blue** adapter can perform upon the **Black** Layer knowledge graph are identical for all **Blue** adapters, regardless of their location in a global 4CF enterprise. However, the specific implementation of this API on the local platform where this adapter is running can vary considerably. This discriminating 4CF feature is very important as it allows implementations to exploit the unique performance features of the local platform, yet maintain platform independence of developed software.

To further greatly reduce the 4CF implementation burden, the **Black** Layer API specification that is visible to all **Blue** adapters is actually also identical to the **Blue** Layer API specification visible to all **Red** adapters. The implementation of these APIs, however, is again quite different. The **Red** Converter, the **Red**-**Blue** isolator pair, and the **Blue** adapter essentially recreate the **Blue** Layer API at the **Black** Layer but with the critical interim hardware-assisted filtering steps necessary for security and privacy enforcement. Whereas a **Black** Layer API implementation exists within (above) the **Red**-**Blue** trust boundary, a **Blue** Layer API implementation is actually responsible for creating and enforcing that boundary. The **Blue** Layer enforcement mechanism employs the convert-isolator pairs for high-assurance operation founded on mathematical proof-of-correctness. It is envisioned that **Blue** Layer API implementations will be additionally augmented with anti-tamper capabilities to further deter cyber adversaries and attempts at privacy violation. The resulting **Blue** Layer component design makes the transmission and/or exploitation of malware, viruses, backdoors, Trojan horses, etc. extremely difficult, significantly raising the risk, level of sophistication, and amount of investment needed by an adversary.

The **Black** Layer API (and thus the **Blue** Layer API as well) is purposefully considered open-source and non-proprietary, with bindings to numerous computer programming languages, tools, and environments encouraged. In this manner, a **Red** Layer system can be constructed fully independent of other **Blue** and/or **Black** Layer implementations. In practice, a variety of **Blue** Layer implementations are envisioned depending upon the level of certification, degree of mathematical verification needed, and sophistication of anti-tamper protection necessary for the particular **Red** system information involved, up to and including those that involve the transmission of personal information that warrant the highest level of privacy protection.

Privacy Certification

The 4CF Black Box construct recognizes that the level of privacy obtainable is directly related to the level of system “impenetrability” that can be achieved, involving a risk–cost–benefit tradeoff. Depending upon the nature of the information to be protected, not all information sharing and analysis application will require the same degree of rigor to ensure adequate protections. Consequently, a multi-level privacy certification rating

is envisioned. Analogous with cryptographic systems, the following four levels of privacy certification are proposed:

- Type 1 Privacy: a device or system that is certified for government use to securely share and analyze private information consistent with the highest level of protections awarded by the U. S. Constitution. Achievement of this rating implies that all components of the end-to-end system have been subjected to strict verification procedures and protected against tampering via strict supply chain controls.

- Type 2 Privacy: a device or system that is certified for commercial use to securely share and analyze public information consistent with U.S. commercial law. Achievement of this rating implies that all interface components of the system have been subjected to strict verification and supply chain controls and that all other components have been subjected to reasonable best industry practices for operation verification and supply chain control.

- Type 3 Privacy: a device or system that is certified for public use to securely share and analyze sensitive information. Achievement of this rating implies that all components of the system have been subjected to reasonable best industry practices for operation verification and supply chain control.

- Type 4 Privacy: a device or system that is registered for information sharing and analysis, but not certified for privacy protection. No assumptions regarding component verification or supply chain controls are made about systems at this privacy protection level.

The **Blue** Layer mechanism described here works equally well for classified and specially compartment information sources. In fact, the design pattern generalizes easily. The exact same technique works well for large, complex global enterprises, enabling multiple privacy domains to be constructed and maintained as shown in Figure 8. Thus, **Blue** Layer technology enables a significant paradigm shift in the integration of systems, the sharing of information, and the protection of cyber infrastructure.

In summary, the **Blue** Layer is designed to provide extremely robust, secure interaction that is free from an enormous range of diverse threats, while simultaneously supporting an arbitrary number of privacy and/or security domains, all protected via the same formally verified and hardened interface. This implementation is extremely efficient, highly flexible, and globally scalable, enabling it to be readily tailored for a wide variety of organizational applications. Secure, enterprise level interoperability is achieved with minimal proprietary components, all thanks to the unique power and simplicity of the **Blue** Layer protocol.

Layer 3 (Black) – Information Sharing and Analysis

The analytic heart of the 4CF is the **Black** Layer. The **Black** Layer provides a collaborative, dynamic, and extensible fabric for organizing, sharing, and analyzing information at very large scale. The primary role of the **Black** Layer is to aggregate and provide secure, private analytic access to the vast collection of information elements (i.e. entities and their relationships) derived by the **Blue** Layer beneath. The resulting aggregation of these elements forms a very large, distributed knowledge graph that provides a real-time representation of the problem domain, originating from the **Red** Layer source data below. Analytic processes within the **Black** Layer access and augment the knowledge graph continuously to aid current understanding and the prediction of future conditions. Users processes within the **Green** Layer above continuously monitor this knowledge graph for real-time situational awareness, decision support, and strategic planning applications.

An overview of the **Black** Layer design is shown below in Figure 9. The primary **Black** Layer components include:

- Knowledge Graph
- Knowledge Model (Ontology)
- Application Programming Interface (API)
- Analytic Engines
- Analytics

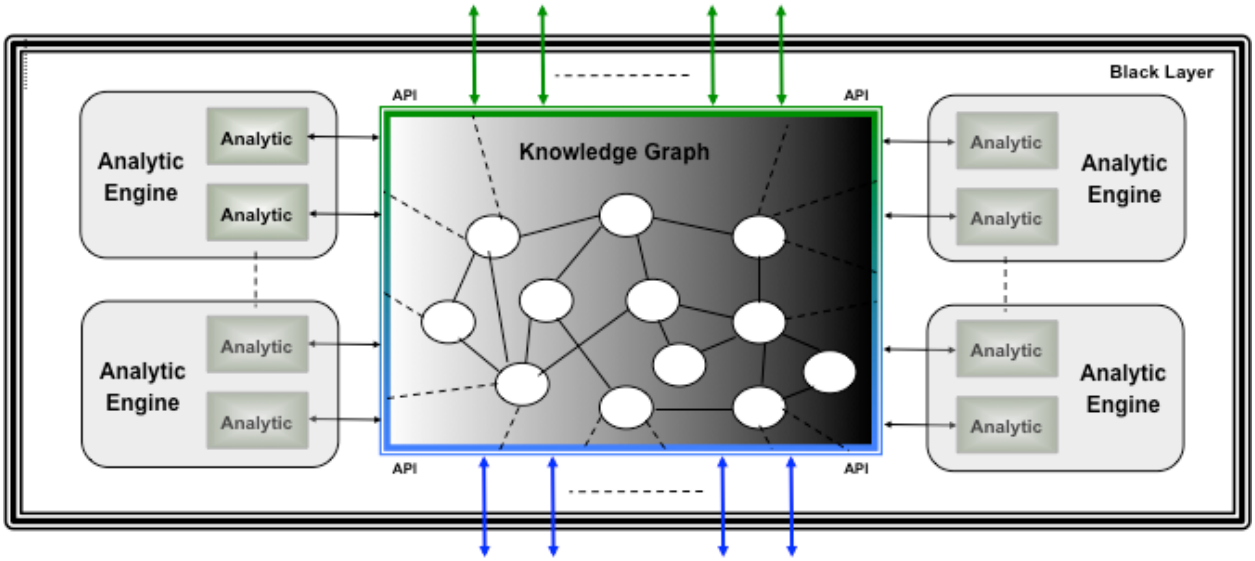

Figure 9. The **Black** Layer

The Knowledge Graph

At the core of the **Black** Layer is the knowledge graph. The knowledge graph is a dynamic and distributed structure for organizing and accessing the vast, aggregate collection of the information elements that are each separately managed within the **Blue** Layer. The knowledge graph manifests as a result of the many **Blue** Layer adapters interoperating via the same interface protocol (i.e. The **Black** Layer API). Conceptually, the knowledge graph is perhaps easiest viewed as a very large attributed node-link model where nodes represent entities (e.g. people, places, things) and links represent relationships between entities (e.g. "belongs to", "member of", "communicates with"). Affixed to each of these nodes and links are collections of attributes that further characterize each node or link's current state or condition. Nodes and links (entities and relationships) are collectively "deposited" (aggregated) by the **Blue** Layer adapters into this single shared (but decentralized) graph space. Conforming to the same graph space protocol (i.e. the API), the **Blue** Layer adapters operate collaboratively to populate this space. With many adapters working in unison over numerous large **Red** Layer data sets, the knowledge graph grows increasing large with the upper bound limited by the population size of the respective entity and relationship types. In this manner, the knowledge graph emerges as the critical tool for shared capture and study of complex relationships between both individual and large groups of entities interacting throughout the world.

To aid research across a diverse range of sociopolitical and technological problem domains, the 4CF knowledge graph structure was specifically designed for global scalability with graph sizes reaching upwards of one trillion nodes and one quadrillion links. A fully realized knowledge graph of this magnitude could quickly catapult the requirements of a single physical computing facility well into the advanced supercomputing computing realm. Consequently, the 4CF **Black** Layer was designed for distributed operation where the graph need not ever be physically instantiated in any one single location. Rather, **Black** Layer analytics need only work with those portions of the knowledge graph that are relevant to their specific line of inquiry and supported by their specific underlying infrastructure. The 4CF knowledge graph was conceived therefore with a pragmatic *crawl-walk-run* evolutionary path. While the 4CF was designed with scaling to the sizes mentioned, realizations of this architecture need only start with a small number of **Red** systems to receive benefit, incrementally adding other of **Red** Layer systems, **Blue** Layer adapters, and **Black** Layer analytic processing capabilities as requirements and resources dictate.

To achieve its scaling goals, the knowledge graph implementation employs a powerful computational science technique based on the concept of an "overlay". Perhaps the most common example of an overlay familiar to all is the World Wide Web. Using a typical browser application (e.g. Internet Explorer, Safari, Firefox, etc.), a web user today is presented with what appears to be a (mostly) seamless space of interconnected web pages. The 4CF leverages this construct presenting a client application with an analogous seamless space, but one of interconnected graph elements instead of

webpages. Just as a web user generally has little interest in the precise physical storage location of a specific page, the 4CF client application need not be concerned with the precise location of a specific graph element. The knowledge graph interface handles this all transparently. To accomplish this transparency, the knowledge graph is implemented predominately using "pointers". That is, when accessing a node or link in the knowledge graph, an application is actually referencing a pointer to an associated **Blue** Layer adapter that realizes that graph element. In this manner, the 4CF knowledge graph contains no actual data and thus there is no inherent need for data replication or centralized data storage. Requests of specific nodes and links and their attributes are automatically routed to the appropriate adapter(s) that support them. Furthermore, the knowledge graph itself need not ever physically resides in any one location (except when cached for specific performance reasons). Rather, the graph is highly distributed throughout a decentralized 4CF enterprise. The knowledge graph thus manifests as a result of the **Blue** Layer adapters and **Black** Layer API, much akin to webservers and the HTTP protocol in the World Wide Web.

For further details regarding the knowledge graph implementation, see the FOUR-Color Framework: *Implementation Guide.*

The Knowledge Model

While separable from the knowledge graph machinery, the knowledge model is arguably the single most critical design component of any 4CF instantiation. In order for the knowledge graph to be of analytic value, the nodes and links within the graph must be encoded using a model that describes their semantics. A critical design decision for analytics systems is whether such model should be problem specific or problem general. A problem specific choice offers the potential opportunity for increased performance gains, as the implementation can be carefully tuned leveraging domain-specific insight. However, this choice often renders the resulting implementation "brittle" to changing problem requirements and can hinder new data/analytic innovations and opportunities. To address this balance, the 4CF was designed to be "ontologically neutral". That is, the 4CF itself does not prescribe any specific domain semantics (i.e. no single ontology) for its knowledge graph. Rather, the 4CF maintains a separation between the knowledge graph and its semantics, enabling each to be dovetailed, but to exist independently. In this manner, the 4CF is capable of supporting a wide variety of diverse analytic applications across numerous sociopolitical and technological domains. This allows multiple analytic use cases to be simultaneously pursued, leveraging the same 4CF infrastructure.

While the 4CF supports multiple concurrent ontologies, a substantial analytic benefit can be gained nevertheless by employing a unified ontology that spans the collective domain set. Integrating multiple domain-specific ontologies into a single ontological framework offers considerable opportunities for achieving higher knowledge densities, thereby potentially enabling much greater analytic inference potential and yield. The 4CF thus advocates the development of a single knowledge model that is common

across all 4CF knowledge graph instantiations, thereby further ensuring analytic interoperability and ultimately the greatest degree of analytic power. Development of such a single knowledge model, however, is a unique challenge requiring a careful balance between analytic expressive power and implementation performance. That is, overly expressive knowledge model choices can lead to significant performance impacts. Similarly, poor knowledge model expressive power can lead to limited analytic utility. Knowledge model design therefore is of critical importance to 4CF enterprise management, ultimately dictating its complexity and operating efficiency.

Technically, the knowledge model is a formal agreement that precisely defines the semantics associated with every node and link in the knowledge graph. At a minimum, this model includes a taxonomy of the allowable node, link, and the associated attribute types that may be instantiated in the graph. Every **Blue** Layer adapter requires this type information in order to know how it is to represent the entities and relationships that is transforms from **Red** Layer source data systems. Similarly, **Black** Layer analytics and **Green** Layer applications require this same information to know how to interpret the elements contained in the knowledge graph. In addition to the type taxonomies, the knowledge model also contains information that describes how each of these types is semantically related to each other. The 4CF does not prescribe any specific tool or technique for designing and managing the resulting ontology, but it does advocate that the ontology be expressible in graph form so that it too may be represented in the knowledge graph. This is ultimately important for advanced analytics and machine-learning applications that employ automated inference techniques.

While the 4CF advocates a unified knowledge model across all 4CF domain instantiations, it recognizes that such a model will likely need to be very dynamic. At increasing scale with greater numbers of **Red** Layer data sources being integrated, it is expected that additional node and link types will need to be regularly added. The 4CF recommends that this be accomplished in an extensible fashion where new additions and refinements can be made without invalidating previous knowledge model assignments. It is assumed, therefore, that knowledge graph type taxonomies are represented via a range of fully enumerated values with new types assigned new unique values and retired types deprecated, but never fully deleted from the taxonomy. Data attributes affixed to nodes and links are name/value pairs and handled in the same fashion. The list of valid attribute names is extensible, but the syntax and semantics for any set of attributes with the same name should be identical. While the 4CF implementation is fundamentally decentralized, management of the knowledge graph model/ontology is assumed to be a centralized activity with either a single organization or representative committee responsible for its development and evolution.

For further details regarding knowledge model development, see the FOUR-Color Framework: *Application Guide.*

The Application Programming Interface (API)

The primary unifying construct of the 4CF that binds all **Blue**, **Black,** **Green** Layer components is the 4CF **Black** Layer API, or simply the API. This interface was carefully crafted to be very simple, yet provide a single, common, powerful mechanism for all software components to access the **Black** Layer knowledge graph throughout a 4CF enterprise. As envisioned, the API is supported by a diverse range of programming languages (e.g. Java, Python, Ruby) and statistical/analytical tools (e.g. SAS, R, MATLAB). This is accomplished through a set of language and tool "bindings" that provide direct access to the API primitives from the native programming or analytic environment.

The power of the API is derived from its generality. The API provides two basic interface patterns: graphs and events. Graphs are the collections of node and link elements (entities and relationships) and their associated methods (attributes and actions) that comprise the 4CF knowledge graph. An event represents something that happens to one or more graph elements. These two patterns combine to yield an elegant set of primitives for general-purpose, scalable, distributed systems integration.

A client application, regardless of the layer within it resides, has access to the knowledge graph via the API. Within the limits of the application's authorization/access permissions, a client may *create* or *delete* an element of the knowledge graph. Creating a knowledge graph element involves the assignment of a universal unique identifier (UUID) so that any other client in the 4CF enterprise (with the proper authorization/permissions) can access that element. Deleting a knowledge graph element releases the element's UUID so that it can never again be accessed.

For an element to have any behaviors (e.g. attributes), that element must first be connected to one or more **Blue** Layer adapters. The API *connect* operation performs that function. The 4CF enables knowledge graph element behaviors to be dynamic, changing over time as adapters are updated, new adapters are brought online, and old adapters are retired. The API *disconnect* operation aids this process, disconnecting a knowledge graph element from an adapter. The *connect* and *disconnect* operations accept a knowledge graph element UUID and a **Blue** Layer adapter address as parameters.

The most frequently used operation of the API is *invoke*. This operation allows both attributes of knowledge graph elements to be accessed and actions on knowledge graph elements to be performed. The core parameters required for all *invoke* operations are the UUID of the knowledge graph element and the name of the "method" (i.e. action to be executed, or attribute to be accessed). For cyber and privacy concerns, the 4CF requires that method names be fully enumerated as specified by the knowledge model taxonomy. That is, the list of valid method names is intended to be centrally managed. While this list can be extended, it is specifically designed to preclude dynamic modification without coordination across the 4CF enterprise. This static, but updatable

nature is important for the formal proof-of-correctness processes needed for privacy and security isolation. Depending upon the method specified, the invoke operation may allow additional parameters, but these items are again subject to formal specification in advance, precisely detailing their type, length, and acceptable data value ranges.

The final two 4CF operators are *publish* and *subscribe*. The *publish* operation is used to notify other processes throughout the 4CF of events associated with graph elements. Accepted parameters include the UUID of the graph element and the name of the event being published. The 4CF requires that all events be associated with a graph element and that the list of acceptable event types is again fully enumerated, compliant with an event taxonomy specified in the knowledge model. Specific event types may allow additional parameters to be included, but these must again be fully specified in advance, similar to method parameters. The *subscribe* operation is used by processes throughout a 4CF enterprise to receive notification of events that are published on graph elements. All event subscriptions must be associated with a graph element. Each API implementation provides a mechanism for notifying the requesting process when such events are published. This is typically accomplished via a software "callback" mechanism particular to the specific programming language binding and its associated run-time environment. The format of all event notifications, however, must again be fully enumerated/specified so that all processes throughout the 4CF enterprise are assured a common structure to aid proof-of-correctness.

For further details regarding knowledge model development, see the FOUR-Color Framework: *Application Program Interfaces (APIs).*

Analytic Engines

The 4CF recognizes that analytic performance can vary significantly across various computational platforms depending up the specific nature of the problem and the available information. The 4CF was thus formulated to leverage a wide range of diverse analytic platforms ranging from simple software packages to large-scale customized hardware solutions. The 4CF prescribes no single optimal platform. Rather, it advocates the single, common API whose implementation can be optimized for each specific platform. While the interface specification is identical for all analytics, the implementation of this interface can vary considerably in order to exploit the unique performance features of the hardware and software that it runs upon for various classes of analytics. As a result, not all analytics will perform equally well on any given analytic engine. On the contrary, some analytics may be restricted to certain engines depending upon the size and computational structure of the problem. However, the performance results of that analytic may perform several orders of magnitude faster or operate over a substantially large segments of the knowledge graph, in some cases perhaps up to and including the graph's entirety.

In general, the 4CF enables significant performance gains to be achieved by employing varying degrees of knowledge space caching and pre-caching, relationship pre-joining, and value pre-computation. That is, as the knowledge graph is populated, analytic engines may often compute fields gradually over time so that if and when ever needed, the required values may be available in near real or constant time versus computing them upon demand. While this may well involve computing and storing many items that potentially may never be accessed, these computational inefficiencies can be spread out over a wide duration in order to minimize the amount of compute power required when actually needed. Analytic engine design and configuration thus involves a complex balance between analytic performance, CPU utilization, storage, communication bandwidth, and memory. The 4CF recognizes that optimization of these factors is problem-dependent, often varying widely across differing analytic platforms. The diversity of platforms include batch, relational, columnar, transaction-oriented, matrix-based, graph-based, vector-based, tuple-based, index, map-reduce, and numerous statistical systems and packages, each exhibiting various pros and cons depending upon the specific analytic problem at hand. Via the single API, the 4CF enables the power of each of these to be incorporated without transferring the burden of their unique features onto the analytic developer. Thus, a growing suite of analytic engines with various accompanying API bindings/implementations is envisioned as a 4CF enterprise increasingly expands in scale.

For further details regarding analytic engine development, see the FOUR-Color Framework: *Implementation Guide.*

Analytics

The final and arguably most important elements of the **Black** Layer are the analytics themselves. Analytics are calculations, algorithms, models, functions, simulations, etc. that jointly accesses the **Black** Layer knowledge graph. In a large-scale 4CF enterprise, the number of analytics envisioned could easily grow into the thousands and far beyond. Each analytic interacts with the knowledge graph via the API. An analytic may *consume* information that already exists within the graph, it may *produce* new information to be incorporated into the graph, or it may do a blend of both. Analytics interoperate with each other via the API and the knowledge graph that the API exposes. Using the API event publication and subscriptions operations, analytics can synchronize their operations to work collaboratively across a 4CF enterprise. While the API is common to all analytics, each may often exploit the unique strengths of the analytic engine upon which they run to best perform their functions. The distinct types of analytics supported by the 4CF is enormous and open-ended, limited mainly by the contents of the knowledge graph and the features of its analytic engine beneath.

The 4CF envisions a very diverse, vibrant, and dynamic ecosystem of analytics gradually emerging over time in a growing enterprise. Some analytics may be developed to work with small sections of the knowledge graph, while others may operate over the whole. Each analytic will likely undergo its own unique lifecycle. Large, complex analytics developed for a specfic purpose may eventually be decomposed into smaller, more modular analytics over time, while

other may be integrated and generalized for greater applicability. The 4CF imposes no particular constraints on shape, form, or function of each. The API and the accompanying knowledge model (ontology) that formally describes the knowledge graph contents are the only two 4CF components common to all within a single 4CF enterprise. In this manner, the 4CF is intended to foster analytic creativity and innovation, but with a deliberate organization to best ensure interoperability, security, and privacy.

For further details regarding analytic engine development, see the FOUR-Color Framework: *Application Guide.*

Layer 4 (Green) – Visualization and Decision Support

As the highest layer in the 4CF, the **Green** Layer is responsible for user interactions including visualization, situational awareness, alerting, decision support, and command and control applications. The **Green** Layer is the interface between the **Black** Layer analytics systems and the operators, researchers, decision leaders, and policy makers. The **Green** Layer accomplishes its mission by providing a suite of tools and services for browsing the knowledge graph, viewing the results of analytics, accessing and responding to incidents and exception conditions, modifying operating parameters and performance characteristics, and evaluating alternatives and mitigation options. As shown in Figure 10, these tools and services are grouped into three distinct, but related components:

- Real-Time Response
- Tactical Analysis
- Strategic Planning

The Real-Time Response component is responsible handling events and initiating actions on the time scale of seconds to minutes. The Tactical Analysis component focuses on analytic issues ranging from hours to days. The Strategic Planning component emphasis is on prediction and simulation in the timeframe of months and beyond. Each component receives and accesses knowledge and events from the **Black** Layer below via the API. The **Green** Layer components use this knowledge to provide status, make recommendations, and/or initiate actions that are appropriate for their respective time scale of operation. In addition, each of these components may interact with other components throughout the layer for local, regional, national, or global collaboration purposes.

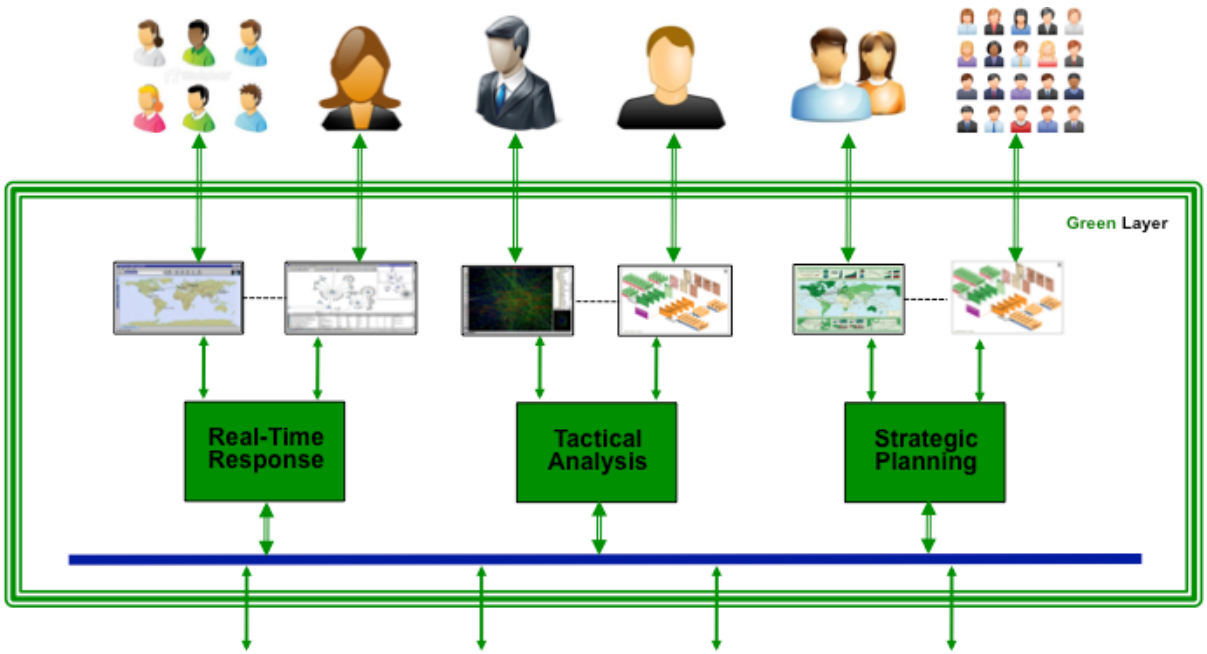

Figure 10. The **Green** Layer

The Green Layer Mission

The **Green** Layer provides support to the user community to help address four fundamental questions spanning each application domain.

Q1. **What is happening?** Intelligent filtering of information, appropriate visualizations to help interpret and coalesce the information, and support for understanding the timeliness and accuracy of the information are essential functions of the **Green** Layer.

Q2. **What can be done?** The **Green** Layer is responsible for presenting researchers, operators, decision makers, etc. with potential courses of action that may be available, consistent with policy.

Q3. **What will happen?** Once a potential course of action is selected the **Green** Layer provides simulations and projections of the near-term effect of an action. To the extent possible, these should support reasonable estimation of effects and potential consequences.

Q4. **How much can I trust what I am being told?** Information available to the **Green** Layer will come from multiple sources beneath, each with varying levels of provenance and trust, and it will not be uncommon for information to be contradictory. To enable human analysts to properly weigh possible courses of action trust relationships and veracity scoring should be employed and the results and alternatives presented to the user.

These four fundamental questions establish the context of the "core mission" of the **Green** Layer – to provide situational awareness to the users. The classic definition of situational awareness incorporates three major components, each of which the **Green** Layer must provide:

1. **Perception** (of the elements in the environment) - the capability to identify and retrieve the information that is necessary to solve the problem at hand.
2. **Comprehension** (of the current situation) – the capability to view and manipulate the data in order to come to an understanding of the problem at hand.
3. **Projection** (of future status) – the capability to select a course of action that will solve the problem at hand with acceptable consequences.

In providing these three capabilities to the user, a host of different tools are needed that users can combine in a wide variety of fashions depending on the particular problem under consideration. To solve one problem, the user may need a certain set of Perception, Comprehension, and Projection tools, but may need a complete different set of tools to solve another problem. Ideally, the user should have a "toolkit" from which he/she can select the appropriate tools for the problem at hand. It is expected that several perspectives and user roles will develop and evolve within the **Green** Layer operations.

Figure 11 shows the expected user roles and frame perspectives on which users working with the **Green** Layer are focused. Within each, there are analysts, analytic writers, policy staff, simulation staff, etc, each targeted at the appropriate level of concern. Each of the three roles interact with the **Black** Layer system in slightly different ways, leveraging differing sets of analytics designed to support their individual role.

It is important to note that this type of approach to giving the user a myriad of tools from which to choose is best based upon a thorough analysis process that determines the needs of specific users. This process of role-based analysis is undertaken to identify the specific tools needed by each user. The 4CF prescribes a "Usability Determination" be made early in each domain development life cycle. Each of these user roles is constrained by the **Black** Layer knowledge that they can access and manipulate.

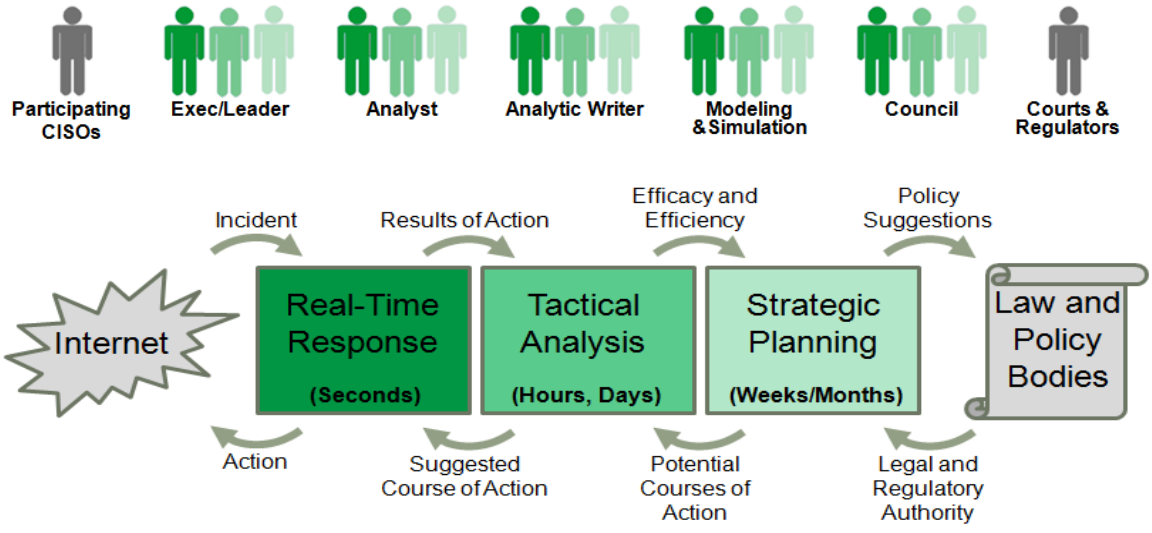

Figure 11. User Roles and Perspectives within the **Green** Layer

The Green Layer Functions

The **Green** Layer is ultimately intended to be the richest, most diverse, and inevitably the most complex of all the 4CF layers. The **Green** Layer is the primary service layer that interacts directly with the user community (responders, system operators, analysts, policy leadership, etc.). These individuals are tasked with an enormous range of activities including assessing situations, making decisions, taking action, and initiating and managing responses. These critical decision support roles involve acquiring knowledge from the **Black** Layer, visualizing that knowledge, and simulating and predicting future system state. In support of these diverse activities, the **Green** Layer application set is purposefully intended to be open-ended, likely ever changing and expanding. It may be tailored for local purposes, or generalized for global applicability. The 4CF provides no particular set of constraints on the **Green** Layer, yet encourages components throughout a 4CF enterprise to interact by acquiring, contributing, and exchanging knowledge via the **Black** Layer and its associated API. Thus, the total collection of the functions involved at this level is similarly open-ended due to the nature and complexity of issues that spanning the many potential 4CF application domains. In general, these functions can be grouped into the following major topic areas:

- Situational Awareness
- Decision Support
- Visualization
- Sense-Making
- Hypothesis Generation
- Simulation and Prediction
- Incident Annunciation and Prioritization
- Response Generation and Coordination
- User Input Management
- Command and Control
- Analyst Collaboration
- Policy Analysis and Compliance
- Ontology Analysis and Input
- Risk Analysis and Management
- Communication
- Health and Status Monitoring

The **Green** Layer performs these functions by leveraging the knowledge graph and the privacy-compliant analytic processing provided and enforced by the **Black** Layer beneath. The actions requested by operational personal flow back into the **Black** Layer and, if compliant with policy and law, are processed and ultimately flow out to the local **Red** systems via the trusted distribution and security/privacy isolation capabilities of the **Blue** Layer.

The interface between operators/users and **Green** Layer, and between the **Green** Layer and **Black** Layer can be elaborated in terms of expected inputs and outputs. These flows are illustrated in Figure 12 and Figure 13, respectively.

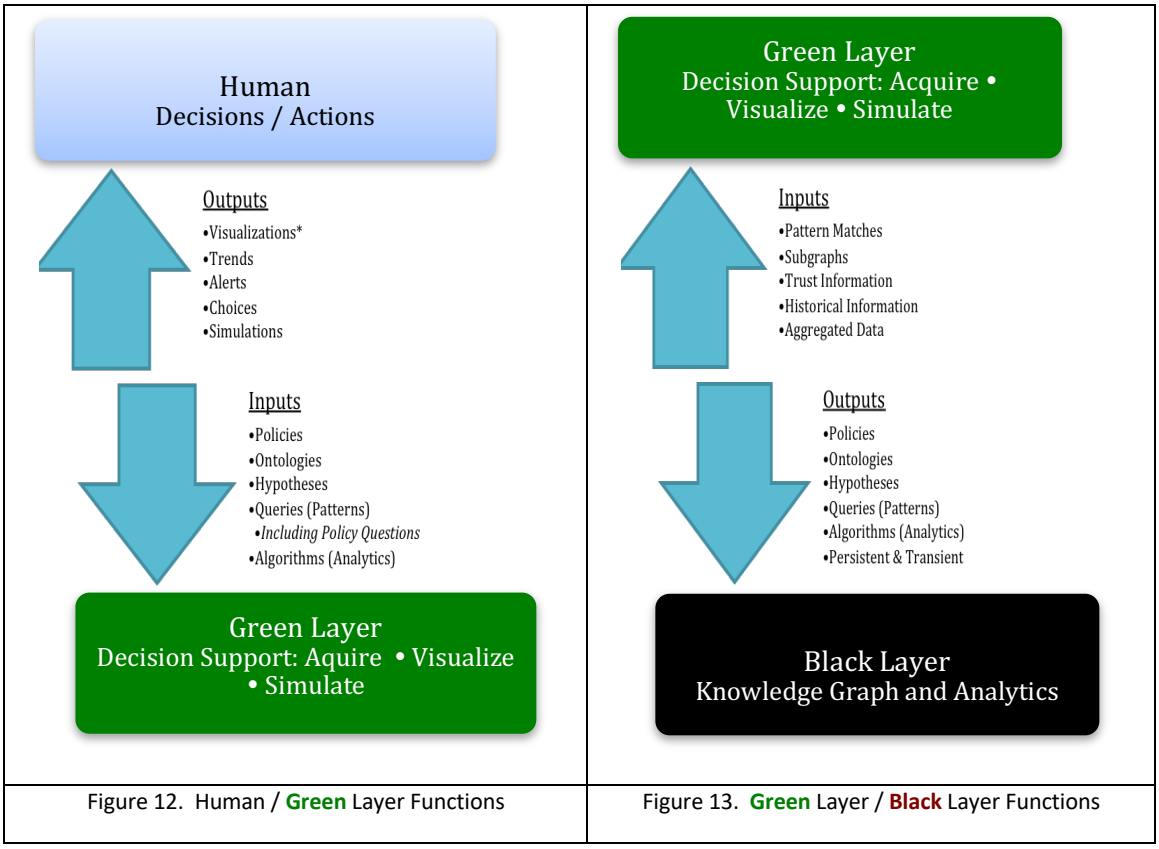

Figure 12. Human / **Green** Layer Functions

Figure 13. **Green** Layer / **Black** Layer Functions

The **Green** Layer tools provide the visual analytics that allow analysts to determine whether anomalies constitute reportable incidents and whether these incidents may pose a specific threats or require a specific response. Using these tools, analysts interact with the **Black** Layer to assess a situation, make a specific decision, ascertain a course of action, and then initiate the necessary response. To encourage the widest and richest set of possible solutions, all major internal **Green** Layer interfaces are encouraged to be published and available to the broad community of tool developers.

Ultimately the **Green** Layer must satisfy an as-yet unknown set of analytic requirements. The 4CF anticipates that new capabilities will need to be regularly introduced, which will necessarily create new user roles and workflows. The requirements for the **Green** Layer will consequently evolve continuously. Users will require new, and often unique, tools and methods, and thus the **Green** Layer should properly be seen as an application framework, along with a common set of applications and libraries for visualization, understanding, simulation, and action. These libraries and applications will, of necessity, need to be integrated with third-party tools and libraries. This must be done in manner that prevents third-party components from contaminating the knowledge, security, and privacy of the **Black** Layer. While some **Green** Layer services may be considered within a trusted boundary, when properly verified, validated, and certified, others may not conform to this same rigor. Thus, **Blue** Layer isolation techniques may be required for these components to ensure **Black** Layer integrity. In essence, the 4CF provides a mechanism for select **Green** Layer components to behave as **Red** Layer components, and visa versa, but under carefully controlled circumstances. This curious and seeming duplicity is a unique and powerful feature of the 4CF, enabling the integration of very diverse, highly

distributed components without the loss of the security and privacy protections awarded by the **Black** Layer with **Blue** Layer isolation. This flexibility is important to accommodate the wide variations in trust, authority, and implementation of the constituent platform systems.

Summary

The four architectural layers of the 4CF serve very specific and deliberate functions to achieve the scaling, resiliency, and policy enforcement for a workable, acceptable, and effective solution for extreme-scale information sharing and analysis. The 4CF provides a comprehensive, integrated systems vision for global scale collaboration and analytic interoperability with strong security isolation and privacy enforcement. The 4CF prescribes a distributed knowledge resource that ultimately could emerge as a national or international asset for addressing some of the most difficult sociopolitical and technical challenges that must be faced.

The analytic community today is very vibrant with many new and exciting ground-breaking techniques and technologies continuously emerging to help address difficult local, regional, and global problems. The 4CF was conceived to leverage these in a very flexible, interoperable manner to help achieve three main goals:

1. Information sharing: exchange of information from local to global scale to achieve the highest levels of knowledge density while enforcing security and assuring privacy.
2. Analytic interoperability: integration of analytic processing across a very diverse and highly distributed data/information enterprise.
3. Situational awareness: a unified operating picture of complex sociopolitical systems at increasing scale from local to global levels.

The world is a very complex place. The FOUR-Color Framework offers a very serious approach for analyzing and understanding that complexity.