



**The
Data
Administration
Newsletter**

Since 1997

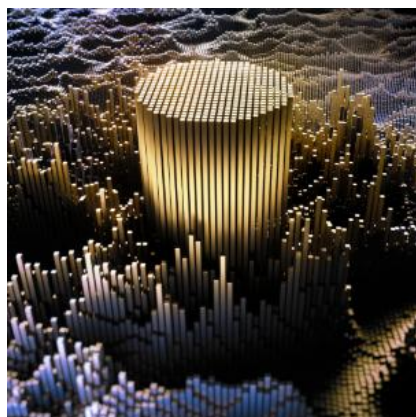
Published by Robert S. Seiner

[Home](#)
[Articles](#)
[Columns](#)
[Features](#)
[Blogs](#)
[News](#)
[Events](#)
[Resources ▾](#)
[Archives](#)
[Strategy](#)
[Governance](#)
[Architecture](#)
[Data Mgmt](#)
[Quality](#)
[Modeling](#)
[Operations](#)
[Metadata](#)
[MDM](#)
[BI](#)
[Applications](#)
[Big Data](#)
[Subscribe to The Data Administration Newsletter Today - it's FREE!](#)
[REGISTER HERE!](#)

[Home](#) / [Data Topics](#) / [Business Intelligence News, Articles, & Education](#) / [Business Intelligence Articles](#) / [Principles for the Data-Centric Paradigm – Part 2](#)

Principles for the Data-Centric Paradigm – Part 2

Posted on October 2, 2019 by Derek Coursen



There is a movement to upend traditional thinking about information systems by putting data and meaning at the center of strategy, architecture, and system development sequencing.

Establishing a complete data-centric paradigm will require foundational principles which underscore the clear break with application-centric thinking, and which establish common ground on the unresolved issues inherited from the history of data modeling.

Three principles which can do this are: First, a good semantic core within the information system is a truthfully designed model of a portion of the real world outside it. Second, a good semantic core transcends functional requirements. Third, a good semantic core engages critically with the affordances of business language.

1. The need for foundational principles

The preceding article, [The Unfolding of the Data-Centric Paradigm](#), described the application-centric paradigm, the habit of viewing the information system as a set of functionalities which happen to need the support of data structures. For decades, various data-centric practices have attempted to remedy the shortcomings of that approach. Today a new data-centric movement—expressed in [the Data-Centric Manifesto](#) [11] and [the Data Doctrine](#) [1]—is challenging the application-centric paradigm more comprehensively than in the past.

The data-centric movement of today has a new strength: it can focus on meaning and the possibility of ontologies that disentangle semantics from structure and technology. Yet it still faces the headwinds of entrenched ways of thought which most information systems practitioners do not notice or question. And the lack of solid foundations for data modeling theory, which were the downfall of past data-centric practices, must still be resolved.

As the Data-Centric Manifesto puts it, "...the main barrier to changing [the application-centric] paradigm is not technical, but mental and inertial." [11] Some of the mental barriers come from the dominance of that paradigm, while others come from the history of data-centric approaches.

Most information systems practitioners—including many of those who primarily work with data and even with data models—have been trained and immersed in the application-centric paradigm for their entire careers. Focusing on functionality first has become second nature. This socially reinforced way of seeing makes it difficult for practitioners to learn to work in data-centric ways. At the same time, there's a lingering hangover from earlier data-centric practices. Advocating for new ones can provoke the automatic response that previous data-centric efforts stumbled decades ago.

The solution is to declare and develop a new paradigm. The application-centric paradigm is bankrupt and must be replaced. But replacing one paradigm with another requires articulating principles which sharply and publicly distinguish the new from the old. Those principles must encapsulate and justify the data-centric way of seeing and designing information systems. They must highlight the limitations of application-centric approaches. They must represent a common denominator of understanding that is sufficient to unite people who currently profess divergent data-centric positions while attracting others who so far do not. They must provide methodological support for the practical work of designing core semantic models that can organize the data of entire enterprises.

To do all that, the principles must establish common ground on the unresolved issues inherited from the history of data modeling. Those can be boiled down and rephrased as questions about the semantic core of an information system, meaning the logical arrangement of concepts into entity types, relationships between them, attribute types, and taxonomies

Search TDAN.com

Search ...	Search
------------	--------

Follow TDAN.com:



Advertisements

that constrain attributes' values. (The term *semantic core* is here used to refer to the formal ontologies that could organize information systems of the future or, equally well, to the data models and associated taxonomies of existing information systems built on current platforms.) The questions are:

How does the semantic core of an information system relate to the real world, to functional requirements, and to the language by which the business describes itself?

And what characteristics comprise high quality in the semantic core of an information system?

Three Proposed Principles

The following principles offer a data-centric basis for information systems:

1. *A good semantic core within the information system is a truthfully designed model of a portion of the real world outside it.*
2. *A good semantic core transcends functional requirements.*
3. *A good semantic core engages critically with the affordances of business language.*

The adjective *good* in each principle points to virtues which may or may not be present, to varying degrees, in the semantic core of any particular information system. The principles are therefore both descriptive and prescriptive.

2. A good semantic core within the information system is a truthfully designed model of a portion of the real world outside it.

At first glance this is reminiscent of a commonplace from data modeling theory. It is phrased, however, to assert a position that goes considerably beyond what most information system practitioners believe. The differences are in locating the model of the real world *within the information system* and in the phrase *truthfully designed*, which relates the semantic core to philosophical accounts of knowledge and to differences between activities of description and design.

Before exploring what this statement asserts, it is necessary to clear away some brush. The idea of a relationship between a data model and the real world has always been subtly controversial. A large proportion of information system practitioners are likely to disagree with it, at least partially, for a variety of reasons. That resistance needs to be understood because it impedes adoption of a data-centric mindset.

2.1 Skepticism about modeling the real world

It is undisputed that at the level of granular facts, an information system is supposed to be a model of the real world. Business stakeholders depend on information systems to receive input and produce output in the form of determinate and structurally homogeneous sentences. For example, *the car having license plate 1234 is a Mazda* and *the car having license plate 5678 is a Fiat*. They consider the information system trustworthy insofar as those input/output sentences do in fact correspond to verifiable perceptions in real life.

That level corresponds to what Michael Jackson posits as the lowest common denominator of phenomenology: *facts* about *individuals* (i.e. discrete instances of any type of object). Truths about the world such as *the car having license plate 1234 is a Mazda* are simpler than assertions about many facts such as *every car has a license plate number and a brand*. [8] Facts at that simple level can be expressed in propositional logic. In database theory, that simple level corresponds to what ISO/TC97/SC5/WG3 called the *information base*: "The description of the specific objects... that in a specific instant or period in time are perceived to exist in the universe of discourse and their actual states of affairs that are of interest." [5]

Constructs for organizing data, however, generalize granular facts into abstract assertions using entity types, attribute types, relationships, and taxonomies. And for a variety of reasons, many people are unwilling to treat that abstract level as a model of the real world.

This doubt has existed since the early days of database design standards. The ISO/TC97/SC5/WG3 built its work around the firm position that the *conceptual schema* within the information system is a set of predicate logic sentences describing a portion of the real world, "the possible states of affairs of the Universe of Discourse including the classifications, rules, laws, etc." Yet the committee acknowledged that the debate on this fundamental was still ongoing: some theorists regarded the conceptual schema as (merely) describing the data in the information base. [5]

Two different meanings of the word *model* can obscure the nature of the opposed positions. To some, *model* means a representation referring back to the real world. To others, *model* just means a design plan for building something. Those are very different ways of thinking.

There are plenty of practitioners today who treat the data models they work with as simply plans for structuring data. Where does their reluctance to treat them as models of the real world come from?

2.2 Methodological and philosophical objections

The most basic and antagonistic objection comes from the idea that a data model exists in order to correctly and completely satisfy the functional requirements known for a particular application development project, and can be judged as successful or not on that basis. This position is close to the nucleus of the application-centric paradigm. Many practitioners—especially those in roles more concerned with business analysis than data structures—find the idea of modeling the real world to be strange and unnecessary, since they consider the world to be entirely mediated by the functional requirements.

A more sympathetic objection arises from the practice of progressing from conceptual to logical to physical data model. It's often taken for granted that the clarity of the original semantics necessarily degrades along the way. Many people might

Upcoming Events

- Data Architecture Summit 2019 - Tutorial - 10/15/2019
- RWDG Webinar: Build Your Own Data Governance Tools - 10/17/2019
- RWDG Webinar: How can Everybody be a Data Steward? - 11/21/2019
- RWDG Webinar: Utilize Governance Working Teams to Improve Data Quality - 12/19/2019

Quote of the Week

Our greatest weakness lies in giving up. The most certain way to succeed is always to try just one more time. -- Thomas A. Edison

Popular Posts



Universal Data Vault: Case Study in Combining "Universal" Data Model Patterns with Data Vault Architecture – Part 1
November 1, 2015



Data Modeling is Data Governance
August 1, 2015



Data Warehouse Design – Inmon versus Kimball
September 1, 2016

Tags

agile big data data data governance data management data modeling data quality Metadata non-invasive data governance seiner

agree that a conceptual artifact for initial planning can be a meaningful model of the real world, yet would be skeptical about making the same claim for the semantic core implemented within the information system. Common experience reinforces this doubt. Textbooks offer many good examples of data models that aspire to represent portions of the real world, but the practice of allowing semantics to degrade during implementation means that in real life, many practitioners have only encountered databases that look like persistence engines for specified processes.

Philosophical objections are also widespread. Ever since William Kent's *Data and Reality* [9], thoughtful modelers have been aware of subjective aspects of their work. Rudy Hirschheim et al point out that there are no unambiguous formal criteria to guide the model builder in deciding what real world things to include in the model, or how to group them into entity types. That is only the thin edge of a much thicker wedge of epistemological objections to the idea of a data model being an objective description of the real world. [7] Graeme Simsion's study of leading modelers, framed in terms of a polarity between description versus design, reveals how these issues play out in practice. [14]

Subjectivity makes many practitioners queasy. In a discipline often thought of as a branch of engineering, there's a strong preference for the determinate. A functional requirement can be treated as determinate, and so can a granular fact. A requirement for the input and output of particular types of granular facts seems satisfyingly objective. A data model, by contrast, is imbued with human subjectivity, and the process of creating it is hard to pin down. That leads many people to discount it as a model of the real world.

2.3 The experience of a truthfully designed model

Although the design of a semantic core necessarily has a subjective dimension, that need not prevent it from being a truthful model of the real world. Truth must not be confused with objectivity. Several kinds of truth that relate to the real world may be present in a data model. Working from philosophical accounts of what *truth* means, John Artz outlines two: *correspondence* and *coherence*. [2] Yet another kind of truth present in many modelers' work can be called *theory confirmation*.

Correspondence is the most familiar. It means that in order for a statement to be true, it must correspond to real-world things or events. For example, transposing this criterion into a relational database environment, it means that tables represent real world entity types, foreign keys represent regular relationships between them, etc. "The strength of correspondence is that when a user asks a question of the database, the answer he or she gets is not just a calculation in the database, but it is also true in the real world." [2]

Insofar as queries can employ terminology (names of entity and attribute types and taxonomic values) mirroring real world phenomena, they can be as complex and innovative—within the scope of the model—as a person's natural questions about the real world might be. Acknowledging that the logical arrangement of concepts involved a degree of subjectivity by the designer does not diminish the truthfulness of the correspondence between the user's query and the (perceived) states of affairs in the real world.

Though correspondence may seem obvious, a high proportion of existing information systems fail to attain that level of truth. There are often record types which are difficult to relate to any real world entity, because they are merely based on how information is processed. Artz describes these as representing a naive sort of truth in which a person has heard something somewhere and not encountered any evidence to the contrary. [2] Data structures of that sort, though, do not rise to the level of a truthfully designed model of the real world; the designer was not focused on the real world at all, but on the task of transforming input into output.

Coherence is a different kind of truth. It means that phenomena make sense to the observer. People make a chaotic and messy world coherent by organizing it into categories and relationships that they create. Artz points out that the meaning within a business domain may be so muddled that the modeler must superimpose order upon it, cleaning up the semantics by replacing terminology which, in its original form, is too ambiguous to be the basis for a model. [2]

Imposing coherence that was not originally present in the business language will clarify the environment for its stakeholders. Users will be able to query the information system and find correspondence between the facts it presents and their reorganized perception of the real world. (A detailed example of imposing coherence on muddled semantics can be found in the framework, created by this author and a colleague, for modeling participant flows in human service programs. [4])

Theory confirmation is yet another kind of truth. As Alan Sokal and Jean Bricmont point out, scientific theories come to be accepted as true because of their successes in predicting new phenomena. [16] In his foreword to David C. Hay's *Data Model Patterns*, Richard Barker writes of "a more active form of modelling... commonly found in mathematics and science, which has a model predict something that was not previously known or provide for some circumstance that does not yet exist." [6] He was referring, in that context, to the practice of designing data structures around higher levels of abstraction than are explicitly present in the business domain. Modelers who work in that way will then find parts of their design either confirmed or disconfirmed, depending on whether or not those parts turn out to gracefully accommodate data as it arises in unforeseen circumstances—changes in the business environment, expansion of the model to further areas of it, or use of the model in a different business environment.

Designers of a semantic core work with all three kinds of truth. All of them involve human judgement which is difficult to specify. Len Silverston quotes a data modeler who, asked why he had chosen one model over another, explained: "I thought that there was a sense of truth to [it] that felt better." [14] Yet all these kinds of truth are necessarily tested against stakeholders' experience of phenomena in the business domain. In this practical pursuit of a truthful design, the dichotomy between objectivity and subjectivity and the polarity between description and design need cause no trouble. The attempt to truthfully describe the real world takes place within the framework of stakeholders' shared and evolving perceptions.

2.4 Implications

The idea of a *truthfully designed model of the real world* encapsulates characteristics that make a model useful and durable. It offers a touchstone for assessing the quality of models.

Locating the model *within the information system* affirms that it is possible and necessary to maintain, through the system implementation process, semantics referring to the real world; data structures must not be a mere persistence engine supporting defined functionality. As Dave McComb points out, the practice of translating from conceptual to logical to physical data models introduces complexity, and therefore costs, which have doomed many attempts at building enterprise data models; the way forward requires eliminating the difference between conceptual and implemented artifacts. [12]

These positions are necessary because the data-centric paradigm aims to create shared data stores that are permanent, while the applications that use them are understood as ephemeral. [11] The goal is stable, reusable data structures that precede application code. [1] Those must be based on something, but that something can no longer be the functional requirements of particular applications. The new paradigm thus implies a different relationship between the semantic core and functional requirements.

3. A good semantic core transcends functional requirements.

The concept of functional requirements has been central to the theory and practice of developing information systems. It organizes the social fact that an information system must be intended to do something(s) for some stakeholder(s). Functional requirements are traditionally treated as the axis around which a system must be designed.

This has led many information system practitioners—especially those in roles more concerned with business analysis than data structures—to assume that data models are subordinate to functional requirements. There is often a misconception in two parts: that the model flows directly from the functional requirements, and that the affordances of the model are bounded by the functional requirements.

A high proportion of existing data models do, unfortunately, reinforce those ideas. But a *good* semantic core transcends functional requirements in both aspects. Its components are based on insights beyond those present in functional requirements, and it offers affordances beyond those specified in functional requirements.

3.1 The poverty of the functional requirements

There is a stubborn popular notion that the data structures and semantics of an information system can somehow be inferred from functional requirements. For the success of the data-centric paradigm, all roles in the information systems profession need to come to understand why that is a fallacy.

A limitation of most functional requirements is that they describe neither the real world itself nor a model of it within the information system. Most requirements describe the system as a mechanism providing functionality at its interfaces with its environment. By avoiding description of what happens inside the mechanism, they attempt to avoid implementation bias. However, they usually don't offer a precise or holistic description of the environment (business domain) either; they tend to simply describe how the information system is supposed to act upon bits of it.

This commonplace practice has been subjected to thorough critiques. Pamela Zave and Michael Jackson argue that requirements are better stated purely in terms of the desired state of the environment, not the mechanism at all. [17] Yet their approach is rare in system development projects.

That being so, functional requirements are unlikely to contain all the raw concepts necessary for creating a model; and even if they did, the insights necessary to arrange them would still need to be provided by the modeler. This is obvious to modelers but not to other stakeholders.

The fallacy endures, though, because the convention of treating the information system as a black box has spilled over into a social convention: Database professionals are expected to know how data is organized inside it; other stakeholder groups—including business analysts and project managers—are usually encouraged to focus on its inputs and outputs. This socially imposed black box subliminally encourages stakeholders to assume that since they only discuss inputs and outputs, those must determine what's inside the box.

Something analogous occurred in the twentieth century debate about language learning. Psychologists such as B.F. Skinner, who were accustomed to constructing input/output models of human behavior, held that stimulus/response mechanisms could explain how people learn language. Noam Chomsky, however, showed the *poverty of the stimulus*, its inability to explain the capacities displayed by language. [10]

Similarly, the data-centric paradigm needs to assert the *poverty of the functional requirements*. The affordances of a good semantic core flow from its truthfully designed relationship with the real world, not from whatever functional requirements it may happen to satisfy.

3.2 Affordances exceed functional requirements

Conversely, functional requirements do not form a boundary around the affordances of a good semantic core. Each of the three kinds of truth—correspondence, coherence and theory confirmation—allows it to exceed requirements.

Most evidently, as long as the design has sufficient correspondence with the real world, it is unnecessary to specify every required output (e.g. report) in advance. From observing the entity and attribute types, relationships, and taxonomies, practitioners can accurately assess the gestalt of the system's possible outputs. Furthermore, when a data model imposes coherence on previously muddled semantics, that creates the potential for functionality that could not have been specified in the original requirements. And insofar as modeling is successful at positing a true theory about the real world, it accommodates future situations, unanticipated by the requirements, which confirm its theory.

3.3 An ontological style of modeling

This principle has strong implications for how modelers treat functional requirements in relation to the real world. Some approaches can be characterized as attempting to model based on requirements without much reference to the real world; others as starting from requirements but then looking *through* them to model the real world; and yet others as starting from more abstract representations of the real world and looking *through* them at the requirements.

Thus one person might read functional requirements, extract relevant terms, and draft a design that attempts to arrange them into the most literal possible entity and attribute types, relationships, and taxonomies that will satisfy the requirements. A second person might start in the same way, but then while designing transpose some of the terminology into more abstract categories that provide more satisfying levels of correspondence, coherence and theory about the world. A third person might skim the terminology looking for clues as to which known abstract architectural patterns are likely to be useful, and then start from those patterns, working backwards toward the specifics of the functional requirements.

These three approaches correspond closely to groups of opinion that Graeme Simsion et al identified when studying leading data modelers: literalists, moderate abstractors, and rule removers (the latter so called because their high levels of abstraction led them to remove business rules from the model for representation elsewhere). [15] Alternatively, the last of these could be called an ontological style of modeling.

The data-centric paradigm will require that information system designers learn to think in that ontological way. They will need to become adept at looking *at and through* the real world, using it as a lens through which to interpret requirements. The semantic core will be judged according to dimensions of quality that transcend particular functional requirements. Under the application-centric paradigm, flexibility (the ease with which a model can cope with change) and integration (consistency with the rest of an organization's data) have been areas of notable deficiency. [13] In the data-centric paradigm, those become essential.

4. A good semantic core engages critically with the affordances of business language.

The application-centric paradigm assesses an information system primarily on the basis of whether it delivers particular functional requirements. Within that framework, it's possible to gather requirements without questioning the meaning of business terminology very much, and for modelers to try to take the terminology as a literal basis for data structures. That approach treats the business language as if it were self-evidently meaningful. But if the language is semantically unsound, then it will lead to data structures that do not have a close correspondence to the real world and are therefore not viable over the long term. That is a common contributor to failed or challenged information systems projects.

Data models having that flaw are, as John Artz points out, victims of a kind of mental confusion identified centuries ago by Sir Francis Bacon. [2] Their use of language involves "idols that have crept into the intellect out of the contract concerning words and names", in which "names, though they refer to things that do exist, are confused and ill-defined, having been rashly and incompetently derived from realities." [3]

Theorists of both data modeling and requirements engineering have emphasized that language is at the center of information systems. Michael Jackson writes that "the central activity of software development is description". [8] Description, of course, consists of language. Similarly, the ISO/TC97/SC5/WG3 pointed out that a database is composed of sentences. Some refer to concrete phenomena such as *the car having license plate 1234 is a Mazda* while others are abstractions such as *every car has a license plate number and a brand*. The former can be formalized as a sentence in propositional logic, the latter in predicate logic. [5] But they are all sentences. They are made up of language— nothing more and nothing less.

Despite these acknowledgements, experience suggests that in the culture of information system development, concern for language is the exception rather than the rule. Academic programs in computer science and information systems do not seem to teach students much about eliciting the semantics of business language. It is very common to meet practitioners who are comfortable talking about user requirements for processing input and output, but not about what the terminology means.

Today's data-centric movement aims toward a future in which "data is an open resource that outlives any given application" and "data is globally integrated sharing a common meaning". [11] Reaching that aspiration requires a change in how the culture views language. New applications will be built upon an existing semantic core, the quality of which will be judged on its ability to easily extend to support any possible functional requirement. Designers of the semantic core will have to work with subject matter experts and other stakeholders of the information system to engage with business language in a critical spirit. Any incoherence in the business language will need to be identified and prevented from undermining the semantic core. The quality of the core, the truthfulness of its model of the real world, depends on the quality of the language chosen.

5. Necessary conversations

A paradigm shift is a more arduous project than promoting a new method or tool. It involves the comprehensive replacement of interlocking and mutually reinforcing assumptions and beliefs. In this case, the most powerful one has to do with the primacy of functional specifications; it is supported by skepticism about the possibility of designing a model of the real world, and by the cultural habit of taking language for granted.

The application-centric way of thinking has powerful inertia, but it is not insurmountable. Many people already do have a data-centric mindset. Others are able to recognize and step back from the patterns of thought in which they were educated so as to make space for an alternative.

The principles proposed above are intended to spark conversations that can lead to a body of work which will articulate a comprehensive justification for the paradigm shift. Those conversations need to include reflection on the history of system development methods. They must also point toward new ways of organizing projects with stakeholders.

These principles may seem to require attention to epistemological, linguistic, and even psychological matters beyond the interest of most information system practitioners, let alone of stakeholders who just want their functionality. But new foundations in those areas are necessary in order to establish a data-centric paradigm that can rescue the information systems field from its current afflictions. Acceptance will only come when data-centric ways of working intuitively make sense to a large proportion of people.

REFERENCES

[1] Aiken, P., & Harbour, T. (n.d.). The Data Doctrine. Retrieved from <https://www.thedatadoctrine.com/>

[2] Artz, J. M. (2007). Philosophical foundations of information modeling. *International Journal of Intelligent Information Technologies (IJIT)*, 3(3), 59-74.

[3] Bacon, F. (2017). *The new organon: or true directions concerning the interpretation of nature*. Early Modern Texts.

[4] Coursen, D., & Ferns, B. (2004). Modeling participant flows in human service programs. *Journal of Technology in Human Services*, 22(4), 55-71.

[5] van Griethuysen, J. J. (Ed.) (1984). *Concepts and terminology for the conceptual schema and the information base*. International Standards Organization.

[6] Hay, D. C. (1996). *Data model patterns: conventions of thought*. Dorset House Publishing.

[7] Hirschheim, R., Klein, H. K., & Lyytinen, K. (1995). *Information systems development and data modeling: conceptual and philosophical foundations*. Cambridge University Press.

[8] Jackson, M. (1995). *Software requirements and specifications: a lexicon of practice, principles and prejudices*. Addison-Wesley.

[9] Kent, W. (1978). *Data and reality: basic assumptions in data processing reconsidered*. New York: Elsevier Science Inc.

[10] Laurence, S., & Margolis, E. (2001). The poverty of the stimulus argument. *The British Journal for the Philosophy of Science*, 52(2), 217-276.

[11] McComb, D. (n.d.). The data-centric manifesto. Retrieved from <http://datacentricmanifesto.org/>

[12] McComb, D. (2018). *Software wasteland: how the application-centric mindset is hobbling our enterprises*. Technics Publications.

[13] Moody, D. L., & Shanks, G. G. (2003). Improving the quality of data models: empirical validation of a quality management framework. *Information systems*, 28(6), 619-650.

[14] Simsion, G. (2007). *Data modeling theory and practice*. Technics Publications.

[15] Simsion, G., Milton, S. K., & Shanks, G. (2012). Data modeling: Description or design?. *Information & Management*, 49(3-4), 151-163.

[16] Sokal, A., & Bricmont, J. (1998). *Fashionable nonsense: postmodern intellectuals' abuse of science*. Picador USA.

[17] Zave, P., & Jackson, M. (1997). Four dark corners of requirements engineering. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 6(1), 1-30.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Tagged coursen data centric data governance Paradigm semantic core

Share

Like 0

Tweet

Share

+ reddit this!



About Derek Coursen

Derek Coursen develops information systems strategy and data architecture for public service organizations. He has led informatics departments at two major nonprofit agencies in NYC and has been adjunct faculty at NYU's Wagner School of Public Service. Derek holds master's degrees in information science (Pratt Institute), information systems (Pace University) and management (NYU). He can be contacted via [Derek Coursen Consulting LLC](#) or [LinkedIn](#).

View all posts by Derek Coursen →

Related Posts