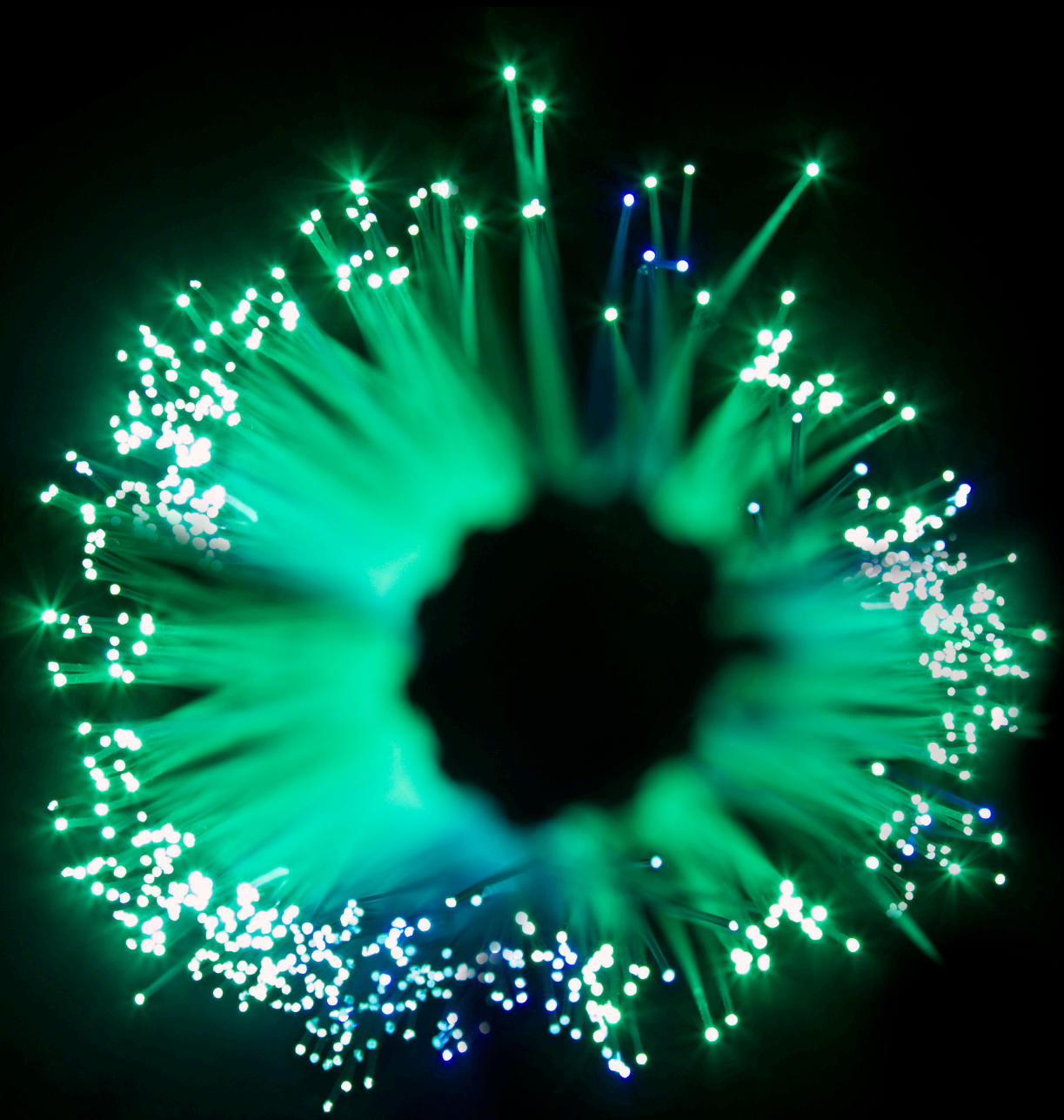
Deloitte.
Digital



Canonical Data Models & Microservices

Clarifying the conflicting views

Article by Tanya du Preez

We have been drawn into countless discussions regarding the usefulness of a Canonical Data Model (CDM). This design pattern is fully supported by the application of the Standardised Service Contract design principle that advocates which service contracts be based on standardised data models.

But the use of a CDM in the microservices space has become the centre of heated debates among CDM advocates and non-advocates alike.

CDM is essentially a Service Orientated Architecture (SOA) concept, promoting the reuse of data structures, attributes and data types during messaging between various systems and components. The use of a standardised data model decouples applications by exposing reusable services.

Although it does simplify the understanding of the business data, it could easily become cumbersome and infinitely difficult to manage as requirements, or the number of web services, expands. It is therefore difficult to design and implement from scratch.

The notion of a Canonical Data Model is rejected by the microservice integration pattern because microservices do not require Canonical Data Models. The data model owned by each microservice is part of a distributed Canonical Data Model. Stefan Tilkov, co-founder and principal consultant at InnoQ, a technology consulting company, strongly opposes the creation of CDMs. He firmly believes that too much time and effort is wasted in countless meetings to define the data model and the result is a model with many optional attributes.

Yet we believe optional attributes are a strength and that a CDM could add value to most integrations, whether based on SOA or other architecture patterns.

Integration technical lead at Deloitte Digital, Miguel de Barros, confirms that a microservice architecture is arguably the next step in the evolution of a SOA paradigm.

Myths regarding Canonical Data Models

Canonical Data Models may not be changed

As the business landscape evolves, data requirements follow suit and canonical schemas have no option but to change as well.

Furthermore, not all data requirements are bedded down at the time of development, which forces the need to extend the existing canonical schema to cater for new requirements.

Having said that, once modeled, CDMs are extended as opposed to making full-scale changes. This could potentially result in the canonical schema having to be revised multiple times before the final version, but does not negate the need to have such a model.

"The Only Thing That is Constant is Change"

Heraclitus
535BC - 475BC

Canonical Data Models are forced on consumers and providers

Though these models should not be forced on other teams, they do provide benefits to the organisation as a whole. By involving various teams during the design of the canonical schema, their contributions can enrich the model and provide a common understanding of the data and its use.

Involving consumers and providers would likely increase the chance of acceptance of the model, as opposed to widespread panic and resistance.

The complete Canonical Data Model must be realised in all systems

A Canonical Data Model and canonical schema is designed to define business entities in a standard manner, including the structure, attributes and data format. As a result, the models tend to be very large and often complex.

A good model would be fairly generic with most fields being optional. This enables various systems to only use the data that they are required to use for a specific service without being forced to provide data that is irrelevant to them. This ultimately promotes reuse of the model.

Additional microservices schemas will enforce various validations per scenario and it is therefore not a requirement that all systems implement a 15 000 line Canonical Data Model. Each system should use the CDM as a basis to implement their own schemas, which contain only the subset of the CDM that they wish to use.

"The CDM does not comprise of all data within the organisation, but merely the data that is shared on the integration."

A CDM is comprised of all data within an organisation

The CDM does not comprise of all data within the organisation, but merely the data that is shared on the integration layer. Data that is internal to a domain would therefore not be included during the design of a Canonical Data Model.

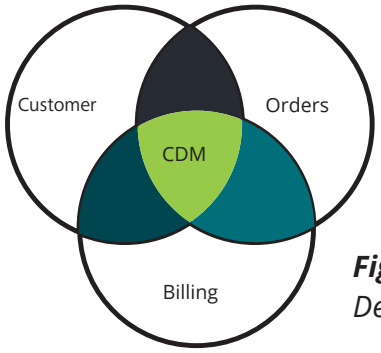


Figure 1: *Defining the CDM*

A business has a single Canonical Data Model

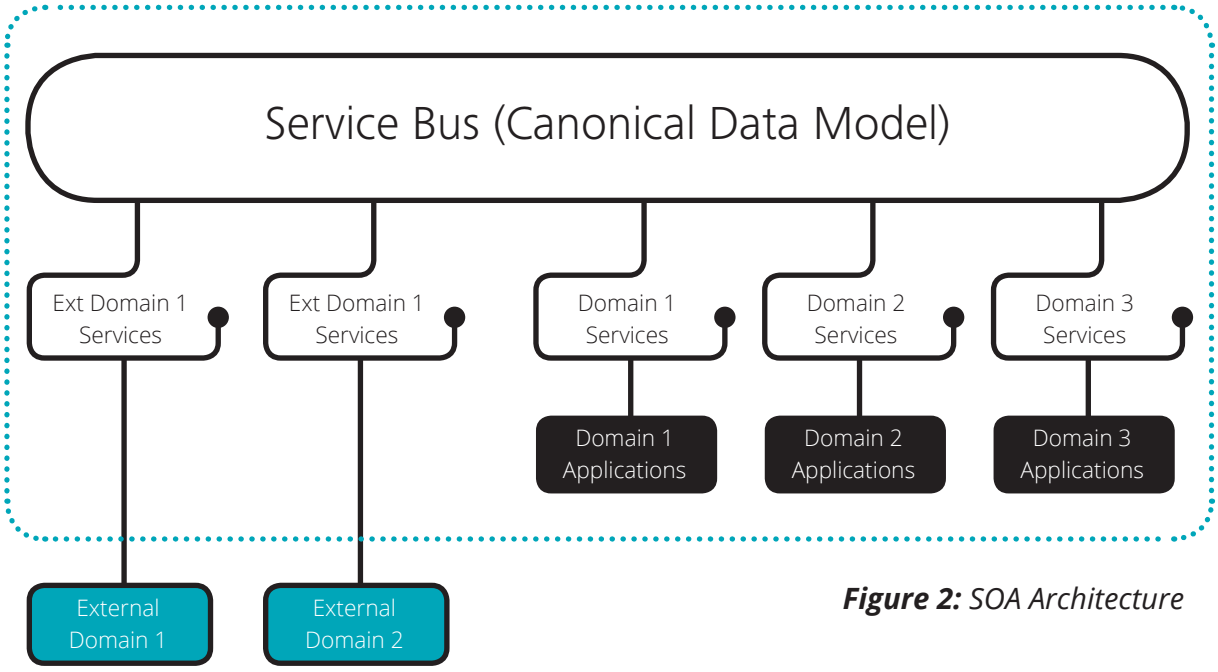


Figure 2: *SOA Architecture*

As stipulated above, a single Canonical Data Model is a SOA concept.

Although a single CDM provides many benefits, a single change to the model would ripple through all services across all domains.

Does this then repudiate the need of a data model? Surely not? The advantages still outweigh the disadvantages if implemented appropriately.

Although Tilkov opposes the creation of standardised data models, he does provide advice on the creation of the Canonical Data Model that closely resembles our view. Tilkov refers to bounded context, a domain-driven design concept that supports the dividing of a large model into smaller contexts, thus allowing for business objects to be modelled differently and according to the need in each context.

De Barros supports the use of multiple CDMs in the microservices architecture and also states that the typical design principles of Abstraction and Composition that are expected in SOA are not only applied to Software, but also to the infrastructure layer.

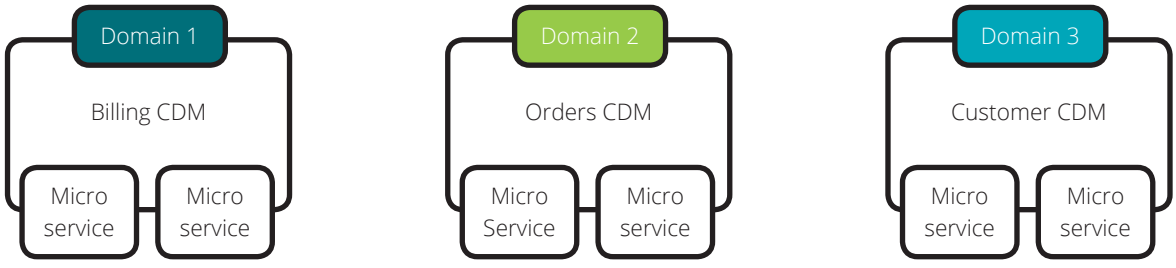


Figure 3: *Microservice architecture*

Similarly the concept of a CDM could be applied to a microservice architecture with a single caveat: The CDM should be organically derived and established as the microservice architecture grows in maturity and size. The idea being that the CDM applied here is as lightweight as possible to only describe what is needed at the current point in time.

Thus we are promoting the idea that a lightweight Canonical Data Model should be designed per functional domain.

If you are building an application for a shipping business, you could potentially have canonical models for customers, orders and billing.

Advantages of Canonical Data Models

Common understanding of business data

It often happens that various teams within an organisation have different terminology for the same entity. A CDM promotes a common definition to eliminate ambiguity, errors of interpretation, and miscommunication.

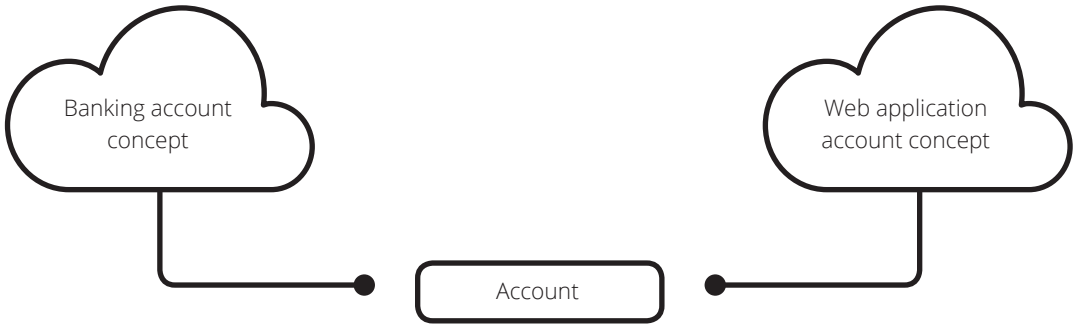


Figure 4: *Standardising definitions*

Reduction of processing overhead

Data transformation is one of many contributors to processing overhead. Once the CDM or subsets of it is implemented across the enterprise, the mappings between applications will no longer be required. This results in driving down the processing overhead and increasing application performance.

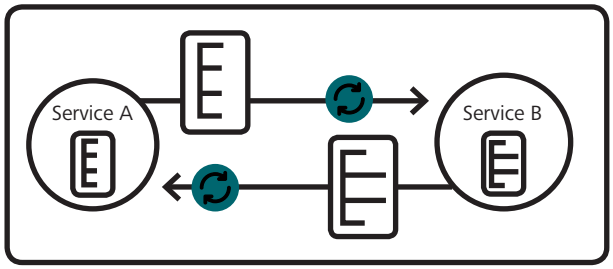


Figure 5: *Data Transformation*

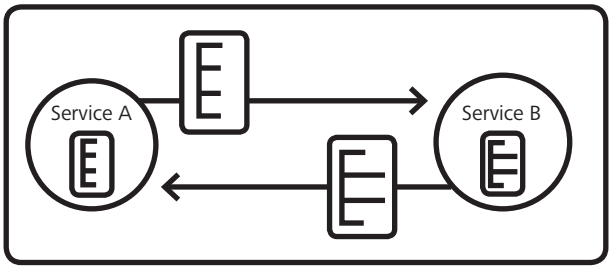


Figure 6: *No Data Transformation*

Reduction of integration costs

Applications transform once to the CDM, as opposed to multiple transformations between systems in point-to-point integration, ensuring the impact of schema changes are minimised. Creating your "canonical" view of what an entity looks like and realising it within your middleware can help shield you from changes in the future.

"The concept of a CDM can be applied to a microservice architecture with a single caveat: The CDM should be organically derived and established as the microservice architechture grows in maturity and size."

Miguel de Barros - Deloitte Digital South Africa

Disadvantages of Canonical Data Models

Development cost

Defining and maintaining common data-models is a significant governance exercise. Adding fields to the CDM would also result in multiple microservices having to be updated, although these changes affect a lesser number of services than the implementation of a single canonical model.

Versioning

Change is inevitable. The canonical schema definition needs to undergo changes and there is a need to implement a versioning strategy for the CDM.

Run-time cost

When a canonical schema is involved a request needs to be transformed first to a canonical format and then to the target provider's format before being processed. Considering the performance overhead inherent in processing – an additional transformation hop may make the canonical unviable for several high performance systems depending on the middleware used.

Conclusion

Although the defining of Canonical Data Models is inherently a SOA concept, and albeit rejected by the microservices architecture pattern, a good case can be made to design and use a CDM during microservice implementation.

The benefits of a standardised data model, including data consistency, a common understanding of business entities and the reduction of processing overhead by removing unnecessary mappings between systems unquestionably supports the use of a CDM.

We suggest implementing a CDM for microservices, by defining a lightweight Canonical Data Model per functional domain. In the case of the domain being too small to implement a CDM, objects from the various CDMs can be reused in the microservices schemas.

This solution results in a common understanding of business entities and the design of distributed Canonical Data Models, without the constraint of "one service to rule them all".

Tanya du Preez
Integration Analyst and Solution Designer at Deloitte Digital
tadupreez@deloitte.co.za

Designed and produced by Creative Services at Deloitte, Johannesburg. (00000/kea)