**OData Design Principles**July 26, 2012

Overview

OData is an application-level protocol for interacting with data via RESTful web services. OData builds on existing standards such as HTTP/REST, JSON, AtomPub, and XML by defining common conventions, semantics, and encodings across different services that expose data.

As the OData Protocol is extended and standardized, it is critical to the API consistency and simplicity to continue to apply the principles that have shaped OData and guided decision making through the open design process.

Design Principles

**OData Services are Model Driven.** Rather than exposing a discrete set of APIs, OData services expose a data model which defines the shape of data exposed by the service. The operations available from the service are derived from the model according to OData conventions for querying, updating, and invoking operations against the exposed model. This improves semantic interoperability between systems while giving clients more flexibility in constructing requests for the desired data.

**OData services are hypermedia-driven and follow REST principles wherever possible**. Following REST principles, and deriving semantics from an exposed data model, results in a protocol that enables loose coupling between clients and services, where agents can be built and versioned independently while retaining a good level of functionality.

**OData Must Remain Simple.** It should be easy to implement a basic OData service over a variety of data sources, with optional incremental work to support additional/advanced capabilities. The conventions and semantics of OData over a well-defined model, along with the ecosystem of available tools and libraries, should make it easier to implement an OData service than a custom RESTful protocol.

**OData favors Reach over Rich.** OData's popularity is based on its simplicity and broad reach. It is not intended to impose semantics across every possibly feature of an underlying store, but rather defines common functionality to enable breadth clients to easily consume appropriate data from a wide variety of sources.

**OData is Consistent.** OData's simplicity is enhanced by providing consistency across the protocol and defining a single way to express each piece of functionality, rather than defining an inconsistent set of possibly redundant concepts.

**OData Services are Client-Focused.** OData prefers to push complexity to the service, allowing the service modeler to define the interesting views of data that a client may consume through simple, common operations, rather than force the client to build complex queries against raw data.

**OData is Extensible.** OData defines extensibility mechanisms for annotating data or metadata, defining custom functions, operations, and query options. These mechanisms allow common, shared extensions for features that are not part of OData's core functionality.

**OData Extensions Are Well Behaved.** Common extensions should not violate the core semantics of OData. Client libraries should be able to ignore extended functionality and still correctly query and consume data according to core semantics.

**OData Extensions Compose Well.** Adhering to the above principles and following core OData semantics help independent extensions to compose well. Additionally, extensions should carefully consider functionality specific to the intended extension versus common across extensions to avoid defining different representations for common concepts across extensions.