Differences between ODATA and GraphQL

ODATA

OData stands for Open Data Protocol, is an OASIS standard for REST API, which defines the best practices for building and consuming them. Microsoft developed it in 2007, which allows the creating and querying of interoperable Restful APIs in a simple way. It allows the programmer to focus on the business logic without worrying about the approaches to define other things like request/ response headers, status code, status code, media types, etc. It provides the rich query language with the features of sorting, filtering, querying, etc. It is strictly RESTful and expects and is URI- centric supporting the REST operations like GET, POST, PUT, DELETE, etc.

GraphQL

GraphQL is a query language for APIs used for fulfilling the queries with the existing data. Facebook developed it in 2015 before public release in 2015. It provides the complete description of data in your API with powerful data tools. It allows the single URI mapped to a web service with the API, which is capable of handling the most complex data requirements. It is not RESTful but provides many libraries for implementing and querying GraphQL. A single endpoint works great in GraphQL which makes it very easy to work for the consuming developer.  It supports all the things related to the query, be it joins, selects, versioning, relationships, etc., in a single request/ response round trip.

Key Differences

* OData is RESTFul, which means that it uses the REST pattern like POST, PUT, GET, DELETE to perform the desired operations, whereas GraphQL is not RESTful, which uses the Http with the POST request only in which the query is passed as body content.
* OData uses the URL to build the query, which sometimes hampers the size of the request, whereas, in the case of GraphQL, the query has its own definition as it is passed in the body is no issue regarding its size.
* OData is comparatively easy to understand and work for the programmers as the programmers are quite familiar with the REST-based frameworks. In terms of learning, GraphQL is a bit complex to understand for querying and implementing.
* OData supports the Delta Feeds, which is a standard way for providing the client specific link without being stateful, whereas GraphQL supports the Subscriptions, which is almost similar to the Delta Feeds in the GraphQL.
* In terms of coupling, OData provides the power of SQL to the URI at the cost of high coupling, whereas GraphQL behaves like a backend which brings all the Rest APIs all together in one interface.
* GraphQL provides a single endpoint to handle all the API requests, and there is no need to maintain the versioning endpoint. It also has an introspection endpoint which makes it easy to see the published schema and makes it easy to find tools. So, in this case, GraphQL is considered to be the best option.
* OData supports joins and relations in the form of relationships, but it does not support the versioning, unlike GraphQL, which supports the versioning, joins, selects, etc., from request/ response round trip.
* One positive point with the OData is that it provides a simpler definition for queries with easy and out of the box integration to migrate with the already existing webAPIs whereas in the case of GraphQL user has to do a lot of work like creating the schema, query, creating types for each query.
* Developer Usability Considering you have existing WebAPI(DotNet Framework) and want to migrate to GraphQL or OData compatible WebAPI then better option is to choose OData because of its easy integration and Out of the Box Filter, OrderBy, Select, Expand, etc feature (Refer MSFT On DotNet OData). If you choose GraphQL then you have to do a lot of work like Create Type, Schema and Query and Implement Resolver for each query.
* Performance depends on your query logic. GraphQL and Odata Both have the capability to get what you request using $select in OData and in GraphQL you can request by their Query convention.  
  s
* API Development From Scratch, If you want only single endpoint for all API request and don't want to maintain versioning endpoint, Autosuggest field name and type of schema then GraphQL is the best option. But Availability of GraphQL Library for Each framework and Community varies for Technology stack(e.g. nodejs, C#, Ruby, Java etc)