Chapter 3
Dataspaces: Fundamentals, Principles, and Techniques

Keywords Dataspaces · Best-effort information · Approximation · Incremental data management

3.1 Introduction

A dataspace is an emerging approach to data management which recognises that in large-scale integration scenarios, involving thousands of data sources, it is difficult and expensive to obtain an upfront unifying schema across all sources. Data is integrated on an "as-needed" basis with the labour-intensive aspects of data integration postponed until they are required. Dataspaces reduce the initial effort required to set up data integration by relying on automatic matching and mapping generation techniques. This results in a loosely integrated set of data sources. When tighter semantic integration is required, it can be achieved in an incremental "pay-as-you-go" fashion by detailed mappings between the required data sources. This chapter introduces dataspaces and the fundamentals of "best-effort" data management.

The chapter is structured as follows: Sect. 3.2 discusses big data and the challenge of data integration with the long tail of data variety, Sect. 3.3 examines the cost of data management, and Sect. 3.4 addresses the emerging trend of approximate, best-effort and "Good Enough" approaches to data management. Section 3.5 provides a detailed explanation of the fundamentals of dataspaces, including their principles and a comparison to contemporary data management approaches. Section 3.6 covers dataspace support platforms, support services, life cycle, and specific implementations. Section 3.7 details the technical challenges for dataspaces, Sect. 3.8 sets out ongoing research challenges for dataspaces, and finally, a summary is provided in Sect. 3.9.

E. Curry, *Real-time Linked Dataspaces*,
https://doi.org/10.1007/978-3-030-29665-0_3

3.2 Big Data and the Long Tail of Data

The emergence of new platforms for decentralised data creation such as the Internet of Things and edge computing, the increasing availability of open data on the web [68], and the increasing number of data sources inside organisations [69] bring an unprecedented volume of data to be managed. In addition to coping with the volume of data, data consumers in the big data era need to cope with data variety where data is created under different contexts and requirements [25]. Consuming data comes with the intrinsic cost of repurposing, adapting, and ensuring data quality for its new context. Data at large scales coming from distributed sources can be erroneous, inconsistent, and incomplete for some users' requirements. Jagadish et al. state that "Big Data increasingly includes information provided by increasingly diverse sources, of varying reliability. Uncertainty, errors, and missing values are endemic, and must be managed" [70].

The growth in the number of data sources and the increasing scope of information systems leads to a *long tail of data variety* [71]. The long tail of data variety (see Fig. 3.1) reflects the distribution of the frequency of use of conceptual elements: in a large domain of interest few entities and attributes have a high frequency of use, followed by a long tail distribution of entities and attributes which have lower frequencies of use. While some concepts are central across many different areas, most of the concepts are specific to a context. In the scientific domain, for example, the long tail of scientific data [72] reflects the conceptual distribution of scientific data.

Traditional relational data management environments were focused on data that mapped to popular business processes and were regular enough to fit into a relational model. The long tail of data variety expresses the shift towards the expanding coverage of data that must be managed; it is less frequently used, more decentralised, and less

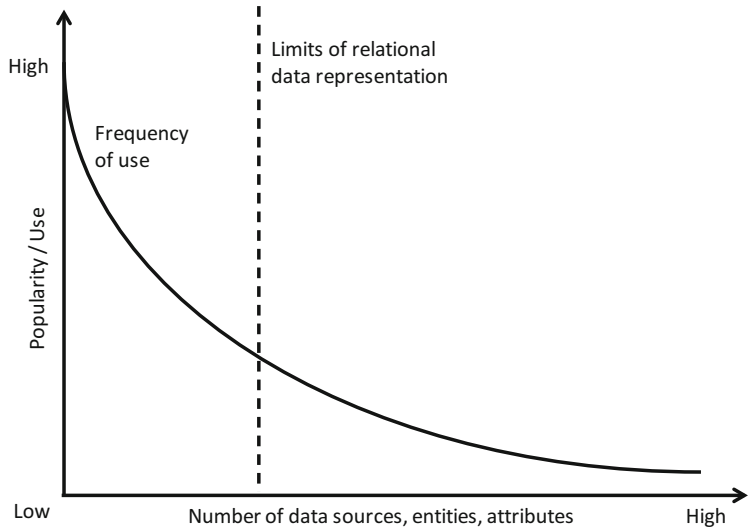

Fig. 3.1 The long tail of data variety. Adapted from [71]

structured. Given this shift in the data landscape, there has been an evolution in the information processing landscape to meet these new challenges. Big Data technologies are moving towards supporting the long tail of data variety to enable consumers to have a richer and more comprehensive model of their domain that they can *search*, *query*, *analyse,* and *navigate*. However, managing this long tail of data comes with a cost.

3.3 The Changing Cost of Data Management

Historically, the construction of information systems and databases has evolved following a model dependent on the cost of formalising a domain and the associated value derived from the efficiency gain. Data management practices for organisations have prioritised the formalisation (conceptualisation) of domains within a centralised model with high levels of control, which have resulted in high data management costs. Propelled by the growth of data and the increasing number of systems and devices producing data, data management requirements are shifting towards the need to cope with data which is generated in a decentralised manner [73, 74], data which is intrinsically heterogeneous, and data with varying levels of structure and different contexts. These trends are contributing to the long tail distribution of data variety.

According to Brodie and Liu [69]:

> The consistency of all views of the same tuple leads the underlying belief in a single version of the truth and the concept of a global schema. The dramatic success of relational technology has propelled data modelling and management requirements beyond the modelling and processing capabilities of the relational technology. The phrase 'single version of the truth' seems intuitively correct and may assure in a confusing world, but it is almost entirely false in the real world. The underlying assumption of the relational world is not just semantic homogeneity but also ontological homogeneity while in reality, semantic heterogeneity dominates. Data management vendors promote the 'single version of truth' assumption as a highly desirable objective and something that their products can provide. Our Digital Universe is no longer a semantically homogeneous set of a few databases but Information Ecosystems of 100s or 1000s of semantically heterogeneous databases to be managed and integrated collectively [69].

Franklin et al. [2] highlight that "in data management scenarios today it is rarely the case that all the data can be fit nicely into a conventional relational [database]". Diving deeper into the practical challenges of managing decentralised and heterogeneous data, Franklin et al. [2] examine the problem along two problem dimensions (see Fig. 3.2):

- *Administrative Proximity:* Describes how data sources within a space of interest are close or far in terms of control. A close control means that many assumptions can hold concerning guarantees such as data quality and consistency, while a far control refers to a loosely coupled environment and a lack of coordination on the data sources within the data management system.
- *Semantic Integration:* Refers to the degree of how much the data schemas within the data management system are matched up. That includes, for example, the types, attributes, and names used within the data sources. On one end of the

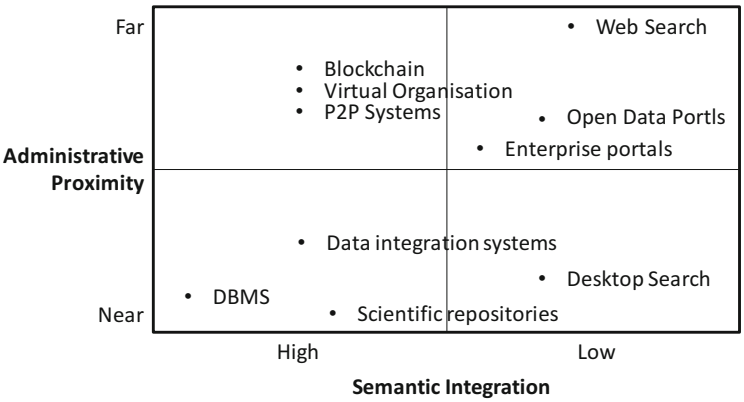

Fig. 3.2 Data management approaches. Adapted from [2]

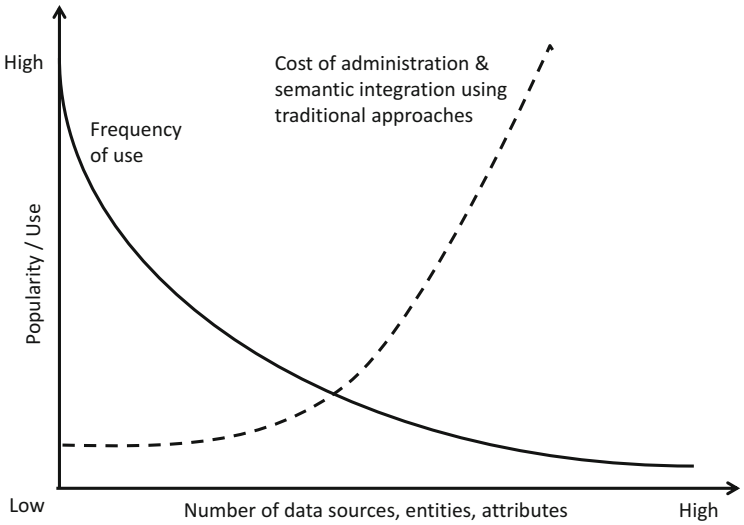

Fig. 3.3 Data management cost associated (administration and semantic) with increasing numbers of schema and sources. Adapted from [71]

spectrum, all data conform to an agreed-upon schema, while on the other end of the spectrum schema information is lacking. This dimension is relevant to how much semantically rich querying can be done.

In this view, data management should be considered as a task that takes place in a spectrum defined by these two dimensions. Within a Database Management System (DBMS), administrative control and semantic homogeneity are key to the data modelling, querying, and integration approaches that depend on the relational data model. This results in a high upfront cost for DBMS, which only gets more significant when dealing with the long tail of data, as illustrated in Fig. 3.3.

The need to develop more effective and efficient approaches for dealing with decentralised semantically heterogeneous environments is highlighted by the Lowell Self-Assessment report [75], which is a roadmap for the future of research in the database community: "A semantic heterogeneity solution capable of deployment at Web scale remains elusive. At Web scale, this is infeasible, and query execution must move to a probabilistic world of evidence accumulation and away from exact answers. Therefore, one must perform information integration on-the-fly over perhaps millions of information sources" [75].

3.4 Approximate, Best-Effort, and "Good Enough" Information

Not every situation or decision requires information that is 100% fresh and accurate [76], as many information consumers can efficiently work with information at a lower threshold. A perfect result is not always necessary, while a lower-quality or less-than-optimal result is sufficient [77]. By relaxing the need for computing to the highest quality, approximate approaches can be used to improve the throughput and response time of services.

Many applications that require data processing can tolerate a reduced quality in the result of the data analysis. Consider the rise of "Recognition" applications, a new class of popular mobile edge applications that range from recognising a single user's surroundings and augmenting it with information, advice, and decision support, to analysing an array of images from traffic cameras within a smart city to manage traffic. Recognition applications may have the possibility to trade off the accuracy of analysis with the responsiveness and computation necessary for the analysis.

The Pareto Principle (or the 80/20 rule) has wide application in many areas from economics and market analysis to business strategy, where it has been observed that 20% of the effort delivers 80% of the results. Within computer science, the principle has been observed within many problems from fixing bugs to writing code. The principle can help us to prioritise actions, for example, focus on the 20% of software bugs that cause 80% of the system crashes. Data management approaches can take advantage of the fact that many users and applications can tolerate approximate results; a trade-off between exact and approximate results can minimise data integration costs and the response time (both network and compute), and maximise throughput. Relaxing the need for maximum quality reduces the required data integration and computation workload, enables a significant reduction of response time, and increases throughput.

Approximate approaches can reduce semantic integration costs due to their ability to deal with the uncertainties of semantics. Approximate approaches can operate in environments with low-cost agreements on administration proximity and semantic integration, and at the same time, achieve acceptable levels of precision and recall

comparable to exact models. These characteristics make these approaches suitable for tackling the long tail of data variety.

Approximate "good enough" approaches are distinguished by a matching model that is not Boolean and supports one form or another of uncertainty, probability, or ranking of the results. This gives the matching model more flexibility to deal with data heterogeneity and thus, improves its ability to address lower levels of administrative control and semantic integration.

The classic relational paradigm defined the use of structured query languages (such as SQL) as the primary mechanism for user–data interaction. SQL offers crisp and accurate answers for relatively small numbers of homogeneous sources [76], typically managed in a centralised and coordinated manner. Within the approximate model, the underlying assumptions of user-data interaction and the use of structured queries to cope with the long tail of data were revisited by [39, 76]:

From: *Clean, semantically homogenous and centralised schema*

To: *Semantically heterogenous and decentralised schema*

As schemas are managed in a decentralised way, different conceptualisations may exist in the same schema. "We can no longer pretend to live in a clean world . . . ." [76]. "Unless the reader of a message or document is specifically programmed for it, there will likely be confusion. The meaning of the message, the interpretation of its fields, and much more, will be subject to approximation and a loss of clarity. Different companies, different countries, and even different regions within a country have different understandings of data" [76].

From: *Manual query-schema mapping*

To: *Automatic query-schema mapping*

Most of the interaction with structured data is dependent on manual mapping among elements of a structured query and schema elements. With the growth in the schema-size and the number of available data sources, the cost associated with this manual mapping process becomes prohibitive (see Fig. 3.3) requiring automated and semi-automated query-mapping techniques.

From: *Absolute precision/full recall in a single query*

To: *Relaxed precision/recall in multiple queries*

As schemas grow and as users cross database boundaries, the cost associated with building structured queries grows exponentially. In this scenario, the expectation of getting a correct and complete answer in a single interaction should be exchanged by approximate answers which are obtained through multiple interactions. As Helland states [76] "Too much, too fast—you need to approximate."

By creating data management approaches that utilise approximate and best-effort techniques, it is possible to reduce the cost of dealing with the long tail of data variety. Approximate approaches trade off administrative and integration costs with reduced accuracy of results. "Best-Effort" thinking is at the core of the new dataspace paradigm of data management (Fig. 3.4).

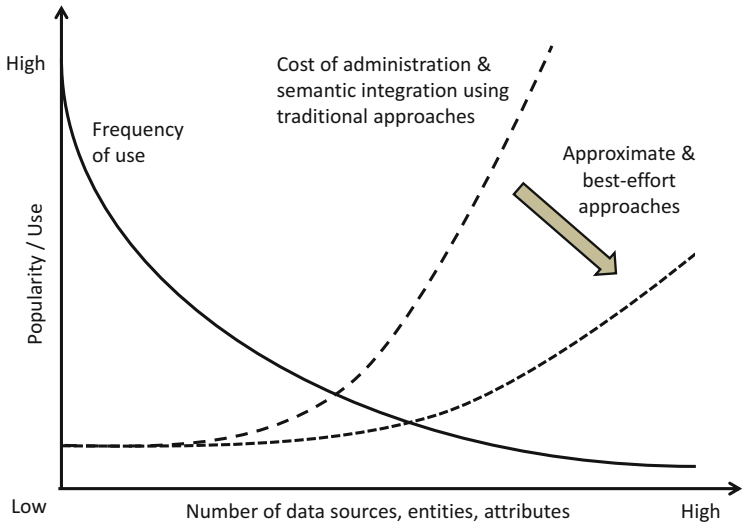

Fig. 3.4 The long tail of data and the improved scalability of data management with approximate and best-effort approaches

3.5 Fundamentals of Dataspaces

A dataspace is an emerging approach to data management that is distinct from current approaches. The dataspace approach recognises that in large-scale integration scenarios, involving thousands of data sources, it is difficult and expensive to obtain an upfront unifying schema across all sources [2]. Dataspaces are not a data integration approach [2]; they shift the emphasis to providing support for the co-existence of heterogeneous data that does not require a significant upfront investment into a unifying schema. Data is integrated on an "as-needed" basis with the labour-intensive aspects of data integration postponed until they are required. Dataspaces reduce the initial effort required to set up data integration by relying on automatic matching and mapping generation techniques. This results in a loosely integrated set of data sources. When tighter semantic integration is required, it can be achieved in an incremental "pay-as-you-go" fashion by detailed mappings among the required data sources. This section details the fundamentals of the dataspace paradigm, including their core principles, comparison to existing approaches, support platform, data services, life cycle, and research challenges.

3.5.1 Definition and Principles

First introduced by Franklin, Halvey, and Maier in 2005 [2], a dataspace can contain all the data sources for an organisation regardless of its format, location, or model.

Table 3.1 Definitions of a "Dataspace" from literature

Definition	Source
"Dataspaces are not a data integration approach; rather, they are more of a data co-existence approach. The goal of dataspace support is to provide base functionality over all data sources, regardless of how integrated they are."	[78]
"A dataspace system manages the large-scale heterogeneous collection of data distributed over various data sources in different formats. It addresses the structured, semi-structured, and unstructured data in coordinated manner without presuming the semantic integration among them."	[79]
"to provide various of the benefits of classical data integration, but with reduced upfront costs, combined with opportunities for incremental refinement, enabling a "pay-as-you-go" approach."	[80]
"enable agile data integration with much lower upfront and maintenance costs."	[81]
"A dataspace system processes data, with various formats, accessible through many systems with different interfaces, such as relational, sequential, XML, RDF, etc. Unlike data integration over DBMS, a dataspace system does not have full control on its data, and gradually integrates data as necessary."	[82]
"Dataspace Support Platforms envision data integration systems where the amount of upfront effort is much smaller. The system should be able to bootstrap itself and provide some useful services with no human intervention. Over time, through user feedback or as sources are added and the data management needs become clearer, the system evolves in a pay-as-you-go fashion."	[83]
"Dataspace is defined as a set of participants and a set of relationships among them."	[84]

Each data source (e.g. database, CSV, web service) in the dataspace is known as a participant. The dataspace can model the relations (or associations) between data in different participants. In its purest form, a dataspace is a set of participants and the inter-relations between them [2]. The modelling of the dataspace can capture different types of relations among participants, from the mapping of the schemas between two participants to capturing that Participant A is a replica of Participant B.

The dataspace concept has gained traction with a number of different groups exploring its usefulness for managing data from different domains and investigating the design of support services. These works have provided a number of definitions for a dataspace as captured in Table 3.1; most of these build on the initial concept from [2].

Dataspaces, as a paradigm of data management, push the boundaries of traditional databases along the dimensions of administrative proximity and semantic integration, as discussed in Sect. 3.3. Within a dataspace, data sources are not tightly controlled, and full semantic integration is not guaranteed. Data management within a dataspace is defined by different principles, as described by Halevy et al. [78]:

- A dataspace must deal with data and applications in a wide variety of formats accessible through many systems with different interfaces. A dataspace is required to support all the data rather than leaving some out because they do not yet conform to a specific schema or data constraint.

Table 3.2 DBMS vs. Dataspace. Adapted from [85]

	DBMS	Dataspace
Model	Relational	All
Formats	Homogeneous	Heterogeneous
Schema	Schema first, data later	Data first, schema later or never
Control	Complete	Partial
Leadership	Top-down	Top-down/Bottom up
Query	Exact	Approximate
Integration	Upfront	Incremental
Architecture	Centralised	Decentralised
Real-time data processing	No	Applicable

- A dataspace must provide an integrated means of search, query, update, and administration. The dataspace does not subsume participant data sources and is not in full control of the data. The same data may also be accessible and modifiable through an interface native to the system hosting the data.
- Queries to a dataspace may offer varying levels of service, and in some cases may return best-effort or approximate answers. For example, when individual data sources are unavailable, the best answers available are returned using the data accessible at the time of the query.
- The dataspace must provide pathways to improve the integration among the data sources in a "pay-as-you-go" fashion.

3.5.2 Comparison to Existing Approaches

Data sources in a dataspace co-exist, and co-evolve over time, and are not subsumed by a rigid data management system. This is in stark contrast to the traditional data management approach based on relational databases. A comparison of the dataspace paradigm to traditional DBMS is provided in Table 3.2.

3.6 Dataspace Support Platform

The goal of a Dataspace Support Platform (DSP), as detailed in Fig. 3.5 [2], is to provide a set of common related support services to all data sources within the dataspace (e.g. keyword search). The DSP provides a base functionality needed for data integration that enables developers to focus on application-specific challenges instead of the common data integration tasks faced when working with multiple data sources. To achieve this goal, the DSP must support all the data in the dataspace requiring it to work, with a large variety of data formats and system interfaces. A dataspace does not host data; the data resides in their native systems. As such, a

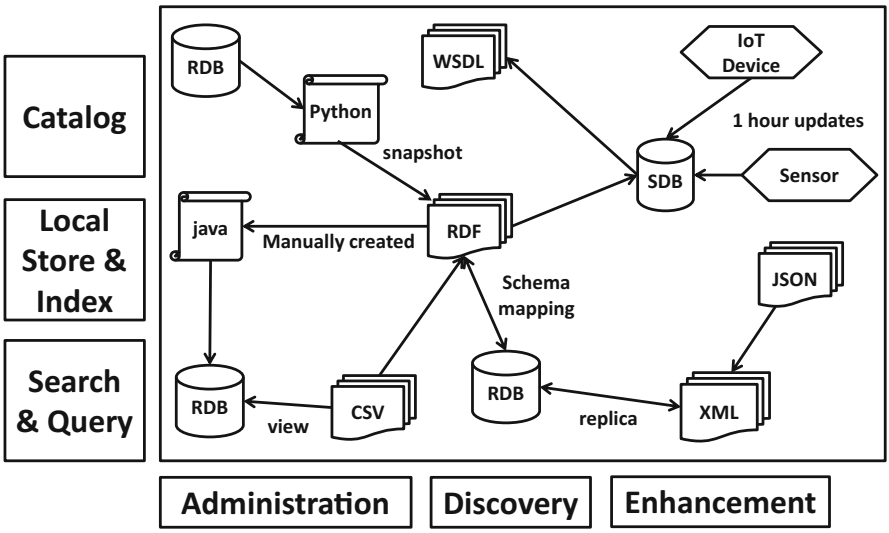

Fig. 3.5 A dataspace support platform. Adapted from [2]

dataspace is not in full control of the data and may only provide weak guarantees of consistency and durability. When stronger guarantees are desired, more effort can be put into making agreements among the various systems. To this end, a DSP must provide tools to support the tighter integration of data in a pay-as-you-go manner. As a result of the varying levels of data integration, the DSP offers varying levels of service and often will only be able to provide best-effort or approximate results using the data accessible at the time of the query [2].

3.6.1 Support Services

Services within a DSP need to support heterogeneous data types and multiple access methods to participants within the dataspace. A core set of support services, as identified by [2] are:

- *Catalog:* A catalog is an inventory of data elements from participants containing basic information about each one, including source, name, location, size, creation date, and owner. A catalog service can provide a basic browse interface across the dataspace for users. The catalog is a core infrastructure that is used by other dataspace support services.
- *Search:* Search is the primary mechanism used by end-users to deal with large collections of unknown data. Search is based on a similarity analysis of data that results in a ranked list of results relevant to an end-user's keywords. The search service should examine all the contents of a dataspace, including metadata. Search interfaces within a DSP should support the interactive refinement of the

results (e.g. facets) so that users can explore a dataset and incrementally improve the search results. The process of refinement could result in a database-style structured query.

- *Query:* The level of support for expressive queries will vary across participants in the dataspace, as some will provide expressive query languages, while others will only have limited interfaces for querying (e.g., web services). The query service of a DSP needs to provide for (1) metadata queries to support the discovery of data sources, (2) monitoring of data sources, (3) local storage and indexing to support associations among participants, increase query expressivity on sources with limited querying functionality, and improve data source availability.
- *Discovery:* It locates participants in the dataspace and supports the creation of relationships among them in an incremental manner.

Not every participant in a dataspace will support all DSP functions. Thus, there will be the need to extend data sources in various ways (e.g. search and query). This requires the services to support dataspace participants in an incremental manner that can be applied in real time to existing as well as new participants joining the dataspace. The incremental nature of the support services is a core enabler of the pay-as-you-go paradigm in dataspaces.

3.6.2 Life Cycle

Similar to any other data management approach, a dataspace has a life cycle of operation. Hedeler et al. [80] have proposed a conceptual life cycle for dataspaces, illustrated in Fig. 3.6, consisting of seven phases with transitions between phases. As dataspaces can be used within different contexts, only a subset of the phases and transitions in the life cycle may be relevant to a specific implementation or deployment. The key phases in the life cycle are:

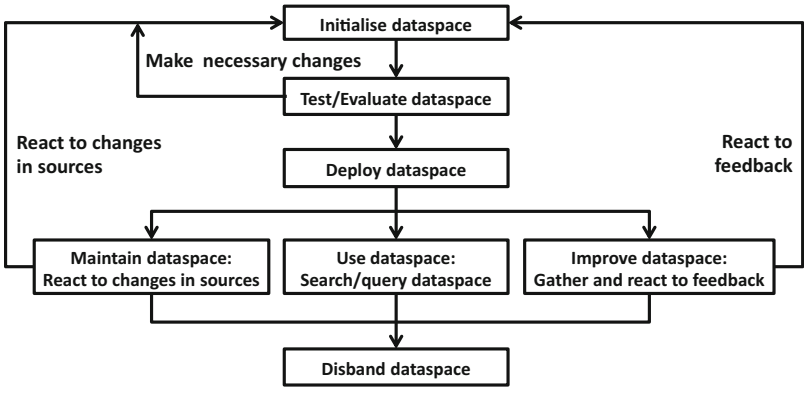

Fig. 3.6 Conceptual life cycle of a dataspace [80]

- *Initialise:* Identification of the data sources to be accessible within the dataspace and the integration of those resources. Sub-phases can include identifying data sources, designing integration schema, identifying matchings, deriving mappings, and deriving integration schemas.
- *Test/Evaluate:* Testing and evaluation of the dataspace before deployment.
- *Deployment:* Enabling access to the dataspace for users and applications, including the deployment of the dataspace on its physical computing infrastructure.
- *Use:* Support users and applications accessing and using the dataspace, including the search and query of sources in the dataspace.
- *Maintain:* Supporting changes to the participants in a dataspace, including adding and removing a source.
- *Improve:* Improving the dataspace during its operation, including the integration of participants. Explicit and implicit user feedback can be a key source of improvement for the dataspace.
- *Disband:* Gracefully close the dataspace by removing the participants and free resources.

As noted by [80], the phases Use, Maintain, and Improve are co-existing to support the "pay-as-you-go" model of data management.

3.6.3 Implementations

The dataspace approach has been implemented using several different technology stacks and used within a number of different contexts, including:

- *Personal Dataspace:* Focusing on the management of the information on a person across various sources, from their desktop [86] and mobile devices to their presence on social networks. Works include: iDM [87], SEMEX [88, 89], iMeMeX [90, 91], CoreSpace [92], and PDSP [93].
- *Scientific Dataspace:* Working with distributed sources of scientific data including astronomy data [94], biomedical data [81, 95], life sciences data with ALADIN [96] and LinkedScales [97], and process materials with the Virtual Data Space [98, 99].
- *Enterprise/Industrial Dataspace:* Targets the use of a dataspace to bring together data from different organisations within the context of energy management [100], air travel (Airbus Skywise), Industry 4.0 [101], or a digital library [102].
- *Global/Web Dataspaces (Web of Data):* Efforts at global or web-scale dataspaces include PayGo [103] and OCTOPUS [104]. The use of linked data technologies to publish data on the web is enabling the linkage of records in distinct databases that can be viewed as a global dataspace [43]. They can have a multi-domain nature such as the Linked Open Data Cloud [105], or a domain-specific purpose, such as financial data [56].
- *Software Development:* The use of a dataspace to support software artefact management [106].

- *Internet of Things/Smart Environment Dataspaces:* Recent works have investigated the use of dataspaces to support the Internet of Things or smart environments [4, 107] and the data they produce.

3.7 Dataspace Technical Challenges

The key technical challenges to realising a dataspace as identified by Halevy et al. [78] centre around the need to answer queries, to introspect on the content of the answers, and to leverage human interaction to enhance the semantic relationships within a dataspace. In this section, we briefly summarise these challenges as detailed by Halevy et al. [78].

3.7.1 Query Answering

In order to understand the fundamental challenges of querying a dataspace, we briefly summarise modes in which we expect users to interact with a dataspace.

Participants and Relationships Dataspaces are modelled as a rich collection of participants and relationships that contain all of the information relevant to a particular organisation or entity regardless of its format, location, or data repository.

Queries Since multiple data models need to be supported within a dataspace, queries will also come in a variety of languages; from keyword searchers to structured forms and formal query languages. The dataspace support platform needs to provide mechanisms for executing queries in different languages and return results from all the relevant sources in the dataspace.

Answers Answers to queries within a dataspace follow a best-effort model which is different from queries over traditional databases. Answers can come in the following forms [78]:

- *Ranked:* A ranked set of answers to structured queries and/or keyword search. Rankings may be based on different methods (e.g. relatedness, similarity) or approximate matchings from different sources.
- *Heterogeneous:* Answers can come from different sources using different data models and schemas.
- *Sources as Answers:* Answers can include pointers to sources where additional answers can be found.
- *Iterative:* User query interaction follows an iterative approach with the user posing a sequence of queries, each being a refinement or modification of the previous query.
- *Reflection:* Completeness of the query coverage and its accuracy is included in the answer.

- In situ*:* Answers can be references to, rather than copies of, the data.

Query Answering model
In order to support the above query answering model, dataspaces need to overcome a number of challenges [78].

Challenge 1: Develop a formal model for studying query answering in dataspaces

- *C1.1:* Develop intuitive semantics for answering a query that takes into consideration a sequence of earlier queries leading up to it.
- *C1.2:* Develop a formal model of information gathering tasks that include a sequence of lower-level operations on a dataspace.
- *C1.3:* Develop algorithms that given a keyword query and a large collection of data sources, will rank the data sources according to how likely they are to contain the answer.
- *C1.4:* Develop methods for ranking answers that are obtained from multiple heterogeneous sources (even when semantic mappings are not available).

Obtaining Answers
Within a dataspace, the answers to queries come from heterogeneous data that may use different terms at both the schema level and the data level. Dataspaces do not rely on semantic mappings, and even when mappings exist, they may be partial or approximate. This poses a significant challenge to answering queries in a dataspace [78].

Challenge 2: Develop methods for answering queries from multiple sources that do not rely solely on applying a set of correct semantic mappings

- *C2.1:* Develop techniques for answering queries based on the following ideas, or combinations thereof:
  - Apply several approximate or uncertain mappings and compare the answers obtained by each.
  - Apply keyword search techniques to obtain some data or some constants that can be used in instantiating mappings.
  - Examine previous queries and answers obtained from data sources in the dataspace and try to infer mappings between the data sources. Whenever we have access to queries that span multiple data sources, try to infer from them how the sources are related (e.g. the join attributes should provide some hint of common domains).
- *C2.2:* Develop a formal model for approximate semantic mappings and for measuring the accuracy of answers obtained with them.
- *C2.3:* Given two data sets that use the same terminology but for different data models, develop automatic best-effort methods for translating a query over one data set onto the other.

3.7.2 Introspection

The data in a dataspace will be uncertain and often inconsistent. The uncertainty will increase due to the best-effort query answering. Answers can be different, depending on the level of latency and completeness within the dataspace. It is often the case that inconsistencies lead to a very particular kind of uncertainty: which of a set of conflicting data values is correct. Both uncertainty and inconsistency need to be resolved, and lineage is often the only method available [78]. A dataspace needs to be able to introspect about lineage, uncertainty, and inconsistency.

Lineage, Uncertainty, and Inconsistency
The challenge is for a dataspace to provide a single unified mechanism for modelling uncertainty, inconsistency, and lineage [78].

Challenge 3: Develop formalisms that enable modelling uncertainty, inconsistency, and lineage in a unified fashion.

- *C3.1:* Develop formalisms that capture uncertainty about common forms of inconsistency in dataspaces.
- *C3.2:* Develop formalisms for representing and reasoning about external lineage.
- *C3.3:* Develop a general technique to extend any uncertainty formalism with lineage and study the representational and computational advantages of doing so.
- *C3.4:* Develop formalisms where uncertainty can be attached to tuples in views and view uncertainty can be used to derive uncertainty of other view tuples.

Finding the Right Answers
Given the challenges of lineage, uncertainty, and inconsistency of query answers in the dataspace, it becomes necessary to determine a "good" answer. Candidate answers can differ along multiple dimensions [78], including:

- Relevance to the query
- Certainty of the answer (or whether it contradicts another answer)
- Completeness and precision requested by the user
- Maximum latency required in answering the query

Challenge 4: Define metrics for comparing the quality of answers and answer sets over dataspaces, and efficient query processing techniques.

- *C4.1:* Develop query-language extensions and their corresponding semantics that enable specifying preferences on answer sets along the dimensions of completeness and precision, certainty and inconsistency, lineage preferences and latency.
- *C4.2:* Define notions of query containment that take into consideration completeness and precision, uncertainty and inconsistency and lineage of answers, and efficient algorithms for computing containment.
- *C4.3:* Develop methods for efficient processing of queries over uncertain and inconsistent data that conserve the external and internal lineage of the answers. Study whether existing query processors can be leveraged for this goal.

3.7.3 Reusing Human Attention

Every participant within a dataspace is provided with a basic level of service when they join it. Over time the dataspace should evolve by forming tighter semantic integration between participants as needed. One key mechanism to achieve this goal is to leverage users' attention as they interact with a dataspace. By analysing the interaction of a user with different participants within the dataspace, we can gain knowledge about the relationships between data sources [78].

Challenge 5: Develop methods that analyse users' activities when interacting with a dataspace and create additional meaningful relationships between sources in a dataspace or other enhancements to the dataspace.

- *C5.1:* Develop techniques that examine collections of queries over data sources and their results to build new mappings between disparate data sources.
- *C5.2:* Develop algorithms for grouping actions on a dataspace into tasks.
- *C5.3:* Develop facilities for explicit enhancement of dataspace information that give a high return on the investment of human attention.
- *C5.4:* Develop a formal framework for learning from human attention in dataspaces.

3.8 Dataspace Research Challenges

The dataspace approach to data management raises several research challenges that need to be tackled to create effective and efficient DSPs. Research challenges include:

- *Data Models, Search, and Query:* A dataspace needs to support the various data models and the different query languages of the participants with varying levels of query expressivity [2]. Research is needed to support a broad view of data modelling [83, 97, 108, 109] and to enable querying over the heterogeneous data models, from basic queries to context-based queries [110] and schema-agnostic question answering systems [111]. There has been limited work on addressing the requirements of real-time processing of events and streams and the investigation of relevant support services for dataspaces.
- *Discovery:* A key challenge within a dataspace is to locate relevant participants in the dataspace, identify relationships among participants, and improve the understanding of existing relationships among participants. [2, 107, 112, 113].
- *Reusing Human Attention:* The primary focus of research in this area looks to capture "user attention" to support management in the dataspace [114–117]. Leveraging user's attention for improving integration in dataspaces is considered an integral part of any dataspace application or platform [2, 78]. Roomba [118] exploits user feedback for improving integration of dataspaces using a decision-theoretic technique to quantify the desirability state

of a dataspace. DSToolkit [81, 119] uses end-user feedback for annotating, selecting and refining schema mappings, and for estimating the precision and recall of query results in a dataspace. MOBS [120] focuses on collecting feedback from many users to make a decision based on the combined result.

- *Storage and Indexing:* A key challenge for dataspaces is dealing with the heterogeneity of storage and indexing mechanisms of the various participants to create a uniformly indexed dataspace [2, 121]. Creating local storage and the placement of data and indices for optimal performance within a wide-area deployment are also active areas of research [122].
- *Correctness Guarantees:* Enabling access to a set of disparate data sources with confidence is a crucial challenge for dataspaces. To achieve this, there are a number of challenges in the quality of results, the effects and permanence of updates, and varying levels of service [2, 123], in a heterogeneous, highly autonomous environment.
- *Theoretical Foundations:* The concept of dataspaces opens several questions on their theoretical foundation. There is a need for research on the formal understanding of data governance [1], data models, relations among participants, and queries in a dataspace [2, 81].

3.9 Summary

In this chapter, we explored the challenges associated with data management and integration in the era of big data. Within large-scale integration scenarios that involve thousands of data sources, it is difficult and expensive to obtain an upfront unifying schema across all sources due to the challenge posed by the long tail of data variety. We detailed the dataspace paradigm as an emerging data management approach that embraces the notion of "good enough" and best-effort approximations as a means of data management. Data is integrated on an "as-needed" basis with the labour-intensive aspects of data integration postponed until they are required. Dataspaces reduce the initial effort required to set up data integration by relying on automatic matching and mapping generation techniques. When tighter semantic integration is required, it can be achieved in an incremental "pay-as-you-go" fashion by detailed mappings among the required data sources. This chapter detailed the fundamentals of the dataspace paradigm, including their core principles, comparison to existing approaches, support platform, support services, life cycle, and research challenges.