Object Centric Networking, from Database to Networked Objects

Hidden due to double blind review
Affiliation placeholder
email placeholder

***Abstract*—Data is key and is the raw material to produce services of the future. An enormous set of data is going to be generated by various sources such as cloud, objects, sensors, and humans. The issue is therefore to trade these data to feed the relevant applications and provide opportunities to business creation or public services. We argue for the development of a service that will provide a seamless mediation between applications/consumers and data/producers. Named Data Networking (NDN) purpose seems to fit this need. We applaud the idea to identify an object with a name.**

Nevertheless, such a matching is solely based on the naming and thus do not consider the nature and structure of the content. In this paper, we demonstrate the benefit of a network architecture where contents are represented by attributes and methods, namely managing multi-dimensional objects interconnected with semantic relationships. Our contribution is threefold.

First, we illustrate the value of bringing this information in the network. Second, we present a routing protocol allowing to transparently match the data produced to consumers' interests. Third, we show how relational algebra can improve data retrieval in this paradigm. We conclude by presenting implementation of our solution.

INTRODUCTION

The broad purpose of a network such as the Internet is to transparently match the users' interests with the resources it offers. In traditional IP networks, this means offering connectivity to remote hosts, identified by their IP address. The recent shift in usage patterns towards data and content motivates the introduction of new communication paradigms, and the move from a location-centric addressing, to a more content-aware network. We see the Named-Data Networking (NDN) approach (which regroups several proposals), as a first step towards such an objective.

The key point to build a service resides in the ability to discover data of interest (eventually distributed over several equipments), interact with this data, and ultimately combine it with semantically related additional information. That is why we aim at designing a network paradigm allowing to transparently trade data, as suggested in [1]. A network offering such primitives would drastically ease the development and the maintenance of a service, since accessing and manipulating remote data would be basically transparent. Offering such primitives require to have a good knowledge of the data produced by each equipment.

NDN approach proposes to organize data using hierarchical names (for instance films/Spielberg/1982/ET/chunkX). Such an approach is efficient to get a specific object, but does not allow to determine efficiently a group of objects (let say all the movies published in 1982): it would require to explore a large part of the hierarchy, or to maintain several hierarchies. Moreover, hierarchies does not allow to exhibit semantic relationship between objects (for instance the movies made by Spielberg). That is why we believe that NDN can hardly be adapted to offer the primitives we are looking for. Rather, we propose to represent data by using objects. This paradigm, named OCN (Object Centric Network) allow to represent a given object through multiple dimensions. This model can easily be extended to catch semantic links connecting the objects each other. Thanks to this additional information, the network can have good understanding of the data the consumers (final users, brokers) are looking for and the data offered by the producers (database, content storage, probes, sensors, clouds, etc.). Therefore, it can acts as a mediator by matching the consumers' interest with the data produced over the network. The ultimate goal of our approach consist in having a network acting as market place allowing to trade date [1], [2].

OCN benefits include those brought by previous ICN approaches. By nature, OCN allows consumers to subscribe to content published by some data providers, and reciprocally, a producer can contact a set of targets equipments as they acts as particular objects of the network. OCN hence natively offers a support of publish/subscribe and multicast. Note that such functionalities are not straightforward in host-centric network like IP, because it requires to always explicit which equipments are involved. This dependency requires complex workarounds (e.g. by deploying CDNs, establishing multicast tree) in order to offer a good performance.

Compared to previous ICN approaches, OCN offers a higher level of expressiveness in order to fetch data of interest. This also impact the core of the network, since each router can take advantage of this additional knowledge to perform in-network processing or to merge similar flows (for instance a flow including all the data carried by another flow). We expect that putting such functionality in the network itself could decrease the overall network load. However, having this additional knowledge may also implies scalability issue. We do not claim that OCN should bring the full connectivity to any object of the network, as IP does with public IP addresses. Rather, we believe that OCN would be more a set of overlapping communities, each of them gathering consumers, brokers and services sharing interest around a same category of objects.

However, building an OCN network comes with many new challenges. Indeed, the richness of services resides in their

ability to combine and compose various data sources, which are most of time heterogeneous and widely distributed. Indeed, OCN first class-citizens are data objects, which are by nature more complex than a name or an IP address. Build a novel communication paradigm in this context is not straightforward. We also expect that such a paradigm leads to interesting problems in various fields, for instance in traffic control (since the notion of flow is not the same as in IP), in security (how to trust data originator, preserve access to private data), or regarding the business models which could emerge.

Section I presents the state of the art of the current ICN paradigmes. Section II details the motivations of our proposal and an overview of our object model. In Section III, we explain how objects are traded and clarify the requirements of our architecture. Section IV details why relational algebra are well-suited to fit our needs and the related background can be used to design a query language over an OCN network. Section V presents a prototype of our architecture named Manifold and used in the context of the network measurements. We discuss a set of potential use cases and conclude this paper with some open questions in Section VI.

I. Related work

Many Information Centric Networks (ICN) architectures have been proposed over the last decades. Name-centric approaches generally aims at offering an abstraction regarding the content location and take advantage of it to optimize the stretch to reach the data. Such approaches requires to previously discover what is the name of a content of interest. Object-based network proposes to describe more accurately the data and thus to formulate more complex interests (basically all the contents matching a given clause).

Named-centric approaches: TRIAD [3] is the first attempt to build an ICN. It consists in identifying contents using user-friendly and location independent names. The name-to-address resolution is performed by dedicated content routers rather than by a DNS server. DONA [4] extends the TRIAD approach and propose a alternative to DNS (which mapping a name with an host) by offering a resolver able to match a user-friendly flat name with a content. 4WARD/SAIL NetInf [5] offers a pub/sub architecture based on DONA-style names. The content are described according to information object (IO) describing the content according to different level of abstraction or point of views. Each IO may point to one or more data object (DO) containing a related content. The level of indirection offered by IO increase the expressiveness of the interest which may be formulated. COMET [6] aims at offering a mediation plane aware in the network status in order to offer content delivery with guaranteed QoS. NDN [7] proposes a redesign of the routing architecture itself, where consumers request a content according to a name. This idea has provoked a wide interest in the research community.

Object-based approaches: PSIRP/PURSUIT [8] proposes to preconfigure rendezvous points in the network. Data publishers characterize their contents and push them on those rendezvous point, while subscribers contact those rendezvous points express their interest. Rendezvous points transmits the data to the relevant subscribers for instance using multicast. In CBN/CBCB [9], content are identified by a set of key value pairs. A user interest consists in a clause of predicates over those attributes. By contrast with named-based approaches, the forwarding is no more based on longest prefix matching. CBN assume that one (or more) forwarding trees are pre-established over the network. Each arc describe thanks to a clause which contents can be reached along the branch. CBCB takes advantage of this to only contact neighbors providing data of interest.

II. Objects in OCN

A network consists in ensuring a transparent access toward all its addressable resources. For instance, IP addresses allow to establish communication between two end-hosts in traditional networks, whereas NDN transparent access to a content using a name. OCN tries to go beyond NDN by using objects to play the role of addresses. It implies that objects are also to identify the entities participating in the network.

A. Requirements

1) The need for a multidimensional address space: IP networks are by nature host-centric and offer *connectivity* to end-host. Such networks are by nature heavily equipment-centric and thus unaware of the data produced or transport by the network. However, most services are on the contrary data-centric: they are more concerned with the data rather than its location and with the performance to access this data if it is offered by several equipments. This increasing needs has led to build indirection layers on top of the IP networks, such as Peer-to-Peer or Content Distribution Networks (by using DNS indirection or anycast addresses).

To address this limitation, NDN [7] proposes to identify each content using a name and offer native caching in the router. By doing so, the authors expect to support a native and efficient per-chunk of data caching and thus improve the overall performance. A large number of contents implies a large number of names. To remain scalable, the authors propose to group similar contents according to an arbitrary hierarchy. NDN routers can therefore maintain a routing table based on a prefix-tree and hence significantly decrease the memory consumption while offering an efficient lookup. However in [10], the authors pointed out the difficulty to apply a hierarchical aggregation in ICN at the Internet scale with today's technology. Another issue resides in the limited semantic offered by hierarchical names to discover contents of interest provided by the network.

In CBCB [11], authors propose to represent each content by using a tuple of attributes. Network destinations are defined using a clause involving object's attribute. Describing objects using multi-attributes object enable the emergence of a powerful query language, since it allows to specific content according to a set of predicates. To do so, we need a better understanding of the nature of the content and thus require to describe contents and services through multiple attributes. Following this idea, we also believe that naming is not sufficient to characterize accurately a resource. We propose to go beyond tuple by using objects instead.

2) The need for semantics: The major purpose of OCN consist in helping new services and thus providing a transparent access to the data of interest. However tuples are not

well-suited to describe the semantic, and that is why we propose to adopt objects instead. This approach has already been proposed in particular case including in Pub/sub [12], P2P overlays [13] and sensor networks [14]. We investigate in this paper how to generalize this idea. Objects can bring (compared to tuples) an additional semantic through typing and inheritance. More precisely, types establish a link toward the corresponding object, whereas inheritance indicates that an object is a specialization of its parent object. This added information can be exploited to dynamically find contents semantically related and at last combine related contents. In this paper we will only consider semantic relationship resulting of the *typing*. We will explicit in the next sections how we model such semantic links.

3) The need for a common referential: OCN aims at becoming a kind of market-place allowing several autonomous authorities to trade objects. It appears clear that OCN requires at least some loose coordination between them, as far as naming is concerned. The main purpose of this referential is to guarantee that a given object name (resp. attributes' name) always has the same meaning over all the considered authorities, as well as a public IP address in Internet. If this hypothesis is violated, an OCN network could return incomplete (resp. wrong) results by missing some relevant the content not named appropriately (resp. by combining unrelated content). We assume that for a given object, authorities do not expose contradictory values. This hypothesis implies for instance that authorities adopt a same compatible unit system for a given numerical attribute.

B. Modeling objects in OCN

1) Representing objects by using the relational model: OCN requires a data model that is both simple and expressive enough to offer a thin and powerful interoperability layer between autonomous authorities sharing objects. The relational approach brings a solid mathematical framework to represent and process multi-dimensional data. That is why we propose to adopt a model inspired from the database background. Although the relational model is not well-suited to all types of data (such as recursive data structures), we believe that it offers a satisfactory compromise between simplicity and efficiency. In the rest of this paper we use the database terminology and make analogies with databases every time it is possible.

Note that relational model is only used to define a common language between the authorities: we do not plan to store data in databases. Each authorities remain free to use its own storage system (as explained in Section III). We just expect that they are able to handle queries corresponding to the collection of objects it exposes (which implies to translate them according to their storage they use internally or their services they provide).

The main advantages with traditional database resides in our ability offering:

- *Logical independence:* by taking advantage of the semantic links, we can transparently combine objects in a correct manner
- *Physical independence:* by providing an abstraction over the location of the data, the underlying equipments ensuring the service are transparent to the user(like in cloud or database clusters)
- *Storing independence:* through this common language, we are not impacted by the technical choice of the underlying infrastructure. This last aspect set of involved servers is transparent to the user. The third aspect reminds LINQ [15], a framework allowing to perform query inspired from database background over a collection of objects of various natures.

2) Connecting and characterizing objects with semantic dependencies: Database community has exhibited several kinds of dependencies, and in data model we only need three of them to describe our semantic relationships. We denote by A the set of attributes of a table T. Basically, tables correspond to a collection of objects while records corresponds to a given object.

1) *Functional dependency (FD)* is very close to the notion of *key*, i.e. a subset of attributes $K \subseteq A$ which uniquely identifies a record in T. More formally, $K \underset{FD}{\mapsto} A$ indicates that for each record of T, the value of attributes in K defines the value of all the attributes in A. Keys can be used to identify a given object, or to establish a relation with it (including in the case of varying attributes as we will see). Keys can also be exploited by an OCN node to detect (and eventually discard) duplicated objects it collects for a given interest.
2) *Relational dependency (RD)* describes a relationship between records of a table T with records of table T'. In OCN, RDs are described by *typing* attributes. An attribute A of a table T refers to a table T' if it is of type T'. They are use to combine consistently two related objects. Since typing is thus our raw material to discover semantic, we require that authorities providing related objects conform to the same typing and naming conventions.
3) *Join dependency (JD)* indicates how to split a given table T into several subtables $T_1, \ldots, T_n$ that can be joined without inducing extra records ($T = T_1 \bowtie \ldots \bowtie T_n$).

In OCN, FDs and RDs are explicitly declared and will be illustrated in Section II-C. JDs are not needed to define our object model, but will be useful (and dynamically discovered) to build our routing table (see Section **??**).

C. The OCN Object model

The outcome Section II-A leads us to consider an object model conciliating both the multidimensional objects and the need for semantic requirements. To do so, we propose an object model inspired from Object Oriented programing and database background, as illustrated on Figure 1. In this example, `name` is a key of the `Country` object and it is the only one (FD). Here, a Movie's country refer to a Country using the corresponding `name` (RD).

This textual representation is enough generic to be used for various collection of objects. It can be used to wrap a database (without regard of the underlying SGBD), a text file (csv, etc.), or even a web service (for example a service returning the country and the year for a given input movie). We could extend this syntax to support more complex declaration (for instance a collection of objects involving arrays, read-only attributes,

```
class Country {
    String  name;
    Int     population;

    KEY(name);
}

class Movie {
    String title;
    Country country;
    Int    year;

    KEY(title);
}
```

Fig. 1. Textual representation of an object type

methods and inheritances) and then tend to objects declarations conforming to these done in Oriented Object Programming. For sake of simplicity, we will omit those consideration and only consider simple typed structures in this paper.

D. Notations

In the next sections, we will use more formal notations. We denote by $\mathcal{O}^T$ the set of objects characterized by the tuple (T, A, Σ), where:

- T corresponds both to the object type and the object name;
- $A = (a_1, \cdots, a_k) \in \mathbb{S}^k$ is the corresponding set of attributes[1];
- Σ holds all semantic dependencies defined in our object;

For a given object $o \in O^T$, we also define the *type* operator: $\mathrm{T}(o) = T$, as well as the *attribute set* operator: $\mathrm{Att}(o) = A$. For $a \in A$, $\mathrm{Dom}(a)$ represents the domain of the attribute, and corresponds to a type declaration (either a base type or an object type). We denote by $\diamond$ the `NULL` value, which belongs to any attribute set.

This model captures the multidimensional addressing space requirement (with A) and the semantics requirement (with Σ) evoked in see Section II-A). Σ and A play a central role to perform routing and forwarding in OCN. Indeed, an interest in OCN may involve several objects semantically related. In one hand, Σ is needed by the routing plan to discover all the objects corresponding to this interest (see Section **??**). In the other hand, A is mostly be used to determine in the forwarding plan which attributes (dimensions of interests) are served by each neighboring router (see Section **??**).

III. Sharing objects in OCN

An OCN network consists of several autonomous authorities willing to offer services by sharing collection of objects. Their corresponding infrastructure is made of OCN routers transporting packets. Since OCN aims at being a convenient support to trade objects, source and destination addresses are naturally collections of objects.

[1]We denote by $\mathbb{S}$ the set of strings allowed to define type and attribute names.

Our objective reminds the Internet architecture, and this leads us to adopt a similar design. Indeed, the Internet is made of several interconnected Autonomous Systems (or AS) offering reachability to Internet IP addresses. Each AS bounds an IP network ruled by a single authority. To access Internet destinations exterior to the AS, router have to learn how to contact them. To do so, ASes have to exchange routing information thanks to a *common language*. Note that this mediation, today achieved by BGP [16], does not affect the way the routing is performed inside the AS. A BGP router announces a set of IP destinations to a neighboring router it is able to serves. By doing so, it accepts that the traffic toward the corresponding destination issued by its neighbor. Announcing an IP destination is rules by the BGP routing policy configured on the router. A BGP router collecting several announces toward a same destination elect the one it prefers according to the BGP metric carried in the announcement and its routing policy. Iteratively, it can decide to forward this best route to its own neighbors (depending on its routing policy). BGP routing policies rules how the IP destinations are traded while preserving the *authority* of network operators on their own network. In practice, BGP announcements are ruled by *mutual interests*. For example, a client AS will typically pays its providers ASes to get the connectivity to all the Internet destinations. Indeed, BGP routing policy expressiveness is enough rich to allow to encode complex ecosystem ruled by business relationships. For this reason, routing policies are often kept private, and BGP routers are thus only aware on the IP resources exposed by their neighbor, but have do not maintain a complete map of the Internet. Hence BGP offers a better *scalability* while preserving routing policy *privacy* of each AS. The chain of AS involved in a BGP path corresponds to a chain of actors finding an interest to forward the traffic toward the corresponding IP destination.

By adopting a similar architecture, we benefits of years of practical experience in operating such networks and a wide range of theoretical tools that can be transposed to analyse the correctness and performance of protocols that we can build on and extend. However, sharing objects rather that addresses identifying equipments over the network has strong implications. The following sections propose an overview of the major differences that we have identified and which will lead to our router architecture.

For sake of simplicity, we will assume in this section that routers exchange *homogeneous* collection of objects. By homogeneous, we mean that for a given object T all the actors expose the same attributes A and the same semantic links Σ. We will see later how to relax this hypotheses.

A. Distributed objects and replicas

The meaning of the announcement in BGP and OCN slightly differs. In BGP, the goal consist in establishing a path toward a given machine having the corresponding IP address. The only role play by intermediate nodes resides in their forwarding ability.

In NDN, router can cache content, and thus a target name can corresponds to a content potentially cached on various machine. However this name is supposed to always correspond to the same content and can be routed toward any of those replicas.

In OCN, the problem is more complex. If a node announce a collection of objects it can mean that it can contact peers offering this collection, or also because it is can offer a specific subset of the corresponding objects. As a consequence, even if a node holds content of interest, contrary to NDN, it may still have to forward this query to its peers. In general, several sources might present overlapping collections or replicas of another source. OCN routing consists in establishing a forwarding tree allowing to contact all of them. When retrieving object, this tree is exploited to aggregate and combine the corresponding target objects.

This major difference raises new challenges that didn't occur in IP nor NDN. If we consider a single target object collection, we have identified three main problems:

1) *Persistent reannouncement:* so far, nothing prevent an announcement in OCN to be forwarded along a cycle of OCN routers. In BGP, such issue can't occur because the announcement carries the set of traversed AS. Hence a router can discard a announcement if it forms an inter-AS loop. In NDN, we can expect that paths having loops are always longer that the corresponding loop-free path. In OCN we also require a mechanism preventing such loops. However, we have to manipulate a more complex object (a tree) than in NDN or IP (a path) and we aim at preserving physical independence.
2) *Multiple replicas:* several nodes may provide exactly the same content. OCN should be able to detect it. We could then be able to retrieve the corresponding content only once, or to detect whether we can perform load-balancing over the two corresponding nodes (for instance by fetching an half on the first node and the rest on the second node).
3) *Multiple forwarding paths:* the announcement corresponding to a given collection and a given node can be forwarded along several paths. As a result a given node could learn from different neighbors the same object collection without detecting they corresponds to the same objects. As a result, the content could be fetched several times.

We will propose in Section **XXX** several mechanisms preventing such issues **(origin, maintain pending query ID on node, maybe we could maintain a SPT instead of AS paths?)**

Now, consider that a given interest involves the combination of several collections. It may exists several possible way to combine and aggregate objects to get the same result. We call the corresponding strategy a *query plan* to conform to the database terminology. Since the number of collections may be high, the number of possible query plans can explode. We cannot pre-compute all the corresponding routes of them and install directly in FIB (as done in BGP or NDN). Section **??** and **??** present our methodology, which consist in splitting a part of this process in the routing plan (determine relevant collections for each peers) and the in the forwarding plan (which determine determine the query plan once the collections and peers of interest have been determined).

B. Destination address aggregation

Due to the potential large number of destinations managed by a network, the address space must provide mean to gather set of destinations. IP and NDN address space are mono-dimensional spaces.

IP space is split by allocating contiguous range of IP addresses called IP prefixes and corresponding to addresses starting with a same sequence of bits. Note that the some organisations controlled the way the IP addresses are assigned. In practice this allocation reflects the underlying network topology since network operators buy ranges of IP addresses. By nature, IP prefixes are a good way to gather closes IP equipments in a single routing table entry. This aggregation is used to scale the routing table of the equipment and provide relevant and enough accurate view of the surrounding IP ecosystems. In a home network just few routes are required (those of the LAN, and a default route) whereas core routers handle thousands of IP prefixes. In case of ambiguity (e.g. if several prefixes contains the target IP address), the IP equipment uses the route having the most specific prefix is used. Prefixes also comes with algorithms for example based on prefix-tree to perform efficient lookups over a routing table.

NDN proposes to adopt a similar approach by grouping similar content using name prefixes. A prefix thus design a set of similar contents. Due to the high volatility of the contents, a node announcing a prefix can hardly guarantee that its ability to serve the whole prefix. In that sense, NDN prefixes lead routers to *over announces* the set of destinations they announce. Note that this issue didn't occur in IP due to the way the IP addresses are allocated. If several prefixes correspond to the target destination, NDN have to choose a strategy to determine which peers of interest must be contacted. The status of the routing is in NDN as far as we know not clearly stated and several strategy are envisioned, including (i) routing the content toward the most specific name (like in IP, but without guarantee to reach the destination), (ii) contact randomly a candidate peer and assign a weight to this route depending on the result, (ii) contact all the peers of interest (flooding). Note that as mentioned in Section III-A, flooding can be very costly since the same content can be fetched several time.

Like NDN, OCN faces the so called "over-announcement" issue. However, OCN address space offer more flexibility in terms of address aggregations, since it does not ruled by an arbitrary hierarchy, like in NDN. Roughly, we hope that flexibility will lead to more accurate aggregate of destinations and thus decrease the impact of over-announcements, while offering to tune the accuracy and the size of the routing table of the OCN routers.

C. Global vs partial connectivity

Services are most of time only interested on a small fraction of objects. Contrary to P2P networks, node will only exchange to a given neighbor the objects they are both concerned to. Typically an OCN node will *provide* a collection of objects to a *customer*. We remark though that sharing a subset of destination is already achieved in Internet, where each AS only advertises through BGP routing policies a subset of the IP destinations it knows for instance due to following economic incentives [17]. Similarly, OCN routing policies will encode the way the objects are traded depending on the peering agreement established between between neighboring ASes and drastically the number of collections handled by a single router.

This will contribute in part to the scalability of the network in term of the number of processed objects.

By playing the role of address, objects are also use to design a data consumer. More precisely, returning a content to a consumer consist in inserting it into the client's collection. Note we could also envision to populate a third collection. Anyway, data providers do not necessarily collect announcements corresponding to the corresponding consumer objects and thus might be unable to reach them. That is why by default, the corresponding packets will be returned along the reverse path. The network thus needs to provision for a return paths for packets. We adopt in OCN a similar approach as NDN, but rather to manage a PIT at the packet granularity, we can in OCN manage a PIT at a flow granularity and thus offer a better scalability.

D. Summary

The major purpose of OCN is to offer a mediation between a set of data consumers and a set of data providers. Some actors can play the both roles like in real life. In one side, OCN producers sell their objects they can serve. In the other side, consumers buy access to objects of interests by issuing queries. The OCN network play the role of transparent interfaces between the consumers and the producers. By transparent, we mean that consumers are not interested on how the providers store and organize the information *(logical independence)* of interest nor on their location *(physical independence)*.

Consumers express their interests to the OCN network by issuing *query packets*. The underlying query language must be enough expressive to accurately describe the consumers' needs. The network has to dispatch and eventually split this query to reach all the producers providing objects of interests. It means that the OCN nodes offering primitives allowing to do so, which means computing an efficient query plan, and process the corresponding reply packets. This query plan is distributed over the network, forming a tree from a consumer toward all the producers of interest. Depending on this query plan, the intermediate nodes process the data thanks to a set of *primitives* and can aggregate objects having the same type, combine objects semantically related, and filter irrelevant data. This processing is quite simple and can be performed at high speed. When a producer handles the corresponding query, it translates it into its in order to query its infrastructure consequently. It returns the result through *reply packets* which by default followed the reversed tree.

We will see that the three challenges we've exhibited in this section have strong implications in terms of design and lead the router architecture we propose.

IV. A DATABASE PERSPECTIVE FOR NETWORKS

A. Motivations

From an external point of view, the OCN architecture we have introduced presents slight differences with traditional IP networks. The major change is that IP addresses are substituted with objects, which induces the consequent differences presented in the previous chapter. In addition, we have considered that the different end hosts announce fully homogeneous and non-overlapping objects, which will be hard to realize in real environments, and thus require us to relax this hypothese. A notable characteristic of addresses and packets in our architecture is that objects and their related content will require some processing from the network, and thus appropriate routing and forwarding algorithms. We want to investigate whether such approach is feasible.

Fortunately, relational algebras, introduced by Codd in XXXX, and since then widely studied in the database community, provided a theoretical framework that can be easily extended to manipulate our notion of objects. In addition, we will see how fundamental results from the database community can help us proposing efficient algorithms for routing and forwarding.

B. Notations and formalism

In order to present our routing and forwarding mechanisms, we need to formally introduce a set of concepts.

1) Objects and semantics: Given an object $o \in O$, we denote $(a_1, a_2, \ldots, a_m) \in \mathrm{Dom}(\mathrm{Att}(0))$ the values of its attributes, and $t = \mathrm{T}(o)$. o can be equivalently represented by its tuple representation:

$$o \sim (t, a_1, \ldots, a_m)$$

We suppose we have tables = objects attributes. Now semantics:

Let $O, O' \subseteq \mathcal{O}$. The left join of O and O' requires that a semantic relationship exists between these objects, expressed by a functional dependency between their attributes:

$$\mathrm{Att}(O) \underset{FD}{\mapsto} \mathrm{Att}(O') \Leftrightarrow (\exists A \subseteq \mathrm{Att}(O)) \left(A \underset{FD}{\mapsto} \mathrm{Att}(O') \right)$$

objects have to be well defined, we cannot invent them. respect the semantic.

2) Collections: A *collection* c is a consistent set of objects $c \subset \mathcal{C} = \mathcal{P}(\mathcal{O})$: they should have the same type and an equivalent set of attributes.

3) Left relational algebra: We define a *left relational algebra* operating on collections, denoted $RA_l = (\mathcal{C}, \cup, \ltimes, \sigma, \pi)$.

The first two are binary operators allowing to combine objects according to some semantic dependencies Σ which we start assuming are known:

- Union ($\cup$) is used to merge consistent collections $(c_i)_{i \in I}$ and corresponds to the traditional set union.
- Left join ($\ltimes$) complements a collection c with collections $(c_i)_{i \in I}$ by extending the set of attributes for each object in c with attributes from semantically related objects in c_i. The use of *left join* is motivated by the preservation of object types in c.

Shall we give more extensive definitions and examples for these operators ?

The *selection* (σ) and *projection* (π) operators are doing simple processing tasks on collections. Selection filters out objects not matching a predicate p on object attributes (we

have in fact σ_p, and $\sigma_p(c) \subset c$). Projection allows to restrict the set of attributes of a collection from A to $A' \subset A$ (and the full notation is $\pi_{A'}$).

The main difference with traditional relational algebras as introduced by Codd [**?**] is the use of left join, and as such we invite the interested reader to refer to the abundant literature in the domain for more details informations (for instance [**?**]).

C. Populating the routing table: semantic inference

Objects received in announces

semantic inference some implicit some explicit

a part done at forwarding time

D. User queries and routing

unique naming assumption ???

E. Handling heterogeneity thanks to database normalization

The relational algebra we have defined suppose objects are unique and well-defined. If we expect to use this framework to model a wide range of Internet data originating from autonomous entities, we need to relax some of these requirements and allow for some degree of heterogeneity in the way the different platforms represent their own data. This means we will have to reconciliate the semantics of the different platforms.

a) Attribute heterogeneity: Let's first assume a simpler situation where two objects of the same type have different sets of attributes A and A'. Based on the *open world assumption*, we consider that the value of missing attributes are simply unknown or not revealed by the source of the object and set their value to $\diamond$, which allows us to perform the union.

b) Key heterogeneity: If the two objects have the Now about keys

No duplicates, so what if objects with the same key, we need some merge of objects. somehow similar to join. We cannot if no common key

Conflicts with uniqueness assumption join = view (pi + split) a quoi sert le split common attributes with another object = normalization

inheritance ???

F. A fundamental assumption

G. Database notions

maximum contained query answering queries using views

normalization

naming and universal relation ?

H. Packet-level behaviour and operator implementation

Collection = a series of packets.

operators can add fields to function (keys for dup, join, etc)

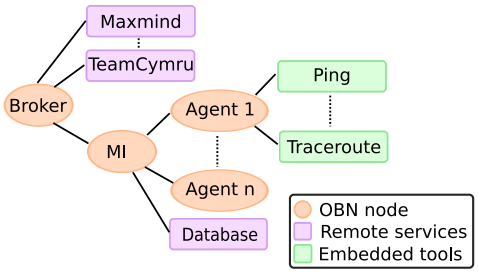

Fig. 2. A OCN ecosystem dedicated to federate network measurements.

I. Query plan optimizations

algebraic

partitions

transformations (cartesian product, right join, etc.)

more : subqueries

V. A prototype for OCN

This section presents how we have built a broker relying on the OCN paradigm and offering monitoring data produced by different authorities.

Architecture: Figure 2 depicts the ecosystem of producers and consumers involved in this architecture:

- *Maxmind* offers an API able to return for a given input IP its geolocation, the corresponding city and the corresponding country.
- *TeamCymru* exposes information mapping an IP with a corresponding BGP announcement such as the corresponding prefix and ASN of its origin AS.
- The *controller* federates a collection of OCN nodes (corresponding to PlanetLab nodes in our use-case) performing periodical and on-demand network measurements.

In Figure 2, ellipses stands for equipment supporting OCN, whereas rectangle represents source of data wrapped in dedicated translators making them OCN-compliant. These translators allow the OCN ecosystem to inter-operate with the legacy services.

Routing plan: Each measurement agent announces the objects corresponding to the network measurement tools it supports. Each of them can push periodical network measurements in a database which can be later be queried by the controller. The controller re-announces the `Ping` and `Traceroute` objects to the broker, and thus hide the measurement architecture complexity to the broker. The broker learns announcements issued by the controller. The announces representing the `Ip` object provided by the Maxmind and the TeamCymru translators are statically configured in its RIB. According to those announces, the broker computes the normalized shown in Figure 3. Solid arrows depicts the specifics view provided by each neighbors, whereas dashed arrow represents the semantic links deduced thanks to typing and achieved thanks to a common identifier (here, an IP address). In the similar manner, the controller stores a schema made of the `ping` and `traceroute` objects served by the agents and the database.

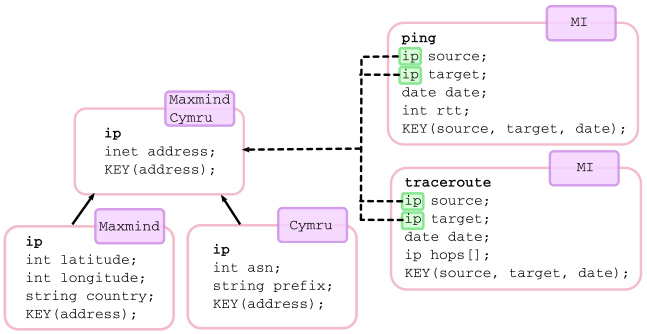

Fig. 3. The broker's RIB contains the schema of objects it perceives.

Extending the monitoring architecture: Note this architecture allow to plug new services and new measurement agents easily, as illustrated by dashed line in Figure 2. If the number of measurement agent become large, scalability issue may arise. We straightforward solution consist in design a hierarchy of controllers. Another nice approach would consists in offering a self-organized overlay of agent guaranteeing that the degree of each agent and of the controller remain small, as proposed in [18].

Forwarding plan: The user can visualize monitoring information through a web portal which queries the broker to retrieve the information to display. Depending on the timestamp (a given date or the `now` keyword) formulated in the input query, the controller can determine whether it corresponds to an on-demand or an archived measurement, and will consequently either query the database or either query the relevant set of agents. It simply requires that the announcements corresponding to the objects stored in the database specify that the timestamp attribute is different of `now`.

Expressiveness brought by OCN: A major benefit of OCN is its ability to hide the complexity induced by the underlying sources of data used by the queried node. Suppose that measurement agents are deployed over PlanetLab [19]. OCN allows the broker to issue high scale and world-wide measurements with very concise query, as demonstrated the two queries proposed in **??**:

- The first query returns a set of AS paths deduced from a set of live traceroute measurements, and could used to enrich public BGP datasets, actually only perform by a small set of BGP trace collectors.
- The second query performs delay measurement from each node toward the the 8.8.8.8 anycast IP and can be used to discover the underlying anycast architecture [20].

One may imagine various other world-wide and high scale measurement campaign to detect other phenomenon like black holes (by probing periodically from various node a given monitored IP address), censorship (by performing http request all around the world), impact of major outage like earthquakes (by joining network measurement and geolocation information).

```
SELECT  source, target, hops.asn
  FROM  traceroute
  WHERE timestamp == now

SELECT  source, rtt
  FROM  ping
  WHERE timestamp == now AND target == 8.8.8.8
```

Fig. 4. A simple query in OCN

VI. Discussion

VII. Conclusion

In this paper, we have presented a new paradigm of Information Centric Network, called OBN, where content are described and manipulated as Objects. We have shown it is possible to offer a high level of query language over a network while abstracting the content locations. We have demonstrated how OBN paradigm could be used to trade data by doing a parallel over an ecosystem of actors providing data and services. We have proposed a routing protocol inspired of BGP allowing to encode complex business relationship, resilient to network and content dynamics. We've have proposed some technique to reduce the routing table size and optimize data retrieval. We've shown that a pub/sub architecture can be built above the OBN architecture and offer a native generalization of multicast. We have developed an open-source implementation of OBN. We've demonstrated through a real use case (a broker over monitoring data and tierce services provided by different authorities) the tractability of our proposal.

Not considered:

- Inheritance

References

[1] S. Fdida and M. Diallo, "The network is a database," in *Proceedings of the 4th Asian Conference on Internet Engineering*. ACM, 2008, pp. 1–6.

[2] J. Hellerstein, "Adaptative dataflow: A database/networking," 2000. [Online]. Available: http://db.cs.berkeley.edu/jmh/talks/DBNetsConverge.ppt

[3] M. Gritter and D. R. Cheriton, "An architecture for content routing support in the internet." in *USITS*, vol. 1, 2001, pp. 4–4.

[4] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 181–192, 2007.

[5] B. Ahlgren, M. D'Ambrosio, M. Marchisio, I. Marsh, C. Dannewitz, B. Ohlman, K. Pentikousis, O. Strandberg, R. Rembarz, and V. Vercellone, "Design considerations for a network of information," in *Proceedings of the 2008 ACM CoNEXT Conference*. ACM, 2008, p. 66.

[6] G. García, A. Beben, F. J. Ramón, A. Maeso, I. Psaras, G. Pavlou, N. Wang, J. Sliwinski, S. Spirou, S. Soursos *et al.*, "Comet: Content mediator architecture for content-aware networks," in *Future Network & Mobile Summit (FutureNetw), 2011*. IEEE, 2011, pp. 1–8.

[7] V. Jacobson, M. Mosko, D. Smetters, and J. Garcia-Luna-Aceves, "Content-centric networking," *Whitepaper, Palo Alto Research Center*, pp. 2–4, 2007.

[8] N. Fotiou, P. Nikander, D. Trossen, and G. C. Polyzos, "Developing information networking further: From psirp to pursuit," in *Broadband Communications, Networks, and Systems*. Springer, 2012, pp. 1–13.

[9] A. Carzaniga and A. L. Wolf, "Forwarding in a content-based network," in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2003, pp. 163–174.

[10] A. Narayanan and D. Oran, "Ndn and ip routing can it scale," in *Proposed Information-Centric Networking Research Group (ICNRG), Side meeting at IETF-82, Taipei*, 2011.

[11] A. Carzaniga, M. J. Rutherford, and A. L. Wolf, "A routing scheme for content-based networking," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 2. IEEE, 2004, pp. 918–928.

[12] Y. Jin and R. Strom, "Relational subscription middleware for internet-scale publish-subscribe," in *Proceedings of the 2nd international workshop on Distributed event-based systems*. ACM, 2003, pp. 1–8.

[13] C. S. D. UC Berkeley, "Pier project." [Online]. Available: http://telegraph.cs.berkeley.edu/tinydb/

[14] TinyDB, "Tinydb." [Online]. Available: http://telegraph.cs.berkeley.edu/tinydb/

[15] E. Meijer, B. Beckman, and G. Bierman, "Linq: reconciling object, relations and xml in the. net framework," in *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*. ACM, 2006, pp. 706–706.

[16] Y. Rekhter and T. Li, "A border gateway protocol 4 (bgp-4)," United States, 1995.

[17] L. Gao, "On inferring autonomous system relationships in the internet," *IEEE/ACM Transactions on Networking (ToN)*, vol. 9, no. 6, pp. 733–745, 2001.

[18] C. Peplin, "Massively distributed monitoring," 2011.

[19] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "Planetlab: an overlay testbed for broad-coverage services," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 3–12, 2003.

[20] "Name placeholder." [Online]. Available: URLplaceholder