Poster: Concept of Multiscoping for Distributed Event-Based Systems

Léon Lim and Denis Conan
Institut Mines-Télécom, Télécom SudParis
UMR CNRS SAMOVAR, Évry, France
<firstname>.<lastname>@telecom-sudparis.eu

ABSTRACT

Distributed Event-Based Systems (DEBS) provide a versatile solution for asynchronously exchanging data in a distributed system, loosely-coupled in space and time. In this work, the software architecture of a DEBS is composed of an overlay network of brokers that are responsible for routing data from producers to consumers. Important issues are the cost (in terms of exchanged messages) of the installation of routing filters (for advertisements and subscriptions) on the brokers, and the cost of routing notifications. In this work, we extend the usage of the concept of scope to propose the concept of multiscoping: the overlay of brokers is logically structured according to several dimensions (geographic location, network characteristics, client membership, etc.) with visibility filters installed at scope boundaries; these overlays are superposed; clients connect to brokers and express their scoping requirements on these dimensions; and distribution of notifications is controlled both by visibility and routing filters.

Categories and Subject Descriptors

C.2.4 [**Computer Systems Organisation**]: Computer Communication Networks—*Distributed Systems*; C.2.5 [**Computer-Communication Networks**]: Local and Wide Area Networks—*Internet.*

General Terms

Algorithms, Design.

Keywords

Middleware, Distributed Event-Based Systems, Scoping.

1. INTRODUCTION

Among the many middleware approaches for designing emerging systems in the context of the Internet of Things (IoT) [1], the publish/subscribe communication model [3] that is brought to play in Distributed Event-Based Systems (DEBS) [6, 9] belongs to the most popular solutions to address sense and respond applications. Producers are clients that publish notifications whereas consumers are clients that react to notifications delivered to them by the system. An advertisement describes a set of notifications a producer is willing to publish. Producers initiate the communication but they do not know any consumer. A subscription describes a set of notifications a consumer is interested in. If a notification matches a subscription, it is delivered to the consumer.

DEBS solutions are originally typically implemented as overlay networks of brokers [10]. The access broker of the consumer is responsible for installing the subscription filter on all the brokers of the overlay network so that notifications that match the subscription are routed towards the consumer. In the context of broad IoT, many brokers may be involved and many messages may be exchanged when installing advertisement and subscription filters, and when routing notifications. The cost in number of exchanged messages is an important issue in wide-area DEBS and especially in heterogeneous systems involving for instance clouds, cloudlets, desktops, laptops, mobile phones, and smart objects of the IoT. In these systems, we can observe that system heterogeneity comes from device heterogeneity but also from the different technologies used for networking, from the end-users characteristics such as social membership, geographic location, etc. When dealing with one type of heterogeneity, e.g. network heterogeneity, and one characteristic, e.g. network range, it is interesting to distinguish several ranges of levels or scales, e.g. PAN, LAN, WAN scales. Then, a distributed system that takes into account several of these scales is said to be a multiscale distributed system [2, 5, 8, 11].

Clearly, some of the sources of heterogeneity impact distributed routing. In this work, we claim that heterogeneity is also a part of the solution. For example, when a consumer declares an interest in the weather and living conditions of travelling family members on vacation abroad, the brokers may deduce that some parts of the overlay network are not concerned. In other words, some sources of heterogeneity delimit logical spaces and these spaces delimit scopes of data distribution, with notifications being visible only into certain scopes.

We extend the concept of scope of [4] to propose the concept of multiscoping: administrators structure the groups of brokers according to different dimensions (geographic localisation, network range characteristics, client membership,

DEBS'15, June 29 - July 3, 2015, Oslo, Norway.

DOI: http://dx.doi.org/10.1145/2675743.2776768.

etc.) with visibility filters installed at scope boundaries; clients connect to brokers and express their scoping requirements in advertisements and subscriptions; and distribution of notifications is controlled by visibility and routing filters. To that end, we extend the requirements of distributed routing to deal with multiscoping.

2. DEBS WITH MULTISCOPING

2.1 Concept of multiscoping

The concept of "scope" is used for structuring publish/-subscribe systems by putting the concept of visibility of notifications forward. The visibility of a notification limits the set of consumers that may get access to this notification. We take the definition presented in [4]: "A scope is an abstraction that bundles a set of clients (producers and consumers) in that the visibility of notifications published by a producer is confined to the consumers belonging to the same scope as the producer; a scope can recursively be a member of other scopes". The concept of "multiscoping" is to consider several disjoint sets of scopes, one set of scopes per dimension —i.e. per measurement of a particular characteristic of a particular architectural viewpoint [7].

As displayed in Figure 1, we illustrate the solution with a context management service, and more particularly with context data from the IoT. The dimension membership of the viewpoint user is important when considering context data distribution. With the IoT, context data are used locally (e.g. in a smart space), and also remotely (e.g. in another administrative area). The dimension administrative area of the viewpoint geography is thus interesting for tagging subscriptions and forwarding notifications to specific areas. In addition, the scales of the dimension network range of the viewpoint network can help in deciding where to perform context data aggregation to limit broadcasting: e.g. in the local WiFi network on-board the bus or in the Intranet of the transportation operator, or even in a public Cloud.

We complement the API of regular DEBS with the management of multiscoping. Before advertising or subscribing, system administrators "partition" the system into scopes by tagging brokers. This is done through the new actions JoinScope and LeaveScope. For instance, the system administrator may tag some brokers with the scope Europe, then others France, next Bordeaux, etc. When clients submit an advertisement or a subscription, they specify the scopes, at most one per dimension. A producer may provide context data for the scope of Bordeaux. Thus, her notifications may be forwarded only in the scope of Bordeaux: This is the role of distributed routing to limit the dissemination. In other words, the idea is that an overlay of brokers joins together producers and consumers that specify the scope Bordeaux. Secondly, scopes are organised into hierarchies and this is the role of the visibility filters of a scope to constrain the condition for the forwarding to subscopes and superscopes. Roughly speaking, notifications are visible to a client if they are visible to that client for every scope of every dimension of the set of scopes specified in the advertisement. A third impact of multiscoping is that subscription filters are only installed on brokers belonging to visible scopes.

2.2 Visibility in a dimension—Monoscoping

By convenience, we add to the set of scopes of a dimension the scopes bottom ($\perp$) and top ($\top$). The membership relation "superscope" between scopes in a dimension d is defined by a directed acyclic graph (DAG) of components (clients and scopes) such that any two clients are not directly connected, and no scope, except $\perp$, is a "subscope" of a client. In addition, we state that $\perp$ is a subscope of every component and that $\top$ is a superscope of every component. The edge directions in the scope graph indicate scope membership but notifications can "virtually" travel in both directions.

In the dimension membership of Figure 1, the scope graph is composed of clients W, X, Y and Z, and scopes es, us, fs, ir, ls, and is. For the sake of clarity, we do not draw all the edges (*component*, $\top$) and ($\perp$, *component*). In this dimension, X is a subscope of ls, ls is a subscope of us, us is a subscope of es, and X is transitively a subscope of es. But, neither us is transitively a subscope of ir nor ir is transitively a subscope of us.

Notifications may cross scope boundaries. However, before "going through" a scope boundary, a notification must match a visibility filter established between the two scopes. Like forwarding filters, visibility filters are content-based. The act of applying a visibility filter is called visibility matching. In Figure 1, the notification n is visible from scope ls to scope us, and from scope fs to scope is. The pathway of a notification in a scope graph is called a scope path. For instance, in Figure 1, in the dimension membership, n is passed along the scope path (ls, us, es, fs, is) from the producer X to the consumer Y.

The following rules determine the set of eligible direct subscopes and superscopes to which a notification is visible. Firstly, when published, a notification is made visible to the scopes the producer belongs to. This rule is applied recursively to make outgoing notifications visible to all further superscopes. Secondly, when an incoming notification is visible within a scope, it is visible to all its children. This rule is applied recursively to make notifications visible to further children. Consequently, outgoing notifications are visible to all the superscopes and to all the sibling scopes. In addition, by convention, we state the following: visibility filters to $\perp$ are always *false*; visibility filters from $\perp$ are always *true*; visibility filters to $\top$ are always *true*; and, visibility filters from $\top$ are always *false*.

A path of scopes that connects a producer X to a consumer Y is called a visibility path and is decomposed into two, possibly empty, parts: an upward part and a downward part such that X is transitively a subscope of K and K is transitively a superscope of Y. In Figure 1, the visibility path $(X, ls, us, es, fs, is, Y)$ is composed of the upward part (X, ls, us, es) and the downward part (es, fs, is, Y).

2.3 Visibility in several dimensions—Multiscoping

In order to structure the overlay network of brokers according to several dimensions, we consider more than one graph of scopes. So, when submitting an advertisement or a subscription, a client provides a set of scopes, at most one scope per dimension. This set of scopes is transformed into a set of paths of scopes (of length one) and the latter set is denoted Φ in the sequel. When publishing, a producer indicates the identity of the advertisement filter and the notification is tagged with the set of (paths of) scopes of the advertisement.

We can now define the visibility of a notification n produced by a producer X and matching the forwarding filter of the advertisement filter f and delivered to a specific consumer Y that subscribes to a subscription filter f' (denoted by $X \;_{\Phi_{X,f}}\overset{n}{\leadsto}_{\Phi_{Y,f'}}\; Y$). The usage of the pairs (X, f) and (Y, f') is for allowing several advertisements and subscriptions per client. In addition, since it is not reasonable to let designers specify a scope for every dimension of the multiscale characterisation of the distributed system, we use the specific scope $\bot$ in advertisements to indicate that $\bot$ should be used for every dimension not explicitly specified in the advertisement —i.e. the evaluation of the visibility from X to Y is replaced by the evaluation of the visibility from $\bot$ to Y. In other words, the producer is imposing no constraint on the visibility of the notification for that dimension. Similarly, we use the specific scope $\top$ in subscriptions to indicate that $\top$ should be used for every dimension not explicitly specified in the subscription —i.e. $\top$ is substituted to Y in the evaluation of the visibility from X to Y. In other words, the consumer wants to be notified regardless the scope of the notification for that dimension. Therefore, a notification n published by client X through the advertisement filter f is visible to client Y through the subscription filter f' if and only if n is visible from X to Y in all the dimensions specified in the set $\Phi_{X,f} \cup \Phi_{Y,f'} \setminus \{(\bot), (\top)\}$, with X being replaced by $\bot$ when no scope for a dimension is present in $\Phi_{X,f}$ but $\bot$ is present in $\Phi_{X,f}$, and with Y being replaced by $\top$ when no scope for a dimension is present in $\Phi_{Y,f'}$ but $\top$ is present in $\Phi_{Y,f'}$.

In Figure 1, let W and X be two producers, and Y and Z be two consumers. Given the following sets of scope paths for the advertisements and the subscriptions: $\Phi_{W,f} = \{(ls), (ch)\}$, $\Phi_{X,f} = \{(ls), (ch), (\bot)\}$, $\Phi_{Y,f'} = \{(is), (lo), (tm)\}$, and $\Phi_{Z,f'} = \{(is), (tm), (\top)\}$, we can deduce that: $W \;_{\Phi_{W,f}}\!\leadsto_{\Phi_{Y,f'}}\; Y$ does not hold because $\neg(W \leadsto Y)_{d_3}$; $W \;_{\Phi_{W,f}}\!\leadsto_{\Phi_{Z,f'}}\; Z$ does not hold because $\neg(W \leadsto Z)_{d_3}$; $X \;_{\Phi_{X,f}}\!\leadsto_{\Phi_{Y,f'}}\; Y$ does hold because $(X \leadsto Y)_{d_1} \wedge (X \leadsto Y)_{d_2} \wedge (\bot \leadsto Y)_{d_3}$; and $X \;_{\Phi_{X,f}}\!\leadsto_{\Phi_{Z,f'}}\; Z$ does hold because $(X \leadsto Z)_{d_1} \wedge (X \leadsto \top)_{d_2} \wedge (\bot \leadsto Z)_{d_3}$.

Before forwarding a notification to a given destination D —i.e. a neighbouring broker or a local client— the receiving broker B checks whether a visibility path associated with that notification can be built in the direction of D. The verification must be done not only according to the point of view of the producer for satisfying the constraints of the producer, but also according to the point of view of the consumer for satisfying the constraints of the consumer. Let Φ_n be the set of scope paths associated with the notification, and Φ_s be the set of scope paths associated with the subscription. From the point of view of the producer, a visibility path exists if there exists a visibility path for each dimension d in the set of dimensions associated with Φ_n. Similarly, from the point of view of the consumer, a visibility path exists if there exists a visibility path for each dimension d in the set of dimensions associated with Φ_s.

2.4 DEBS with multiscoping

At the access broker of the producer, the set of scope paths of the advertisement filter f_a becomes the set of scope paths of the notification —i.e. $\Phi_{f_a} = \Phi_n$. A notification matches a subscription filter f_s at the access broker of the consumer if a visibility path can be deduced from the set of scope paths of the notification Φ_n and the set of scope paths of the subscription Φ_s, and if the notification matches the forwarding filter of the subscription.

Therefore, a distributed event-based system with multiscoping satisfies the following requirements: (safety) *a)* A client receives only notifications it is currently subscribed to; *b)* A client receives only notifications that have been previously published by other clients and from which they are visible; *c)* If the filter of the publication call does not belong to the active advertisements of the publishing client or if the notification does not match the filter specified in the publication call, the notification should not be delivered to any client; *d)* A client receives a notification at most once; and (liveness) *e)* A client eventually receives every notification that is visible to it and that matches some forwarding filter of its subscriptions.

These requirements define the properties of the solution when seen as a black box. This specification does not suit when designing the routing part of the system because it does not specify the central role of brokers in the dissemination of notifications. In other words, we need a property that only depends on the configurations of neighbouring brokers for taking decisions on routing matter. In the next section, we explain distributed routing with multiscoping.

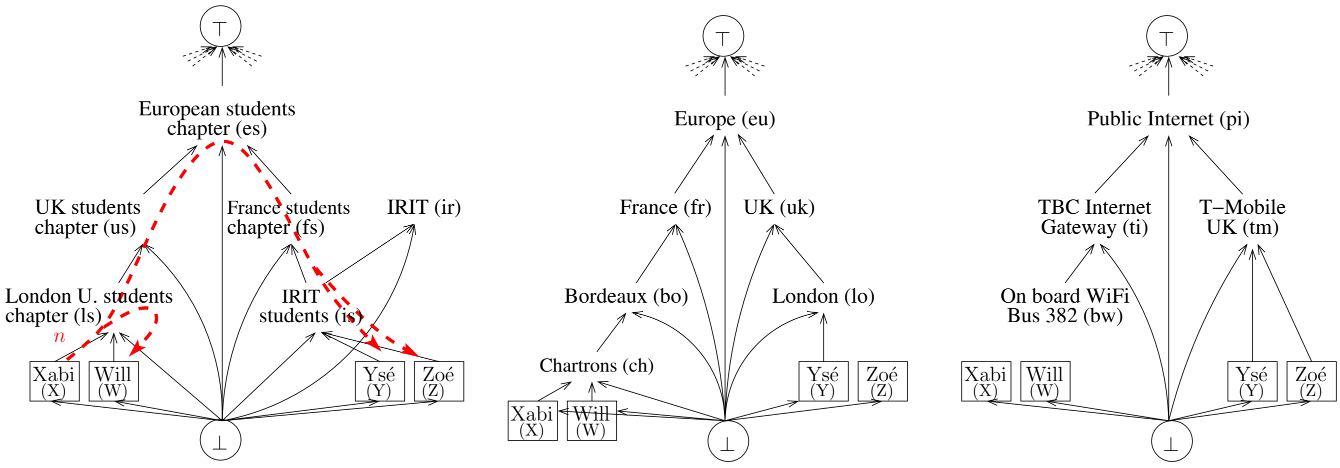
3. DISTRIBUTED ROUTING WITH MULTISCOPING

We reify a scope as an overlay of brokers and clients, with the specific scopes $\bot$ and $\top$ being virtual. Scope overlays are built such that two overlays of brokers corresponding to two scopes related by the relationship "superscope" intersect. At a broker, a notification tagged with sets of scope paths can be either forwarded to neighbouring brokers, or "mapped up or down". In the case of forwarding, n is routed within the overlay of brokers belonging to the same set of scopes and n does not go across any scope boundary. In the case of visibility mapping, Φ_n is transformed so that n becomes visible to its superscope(s) or subscope(s). Thus, n may be routed from broker to broker or transformed several times before reaching a given destination.

Then, the routing of a notification involves two kinds of treatments: forwarding between neighbouring brokers and scope transformation at a broker. The forwarding is the routing of classical DEBS without scoping.

In order to route a notification tagged with a set of scope paths, each broker classically maintains locally a routing table. The multiscoping aspect is taken into account by organising routing table entries into triples (set of scope paths Φ_s, forwarding or visibility filter f, destination D). In the case of a forwarding filter, an entry is read as follows: If D is a client, if a visibility path can be built from Φ_n, and if n matches f, then n is forwarded to D; If D is a broker, if a visibility path can be built from Φ_n and Φ_s, and if n matches f, then n is forwarded to D. In the case of a visibility filter, Φ_s contains only one scope path sp and $sp = (s)$; an entry is read as follows: if there exists a scope path sp_n in $\Phi_n \setminus \{(\bot), (\top)\}$ such that a new path sp' can be built from sp_n and sp —i.e. s is an eligible next-hop scope of $last(sp_n)$— and if n matches f; then Φ_n is transformed and the destination D is the same broker.

Therefore, a distributed routing algorithm with multiscoping satisfies the following requirements: (local validity) *a)* Only notifications that match the forwarding filter of their advertisement filter at the producer's access broker are



dimension *membership* (d_1) geographical dimension *administrative area* (d_2) dimension *network range* (d_3)

Figure 1: Dimensions, scopes, clients, and visibility (superscope relations $k \lhd \top$ are not drawn)

forwarded; *b)* Only notifications that match one of the forwarding filter of the subscription filters at the consumer's access broker are delivered and they are delivered immediately; (eventual monotone remote validity) *c)* The routing table entries with forwarding filters build delivery paths from producers to consumers; and *d)* The routing table entries with visibility filters build visibility paths from the set of scope paths of advertisement filters to the set of scope paths of subscription filters.

4. CONCLUDING REMARKS

We show that some sources of heterogeneity such as network range capabilities, geographic location, or social membership that characterise multiscale systems can be used for controlling the dissemination of subscriptions and publications. Our solution proposes to relate the scales of the multiscale characterisation of a distributed system to scopes: each relevant dimension of the multiscale characterisation leads to a graph of scopes; each scale in a dimension is related to a scope. In order to highlight the structuring of the overlay of brokers according to several dimensions, we name our approach "DEBS with multiscoping". We provide the specifications of DEBS with multiscoping and of distributed routing with multiscoping. The algorithms are implemented as an open source solution called muDEBS (`https://fusionforge.int-evry.fr/www/mudebs/`).

5. ACKNOWLEDGEMENTS

This work is part of the French National Research Agency (ANR) project INCOME (ANR-11-INFR-009, 2012-2015, `http://anr-income.fr`). The authors thank all the members of the project that contributed directly or indirectly to this work.

6. REFERENCES

[1] L. Atzori, I. Antonio, and G. Morabito. The Internet of Things: A survey. *Computer Networks*, 54(5), May 2010.

[2] G. Blair and P. Grace. Emergent Middleware: Tackling the Interoperability Problem. *IEEE Int. Comp.*, 16(1), Jan. 2012.

[3] P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec. The Many Faces of Publish/Subscribe. *ACM CS*, 35(2), June 2003.

[4] L. Fiege, M. Cilia, and B. Mühl. Publish-subscribe grows up: Support for management, visibility control, and heterogeneity. *IEEE Int. Comp.*, 10(1), Jan. 2006.

[5] M. Franklin, S. Jeffery, S. Krishnamurthy, and F. Reiss. Design Considerations for High Fan-in Systems: The HiFi Approach. In *Proc. 2nd Conf. on Innovative Data Systems Research*, Jan. 2005.

[6] A. Hinze, K. Sachs, and A. Buchmann. Event-Based Applications and Enabling Technologies. In *Proc. ACM DEBS*, July 2009.

[7] ISO/IEC/IEEE. Systems and software engineering — Architecture description. International Standard ISO/IEC/IEEE-42010:2011, ISO/IEC/IEEE Joint Technical Committee, Dec. 2011.

[8] M. Kessis, C. Roncancio, and A. Lefebvre. DASIMA: A Flexible Management Middleware in Multi-Scale Contexts. In *Proc. 6th International Conference on Information Technology: New Generations*, 2009.

[9] G. Mühl, L. Fiege, and P. Pietzuch. *Distributed Event-Based Systems*. Springer, 2006.

[10] B. Oki, M. Pfluegel, A. Siegel, and D. Skeen. The information Bus: An Architecture for Extensible Distributed Systems. In *Proc. 14th*, pages 58–68, Asheville, USA, Dec. 1993.

[11] R. Rottenberg, S. Leriche, C. Taconet, C. Lecocq, and T. Desprats. MuSCa: A Multiscale Characterization Framework for Complex Distributed Systems. In *Proc. 3rd Workshop on Model Driven Approaches in System Development*, Sept. 2014.