Distributed Event-Based System — Scoping

Master CILS — Module IAAIO, Revision : 1647

Denis Conan, avec Léon Lim

October 2017
References:

- Publications of the MARGE team
Outline

1. Objectives and Motivations
2. Definition of scope
3. Specification of DEBS with scoping (one scope graph)
4. (optional) From monoscoping to multiscoping
1 Objectives and Motivations

- Control visibility of notifications \(\implies\) Some form of control on communication
  But without impairing the benefits of loose coupling
  - Delimit areas where loose coupling and implicit coordination apply
  - Intersect areas and control distribution at the frontiers
    - Visibility of a notification limits the set of consumers

- Scoping concept for structuring applications using DEBS
  - Scope = Bundle of producers, consumers and brokers with visibility limited to the enclosed components
  - Organise components into bundles and manage communication inside a bundle differently from communication with the outside
2 Definition of scope

- Scope = an abstraction that bundles a set of producers and consumers so that the visibility of notifications published by a producer is confined to the consumers belonging to the same scope as the publisher.

- A scope can recursively be a member of other scopes.

- Notifications are scoped — i.e., tagged with scope metadata.

- Producers and consumers may take part to several scopes.

- Output/input scope interfaces for transmission policies between scopes:
  - Transformation between different data models of notifications
  - Security policies to control the access to the scope
  - Etc.
2.1 Scopes as overlays

- Here, scopes are projected onto the overlay of brokers and clients
- Other architectural choices exist
  - E.g. reification of a scope as an architectural entity
3 Specification of DEBS with scoping (one scope graph)

3.1 Scope graph and visibility of components
3.2 Delivery path, incoming / outgoing notification
3.3 Visibility matching, output and input interfaces of scopes
3.4 Visibility filters
3.5 Visibility of notifications
3.6 Formal specification of DEBS with scoping and advertisement
Scope graph and visibility of components

- $C$: the set of clients (publishers or consumers)
- $S$: the set of scopes
- $\mathcal{K} = C \cup S$
- Scope graph: Acyclic directed graph $G = (C, E)$, with $C \subseteq \mathcal{K}$ and $E \subseteq \mathcal{K} \times \mathcal{K}$, so that $(C_1, C_2) \in E \implies C_2 \in S$
  - If $C_1 \in C$ then $C_2$ is a superscope of $C_1$
  - If $C_1 \in S$ then $C_1$ is a subscope of $C_2$ and $C_2$ is a superscope of $C_1$
- $C_1 \prec C_2 \equiv (C_1, C_2) \in E$, with the transitive closure noted $\prec$
- $C_2 \succ C_1 \equiv (C_1, C_2) \in E$, with the transitive closure noted $\succ$
3.2 Delivery path, incoming / outgoing notification

Delivery path \( p = (C_1 = X, C_2, ..., C_{n-1}, C_n = Y) \) between a producer \( X \) and a consumer \( Y \) such that

- \( p \) is an undirected path in the graph of scopes
- \( \forall 1 \leq i < n, \exists Z : X \prec Z \land Z \prec Y \)

Every delivery path can be subdivided into two, possibly empty, parts:
- An upward path \( (C_1, ..., C_j) \) where \( \forall i < j : C_i \prec C_{i+1} \)
- A downward path \( (C_j, ..., C_n) \) where \( \forall i \geq j : C_{i+1} \prec C_i \)
3.3 Visibility matching, output and input interfaces of scopes

- **Visibility matching** (by analogy with subscription/advertisement matchings)
  - Before forwarding a notification from one scope to another, apply filters
    - These filters are called “visibility filters”
  - Applying these filters is called visibility matching

- **Output interface of a scope** = a set of filters such that
  - A notification that matches one of these filters is forwarded up into superscopes as an outgoing notification

- **Input interface of a scope** = a set of filters such that
  - A notification that matches one of these filters is forwarded to scope members as an incoming notification

- **Visibility matching is a test that precedes any subscription or advertisement matching**
3.4 Visibility filters

- Visibility filters transform notifications at scope boundaries
  - Binary, asymmetric relations on the set $\mathcal{N} \cup \{\epsilon\}$, with $\epsilon =$ empty notification
  - $n_1 \searrow_{(S, T)} n_2$: when “traveling” upwards along the edge $(S, T) \in E$, $n_1$ is transformed into $n_2$
  - $n_1 \nearrow_{(T, S)} n_2$: when “traveling” downwards along the edge $(S, T) \in E$, $n_1$ is transformed into $n_2$

- $n_2 \neq n_1$ means data models of $S$ and $T$ differ
- $n_2 = \epsilon$ means $n_1$ has been filtered out, for instance due to security requirements

- From now on
  - $F : \mathcal{N} \cup \{\epsilon\} \rightarrow \mathcal{N} \cup \{\epsilon\}$
  - $F(n) = n' \neq \epsilon$ means that the notification $n$ matches the filter $F$
  - $F(n) = \epsilon \notin \mathcal{N}$ means that the notification $n$ does not match the filter $F$
  - $F(\epsilon) = \epsilon$ means that $\epsilon$ matches no filter
3.5 Visibility of notifications

Visibility of a notification $n_1$ produced by $X$ to a specific consumer $Y$ as notification $n_2$

- $X \xrightarrow{n_1} n_2 Y$: $n_1$ visible to $X$ is also visible to $Y$, possibly after several mappings

$$X \xrightarrow{n_1} n_2 Y \equiv X = Y \land n_1 = n_2$$

- Possibility of several mappings:
  $$\lor \exists (X, Z) \in E, \exists n_z \neq \epsilon : n_1 \xrightarrow{(x, z)} n_z \land Z \xrightarrow{n_z} n_2 Y$$
  $$\lor \exists (Z, Y) \in E, \exists n_z \neq \epsilon : X \xrightarrow{n_1} Z \land n_z \xrightarrow{(z, y)} n_2$$

- When the resulting notification is not important, we write $X \xrightarrow{n} Y$

- When the producer is not important, we write $\xrightarrow{n_1} n_2 Y$
3.6 Formal specification of DEBS with scoping and advertisement

- **Safety:** simple DEBS + producer and consumer must be visible

\[
\square [ \text{notify} (Y, n') \implies [n' \in N(S_Y)] ] \quad (=a)
\]
\[
\land \left[ \exists X : n \in P_X \land X \sim_{n'} Y \right] \quad (=b \text{ with scoping})
\]
\[
\land [\Diamond \Box \lnot \text{notify} (Y, n')] \quad (=c)
\]

- **Liveness:** with a static scoping graph and static visibility mappings

\[
\square \left[ \square X \sim_{n'} Y \implies \Diamond \left( \square (F \in S_Y) \implies \Diamond \Box \left[ pub (X, n) \land (n' \in N(F) \implies \Diamond \text{notify} (Y, n')) \right] \right) \right]
\]
4 (optional) From monoscoping to multiscoping

4.1 Example with three scope graphs
4.2 Pseudo-scopes $\perp$ and $\top$
4.3 Distributed routing with multiscoping
4.1 Example with three scope graphs

- Notification $n$ published by $X$ is visible to $Y$ if and only if there exists a delivery path from $X$ to $Y$ in all the scope graphs.

- See next slide for the two specific scopes $\bot$ (bottom or “bot”) and $\top$ (top).
4.2 Pseudo-scopes ⊥ and ⊤

Not reasonable to let designers specify a scope for every scope graph

- ⊥ in advertisements = Producer imposing no constraint on the routing of the notification for the dimension not specified
- ⊤ in subscriptions = Consumer wanting to be notified regardless the scope of the notification for the dimension not specified
4.3 Distributed routing with multiscoping
4.3.1 Transition path from the producer to the consumer

- Transformations of scopes at brokers
  - Visibility path from a producer to a consumer

- Forwarding between brokers \(\equiv \) the routing in a regular DEBS
  - Forwarding path from a producer and a consumer
4.3.2 Mixing delivery path and visibility path
4.3.3 Notification with scoping metadata
4.3.4 Specification of distributed routing with multiscoping

- Local validity:
  a. Only notifications that match their advertisement filter at the producer’s access broker are forwarded
  b. Only notifications that match one of the subscription filters at the consumer’s access broker are delivered and they are immediately delivered

- Eventual monotone remote validity:
  c. The routing table entries with routing filters build delivery paths from producers to consumers
  d. The routing table entries with visibility filters build visibility paths from the sets of scopes of advertisement filters to the sets of scopes of subscription filters
4.3.5 Routing table entries with forwarding and visibility filters

- Triples (scope path $\Phi_s$, forwarding or visibility filter $f$, destination $D$)

- If $f =$ forwarding filter
  - If $D =$ broker and if a delivery path can be built from $\Phi_n$ and $\Phi_s$, and if $n$ matches $f$, then forward $n$ to $D$
  - If $D =$ client and if a delivery path can be built from $\Phi_n$, and if $n$ matches $f$, then forward $n$ to $D$

- If $f =$ visibility filter
  - $\Phi_s$ contains only one scope path $sp$ and $sp = (s)$
  - If there exists a scope path $sp_n$ in $\Phi_n \setminus \{(\perp), (\top)\}$ such that a new path $sp'$ can be built with $sp_n$ and $sp$ and if $n$ matches $f$, then $\Phi_n$ is transformed and the destination $D$ is the same broker

---

1. In fact, set of scope paths