UiO : Department of Informatics
University of Oslo

Implementation and Evaluation of Pub/Sub overlays for P2P Social Networks

Nils Peder Korsveien
master thesis autumn 2013

Implementation and Evaluation of Pub/Sub overlays for P2P Social Networks

Nils Peder Korsveien

14th November 2013

Abstract

Contents

I Introduction **1**

1 Introduction **3**
1.1 Thesis Outline 3
1.2 Motivation 3

2 Background **5**
2.1 The P2P Network Architecture 5
2.2 The Publish-Subscribe Communication Paradigm 5
2.2.1 Variations on Pub/Sub 6
2.3 Pub/Sub in Social Networks 7

3 Existing Approaches to Topic-based Pub/Sub **9**
3.1 Desired System Properties 9
3.2 Handling Trade-offs 9
3.2.1 Overlay construction 10
3.3 Event dissemination 12

II Implementation **15**

4 Implementation Strategy **17**

5 Implementation Details **19**

6 Protocol Descriptions **21**

7 Experimental Setup **23**
7.1 Workloads and Data Sets 23

III Evaluation **25**

8 Evaluation Metrics **27**

9 Results **29**

IV Conclusion **31**

10 Discussion **33**

11 Future Work **35**

List of Figures

2.1 The basic architecture of a Pub/Sub system. 6

List of Tables

3.1 Comparison of the different protocols and their overlay properties 12
3.2 Comparison of the different protocols and their routing properties 13

Aknowledgements

Part I

Introduction

Chapter 1

Introduction

1.1 Thesis Outline

1.2 Motivation

Chapter 2

Background

2.1 The P2P Network Architecture

2.2 The Publish-Subscribe Communication Paradigm

There are three main system components in the pub/sub interaction scheme: the publishers, the subscribers and the event service. The publishers publish events, and the subscribers subscribe for events, while the event service handles the management of subscriptions and publications, as well as routing events to the subscribers. The basic architecture of a typical pub/sub system is outlined in Figure 2.1.

The event service functions as an intermediary between publishers and subscribers. It provides a level of indirection, as well as an service interface. Publishers are able to generate new events through the `publish ()` service call. It is now the responsibility of the event service to determine which subscribers are interested in receiving this event, and how to route the event to them. The subscribers register their interest through a `subscribe ()` service call. The event service will then store each subscribers interest in order to disseminate events correctly. The publishers are then able to cancel their subscriptions through a `unsubscribe ()` service call. No information is forwarded from subscribers to publishers or from publishers to subscribers.

The pub/sub paradigm provides a higher degree of decoupling than other traditional approaches. In general there are three types of decoupling pub/sub system provides us with:

Space decoupling The publishers and subscribers do not need to know about each other.

Time decoupling Events are delivered regardless of whether or not publishers and subscribers are online at the same time.

Synchronization decoupling Neither publishers nor subscribers are blocked when attempting to perform their operations.

While many other approaches can provide the first two forms of decoupling, the advantage of pub/sub is its fully asynchronous nature.

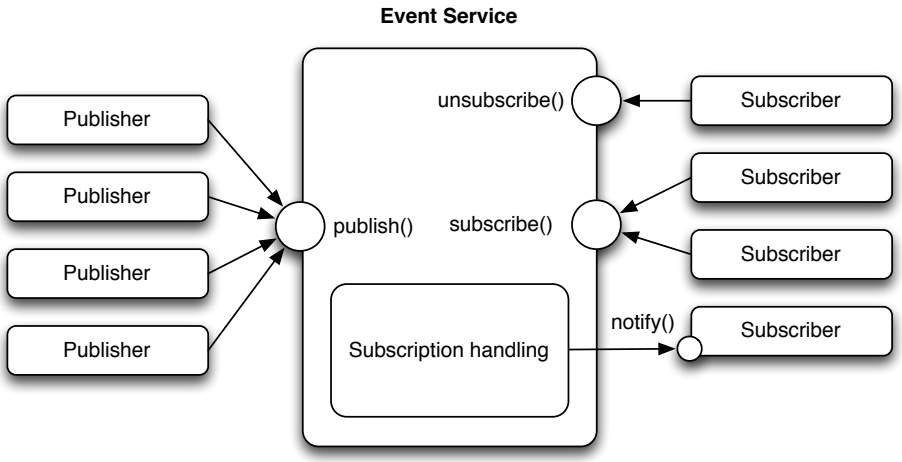

Figure 2.1: The basic architecture of a Pub/Sub system.

Approaches such as tuple spaces or message queues cannot completely provide this synchronous decoupling, as messages are retrieved in a synchronous manner. This property is key to the suitability for pub/sub in large distributed system. [**manyfaces**]

2.2.1 Variations on Pub/Sub

The subscription semantics of the pub/sub paradigm plays an important role in the performance and flexibility of the system as event messages are routed and managed based on topic or content. There are three distinct types of subscription schemes:

Topic-based Events are split into topics, usually represented by a string.

Type-based Filters events based on the structure of the data. Provides type safety at compile time.

Content-based Events are filtered based on a global list of universal event attributes.

Content-based provides better expressiveness in terms of filtering out the relevant events. However, this comes at the cost of higher overhead with regards to handling subscriptions. The complex filtering algorithms limit the scalability of such systems with regards to the number of subscriptions. Type based is similar to content-based in the sense that the public members of the types together form a description of the content of the event. Although this ties the implementation of the pub/sub system closer to the programming language, it still suffers from the same drawbacks as content-based.

Topic-based offer less expressiveness than the other two subscription schemes, but better performance if the set of possible event properties is limited. Also, topic-based is more suited for dissemination and

multicasting, as topics can be thought of as groups, where subscribing to topic T can be equivalent to joining the group for that topic. This is the approach taken by several proposed pub/sub systems which will be discussed later in this paper.

Traditionally, reliable multicasting of data through deterministic dissemination has been the common approach. However, more recent implementations investigate the potentials of probabilistic protocols, which are more suited to the nature of decentralized systems and P2P. These protocols do not guarantee full reliability, but provides a high quantifiable *probability* that events are delivered to all subscribers.

2.3 Pub/Sub in Social Networks

Chapter 3

Existing Approaches to Topic-based Pub/Sub

3.1 Desired System Properties

3.2 Handling Trade-offs

Designers of existing approaches have been facing the challenges of handling the trade-offs discussed in the previous section. One of these challenges is building both a reliable and robust overlay. A naive approach to this problem would be to create a separate overlay for each topic as in TERA [3]. However, this approach suffers from poor scalability as the number of nodes and topics increase. Another approach would be to structure the overlay by creating a spanning tree per topic as seen in Scribe [4], Bayeux [17] and Magnet [8]. However, these structures are conceptually fragile in the presence of churn, requiring mechanisms for mending the structure when nodes fail. This increases the overhead of overlay maintenance. Also, the root node of the spanning tree represents a single point of failure as well as a bottleneck in the system. This is especially true for popular topics where all events must travel through the root node. In Scribe, the root node is used as a *rendezvous* point for topics by using the routing capabilities of the underlying Pastry [12] DHTṠuch dedicated nodes represent a single point of failure in addition to being detrimental to load balancing and scalability.

Minimising the average node degree while simultaneously achieving a *topic-connected* overlay is another difficult trade-off to consider. *Topic-connectivity* is achieved when no other than the nodes who registered their interest in a topic takes part in routing events for that topic. The desired goal with regards to topic-connectivity is not only to avoid routing events through uninterested nodes, but also to minimise the node degree by reusing links for several topics. This approach achieves better scalability with regards to the number of topics in the system. Also, it decreases the message overhead incurred by both event dissemination and overlay maintenance mechanisms such as heartbeat messages. In addition to this, keeping an overlay topic-connected simplifies the message

routing mechanism as no designated relay or gateway node needs to be implemented in the protocol such as in Scribe and Vitis.

ElastO [5] propose an interesting approach to overlay construction, that aims at constructing a topic-connected overlay (hereby referred to as a TCO) while maintaining a low node degree. The construction of the overlay is performed by a centralised component which requires global knowledge, and by using any existing static or decentralised algorithm. However, the maintenance of the overlay is performed in a distributed manner in response to churn events. By using a centralised algorithm for overlay construction, ElastO can provide a more optimal TCO than decentralised solutions, while still maintaining the high performance of a decentralised repair mechanism to handle node departure or arrival.

With regards to the reuse of links for several topics, an observed correlation [9] between subscription sets in practical workloads is useful to consider when constructing overlays. This observation is exploited in Polder-cast in order to lower the number of links to maintain. Also, it is used as a basis for overlay construction in both StaN [10] and SpiderCast [6]. However, these two protocols only provide a probabilistic guarantee that the resulting overlay will be fully topic-connected. In contrast, PolderCast claim deterministic guarantees of providing a TCOḢowever, this relies on two factors: (1) that there is no churn in the system, and (2) that the underlying Cyclon [14] protocol, which is used for peer sampling in Polder-Cast, can guarantee a connected overlay. Consequently, the deterministic guarantees of PolderCast could be questioned. daMulticast [1] on the other hand provide a deterministic guarantee of topic-connectivity through quite a different approach of overlay construction. More specifically, daMulticast constructs a topic hierarchy, where events are disseminated through gossiping each level of this hierarchy in a bottom-up approach.

As mentioned in section 2, several protocols attempts to create an overlay that exhibits *small-world properties*. In Vitis, the subcluster together with the relay paths form an overlay similar to a small-world networks, which benefits the routing delay but includes uninterested gateway and relay nodes. In Magnet, small-world properties are provided by the underlying Oscar DHT [7] which also cluster similar nodes together.

3.2.1 Overlay construction

There are several different approaches to overlay construction. Structured approaches such as dissemination trees have already been mentioned, but there are also unstructured approaches. In Quasar [15], a novel approach to event dissemination using random walks removes the need for a structured overlay. There are also hybrid approaches to structuring overlays such as in ElastO [5] and Vitis [11]. In ElastO, the bootstrapping of the graph is performed by a centralised entity, while in Vitis nodes with similar interests are clustered together. A topic in Vitis might consist of several subclusters in Vitis, which are connected to each other through relay paths. This creates an overlay that is similar to dissemination trees, but where single nodes have been replaced with clusters of nodes.

However, the drawbacks are still similar to the ones found in systems relying on multicast trees, as it relies on designated gateway nodes within subclusters communicating with rendezvous nodes along the relay path. In Poldercast [13], a structured ring per topic is used in combination with a form of epidemic dissemination that resembles gossiping. Publishers are themselves part of the ring of the topic they publish, and the structures are combined into a single overlay through random links. Such an hybrid approach is an attempt at balancing the reliability of a structured overlay with the robustness of epidemic dissemination. When it comes to node degree however, Poldercast might introduce hotspots in the system as the distribution of random links might be skewed.

In Magnet, the aforementioned subscription correlation is used to build dissemination trees such as the ones seen in Scribe and Bayeux. As mentioned these structures are not ideal in dynamic systems as they require maintenance. However, tree structures do have an advantage when it comes to the case of communication overhead, as they naturally avoid any duplicate messages. This is not the case for dissemination protocols who rely on epidemic dissemination such as in daMulticast and PolderCast. However, the duplicate message delivery is also the reason why such epidemic dissemination techniques are so robust during churn. Furthermore, there is usually an adjustable fanout parameter in such systems which can be manipulated in order to control the number of messages that are forwarded by a node. Thus, there is some control over the amount of communication overhead in the system. Both PolderCast and daMulticast include such a fanout parameter. Also, it bears mentioning that structured overlays are not immune to communication overhead, as the structures usually requires both maintenance and mending in case of failure. Control messages such as heartbeats are commonplace in such protocols e.g. in Scribe where the each non-leaf node in the multicast sends heartbeats to its children periodically. This increases bandwidth consumption and adds a higher communication overhead compared to unstructured overlays where such maintenance is not required.

Node degree is another important issue to consider when designing overlays. A low average node degree increases scalability as topics and number of nodes in the system increases. In protocols who rely on an underlying DHT, node degree is usually either constant or a logarithmic function of the total number of nodes in the graph. Such is the case in Bayeux which relies on Tapestry [16], or Scribe which relies on Pastry [12]. Other implementations might have a constant node degree, which is the case in Vitis which might result in the separation of a topic into subclusters. Some extreme examples include TERA [3] and daMulticast who have a node degree that grows in the order of the number of topics the node has subscribed to. In the worst case scenario, this is also the case in PolderCast, as maintaining a low node degree depends on the degree of correlation in the subscription sets. Indeed, when using workloads from Facebook, the node degree in PolderCast grows almost linearly with subscription size, as shown in [13]. This suggests a scalability issue in a scenario where the subscription correlation is weak.

Protocol	Overlay	Structures?	TCO?	Central nodes*	sub. corr.?	Node degree
Scribe [4]	Structured	Trees	✗	RV	✗	$O(\log \lvert\mathcal{V}\rvert)$
Magnet [8]	Structured	Trees	✗	Relays	✓	$O(1)$
Bayeux [17]	Structured	Trees	✗	RV	✗	$O(\log \lvert\mathcal{V}\rvert)$
Vitis [11]	Hybrid	Trees	✗	RV&GW	✗	$O(1)$
StaN [10]	Unstructured	None	prob.	None	✓	$O(\lvert\mathcal{T}_v\rvert)$
SpiderCast [6]	Unstructured	None	prob.	WB	✓	$O(K \cdot (\lvert\mathcal{T}_v\rvert))$
daMulticast [1]	Unstructured	None	det.	None	✗	$\Theta(\lvert\mathcal{T}_v\rvert)$
Quasar [15]	Unstructured	None	✗	None	✗	Unknown
PolderCast [13]	Hybrid	Rings	det.	None	✓	$O(\lvert\mathcal{T}_v\rvert)$
ElastO [5]	Structured	Ring	det.	None	✗	$O(\rho \log \lvert\mathcal{V}\rvert\lvert\mathcal{T}\rvert)$

*RV: Rendezvous GW: Gateway WB: Weak bridge

Table 3.1: Comparison of the different protocols and their overlay properties

Table 3.1 provides an overview over several different state-of-the-art protocols, comparing their different system properties such as topic connectivity and whether or not it takes advantage of the observed subscription correlation. Note that PolderCast has received benefit of the doubt in this table, and have been marked as providing a deterministic guarantee of topic-connectivity. Magnet has a different approach to the spanning tree structures, where messages are disseminated bottom-up. This means that the root node is not a rendezvous node according to the traditional definition [2], but it is still conceptually a single point of failure as it is responsible for propagating messages back down the tree when it receives a message. In SpiderCast, there is a possibility of the overlay forming into a pattern of highly connected clusters inter-connected through a small number of links which we refer to as *weak bridges*. The node degree in SpiderCast also relies on the *K-coverage* parameter of the system, where, for each topic, a node attempts to connect to K neighbours who share the same interest. Protocols who rely on a underlying DHT typically have a node degree which grows logarithmically with the number of nodes in the system. The exception is Magnet which leverages a DHT providing small-world properties [7], creating a fixed node degree that is independent of both subscription size and number of topics [17]. Note that the node degree in Quasar is omitted from the table, as it is dependent on the implementation of the bloom filters used to represent the neighbours. In ElastO, ρ is a system parameter which balances between average and maximum node degrees when choosing new edges to recover from churn. Note that even though ElastO has a higher node degree than the centralised solutions, it provides a more optimal TCO.

3.3 Event dissemination

In terms of event dissemination, it should be mentioned that some of the systems described earlier do not concern themselves with this aspect, and focus instead of the construction and maintenance of the overlay itself. In specific, this includes SpiderCast, StaN and ElastO. Thus, any discussion regarding dissemination technique or routing performance will

Protocol	High hit-ratio during churn	100% hit-ratio in absence of churn	Message Delay	Avg. Duplication Factor
Scribe [4]	✗	✓	$O(\log \lvert\mathcal{V}\rvert)$	None
Magnet [8]	Unknown	Unknown	$O(\log \lvert\mathcal{V}\rvert)$	None
Bayeux [17]	Unknown	Unknown	$O(\log \lvert\mathcal{V}\rvert)$	None
Vitis [11]	✓	✓	$O(\log^2 \lvert\mathcal{V}\rvert/k)$	Scoped flooding
daMulticast [1]	✓	✗	$O(\log \lvert\mathcal{V}\rvert)$	Gossiping
Quasar [15]	✓	✗	Unknown	Random Walk
PolderCast [13]	✓	✓	$O(\log \lvert\mathcal{V}_i\rvert)$	$\leq fanout(f)$

*RV: Rendezvous. GW: Gateway. WB: Weak bridge.

Table 3.2: Comparison of the different protocols and their routing properties

be irrelevant for these systems. For other systems however, a comparison of techniques is in order.

As mentioned, the dissemination in systems relying on multicast trees removes any message duplication and usually offers on average dissemination of events in logarithmic time as is the case in Scribe, Magnet and Bayeux. In Vitis, event dissemination is performed by flooding inside the subcluster, while simultaneously forwarding the event to other subclusters if needed. As mentioned, gossiping is the main approach in both daMulticast and PolderCast. Gossiping usually implies an exponential dissemination speed, however, there might be other implementation specific factors in play which inhibits this property of gossiping. As an example, in PolderCast, skewed random link distribution might be detrimental to the speed of the gossiping protocol. Quasar [15] proposes quite a different approach to event dissemination, as routing is performed by having nodes install routing vectors in nearby overlay neighbours. Messages are disseminated through random walks, which are directed towards the subscribers when passing through a node with the relevant routing information. This approach is likely to be highly robust against churn. However, as observed in [15] the hit ratio stagnates at 97% in a static system. This is due to a phenomenon where some group members might be obscured by other members who absorb messages from all directions. It could be interesting to see how this phenomenon could affect the system during churn.

Table 3.2 describes the routing properties of different protocol being discussed. Note that protocols relying on a DHT usually have an expected delay which is logarithmic or squared logarithmic with the total number of nodes in the system. In Vitis, the underlying DHT provide squared logarithmic routing complexity with the total number of nodes divided by k, the number of long-range neighbours. To the best of our knowledge, there is no evaluation of the hit-ratio of Magnet or Bayeux, which is the reason of these being marked as unknown. The message delay of the system is a description of the expected path length of a event before it reaches a subscriber in number of hops. Systems relying on an underlying DHT usually provides routing performance which is logarithmic to the number of nodes in the system. Magnet differs in its approach to routing, as it relies on random walks with associated TTL values. However, as described in [8] this value is usually set to the logarithm of the total number of nodes in the system. The average duplication factor describes

the message overhead of the system, where gossiping is usually dependent on a fanout system parameter. The novel approach in Quasar means message overhead is dependent on the number of parallel random walks initiated by a node. As described earlier, protocols who create specialised dissemination structures, in these cases spanning trees, have no message duplication.

Part II

Implementation

Chapter 4

Implementation Strategy

Chapter 5

Implementation Details

Chapter 6

Protocol Descriptions

Chapter 7

Experimental Setup

7.1 Workloads and Data Sets

Part III

Evaluation

Chapter 8

Evaluation Metrics

Chapter 9

Results

Part IV

Conclusion

Chapter 10

Discussion

Chapter 11

Future Work

Bibliography

[1] S. Baehni, P.T. Eugster and R. Guerraoui. 'Data-aware multicast'. In: *DSN*. 2004.

[2] Roberto Baldoni and Antonino Virgillito. 'Distributed event routing in publish/subscribe communication systems: a survey'. In: *DIS, Universita di Roma La Sapienza, Tech. Rep* (2005), p. 5.

[3] R. Baldoni et al. 'TERA: topic-based event routing for peer-to-peer architectures'. In: *DEBS*. 2007.

[4] M. Castro et al. 'Scribe: a large-scale and decentralized application-level multicast infrastructure'. In: *IEEE Journal on Selected Areas in Communications* 20 (2002), pp. 1489–1499.

[5] Chen Chen, Roman Vitenberg and Hans-Arno Jacobsen. 'ElastO: Dynamic, Efficient, and Robust Maintenance of Low Fan-Out Overlays for Topic-based Publish/Subscribe under Churn'.

[6] G. Chockler et al. 'SpiderCast: a scalable interest-aware overlay for topic-based pub/sub communication'. In: *DEBS*. 2007.

[7] Sarunas Girdzijauskas, Anwitaman Datta and Karl Aberer. 'Oscar: Small-world overlay for realistic key distributions'. In: *Databases, Information Systems, and Peer-to-Peer Computing*. Springer, 2007, pp. 247–258.

[8] S. Girdzijauskas et al. 'Magnet: practical subscription clustering for Internet-scale Pub/Sub'. In: *DEBS*. 2010.

[9] H. Liu, V. Ramasubramanian and E. G. Sirer. 'Client behavior and feed characteristics of RSS, a publish-subscribe system for web micronews'. In: *IMC*. 2005.

[10] M. Matos et al. 'StAN: exploiting shared interests without disclosing them in gossip-based publish/subscribe'. In: *IPTPS*. 2010.

[11] F. Rahimian et al. 'Vitis: A Gossip-based Hybrid Overlay for Internet-scale Publish/Subscribe Enabling Rendezvous Routing in Unstructured Overlay Networks'. In: *IPDPS*. 2011.

[12] A.I.T. Rowstron and P. Druschel. 'Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems'. In: ed. by R. Guerraoui. Vol. 2218. LNCS. Springer, Berlin/Heidelberg, 2001, pp. 329–350.

[13] Vinay Setty et al. 'PolderCast: fast, robust, and scalable architecture for P2P topic-based pub/sub'. In: *Proceedings of the 13th International Middleware Conference*. Middleware '12. 2012.

[14] S. Voulgaris, D. Gavidia and M. van Steen. 'CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays'. In: *Journal of Network and Systems Management* 13 (2 2005), pp. 197–217. ISSN: 1064-7570. URL: http://dx.doi.org/10.1007/s10922-005-4441-x.

[15] B. Wong and S. Guha. 'Quasar: a probabilistic publish-subscribe system for social networks'. In: *IPTPS*. 2008.

[16] Ben Yanbin Zhao, John Kubiatowicz, Anthony D Joseph et al. 'Tapestry: An infrastructure for fault-tolerant wide-area location and routing'. In: (2001).

[17] S.Q. Zhuang et al. 'Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination'. In: *NOSSDAV*. 2001.