# Towards Fair Event Dissemination
## *Position Paper*

Sébastien Baehni[*], Rachid Guerraoui[*], Boris Koldehofe[†] and Maxime Monod[*]

[*]School of Computer & Communication Sciences
Ecole Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne

[†]Institute of Parallel and Distributed Systems (IPVS)
University of Stuttgart
D-70569 Stuttgart

## Abstract

*Event dissemination in large scale dynamic systems is typically claimed to be best achieved using decentralized peer-to-peer architectures. The rationale is to have every participant in the system act both as a client (information consumer) and as a server (information dissemination enabler), thus, precluding specific brokers which would prevent scalability and fault-tolerance. We argue that, for such decentralized architectures to be really meaningful, participants should serve the system as much as they benefit from it. That is, the system should be fair in the sense that the extend to which a participant acts as a server should depend on the extend to which it has the opportunity to act as a client. This is particularly crucial in selective information dissemination schemes where clients are not all interested in the same information. In this position paper, we discuss what a notion of fairness could look like, explain why current architectures are not fair, and raise several challenges towards achieving fairness.*

## 1. Introduction

The last decade has seen a drastic evolution of the scale of computing applications and systems that allow several, geographically distant participants to interact and share resources. A fundamental task underlying such applications is the reliable and effective information dissemination from some source to some "interested" parties. Typically, one would like every participant in the system to inform other participants about events of interest to them. This form of selective information dissemination is sometimes modeled through the publish/subscribe paradigm. By performing a subscribe operation, a participant (subscriber process) can express its interest and expect thereafter to be notified about all published events corresponding to its interest.

A lot of research has been devoted to providing scalable, decentralized and robust information dissemination algorithms. However, only a little part of this research has taken into account how the work a participant process performs is related to the actual *benefit* the process receives from the system. In many cases, the benefit of processes having the same interest can differ largely, and in this sense the system behaves *unfairly*. Such behavior can lead to unpredictable high churn and overhead where users repeatedly disconnect from the system because they feel treated unfairly and then try to reconnect when they expect to benefit. A *fair* system on the other hand is, intuitively, one where the processes benefit proportionally to their contribution.

In this position paper, we argue that selective event dissemination systems should consider *fairness* as a fundamental aspect of their design and implementation. We discuss what a notion of fairness could look like, explain why current architectures are not fair, and raise several challenges towards achieving fairness.

## 2. Selectivity and Interest

A selective information dissemination system consists of a dynamic and unbounded set of processes $\mathcal{P} = \{p_1, p_2, \ldots\}$ together with an interest function $\mathcal{I}(\mathcal{P}, E)$ such that for $p \in \mathcal{P}$ and some event $e \in E, \mathcal{I}(p, e)$ evaluates to true if and only if an event is interesting to $p$. The interest of processes is typically expressed using a subscription language.

The selective information system ensures that an event $e \in E$ for which $\mathcal{I}(p, e)$ evaluates to true is eventually delivered by $p$, assuming certain reliability conditions. Depending on the expressiveness of the subscription language, a process can express and change its interest either by relying on expressive filters (content-based publish/subscribe) or simple topics (topic-based publish/subscribe).

A filter allows to specify several attributes and corresponding conditions under which it evaluates to true. An event carrying attributes and corresponding values is matched to a filter if it provides all attributes specified by the filter and satisfies the corresponding conditions.

A topic can be regarded as a filter which consists of a single attribute without conditions. In a topic-based selective information dissemination system, events are only associated with a single topic and are matched against a corresponding filter.

For a process $p_i \in \mathcal{P}$ and a filter $f_j$ a selective information dissemination system provides three operations which form the execution of a selective information dissemination system:

1. *publish*$(e)$: Defines the *publication* operation. In a failure free execution (i.e., an execution where no message loss and process failures occur) the *event* is delivered to all processes $p \in \mathcal{P}$ with $e \in \mathcal{I}(p, e)$.

2. *subscribe*$(f_j,$ *callback[]*$)$: Defines the *subscription* operation. After the operation has completed, $p_i$ is guaranteed in a failure free execution to receive all events, through the provided callbacks, which match $f_j$. Once, $p_i$ has executed this operation, it is said to be *interested* in events matching its filter.

3. *unsubscribe*$(f_j)$: Defines the *unsubscribe* operation. After the operation has completed, $p_i$ is not guaranteed anymore to receive further events which match $f_j$.

The level of expressiveness for which subscriptions are performed influences how many processes need to be involved in propagating events and how many concurrent events the selective information dissemination system can handle. Basically, to promote fairness, a process which receives a large number of messages will have to participate according to the cost of disseminating all these events. Moreover, a process which places many filters will have to work for the selective information dissemination system according to the cost it takes to match these filters. With respect to a selective information dissemination system we quantify the work a process contributes by the number of forwarded messages. These might include application messages as well as infrastructure messages.

## 3. Load Balancing vs Fairness in P2P Systems

Some decentralized solutions implementing selective information systems rely on a subset of servers (sometimes even one), or brokers (e.g., [6, 9]). A low number of brokers, however, can be limiting to selective event dissemination, since in every dissemination step, at least one broker needs to be involved and the dissemination rate is coupled to the processing and communication capacity of this broker. Moreover, a system relying on few brokers is more vulnerable to failures and can reduce, if some brokers fail, the reliability of the event dissemination system.

Overcoming the limitations of broker-based solutions in terms of scalability and failure resilience has driven the development of decentralized peer-to-peer algorithms using structured or unstructured overlays (e.g., [1, 3, 8, 16]). These solutions can be regarded as server-less by having each participant acting both as client and server. Sever-less solutions intend to distribute the overall load of work evenly between all participants, i.e., the infrastructure which was previously maintained by a server or a subset of servers (brokers) is now distributed across all participants.

### 3.1. Load Balancing

Enforcing an even distribution of the work performed in the system can be characterized by *load balancing*. Load balancing is beneficial for a selective information system in providing scalability, reliability, and low latency to its participants.

Different ways of distributing the work for selective information systems have been explored. These include, for instance, hierarchical solutions like application level multicast trees embedded in structured systems. Interestingly, application level multicast trees do not even provide a balanced distribution of work between all peers, since some peers which constitute leaves in the multicast tree perform less work than other processes which need to forward messages.

SplitStream [7] addresses load balancing in the context of application level multicast trees by involving processes at different levels of an application level hierarchy. This idea has been explored further by [4] on finding efficient ways of splitting data streams to minimize the overall performed work. This method is intended for large and continuous data such as multimedia streams. In selective information dissemination systems however, it is typical that small events are used and some subscribers may rarely deliver events. Some subscribers may still contribute significantly more than others receiving in proportion to their contribution many events in the system.

### 3.2. Fairness

The idea underlying load balancing is to ensure that the total amount of work should be divided evenly across all participants. It is important to notice here that this distri-

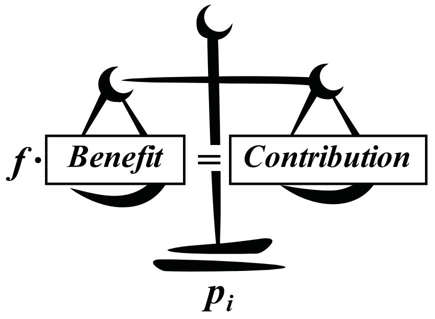

**Figure 1. The ratio contribution/benefit of each peer $p_i$ in the system must be equivalent to be considered fair.**

bution of work is irrespective of the benefits or contribution of the actual participants. Although it is somehow assumed that a decentralized peer-to-peer solution is inherently fair, the interest of processes may exhibit big differences and processes may be able to manage different loads.

The actual success of a decentralized community system seems to be tightly coupled to how fair the system is perceived by the participant processes. There are two main reasons. First, from a user point of view, perceiving an unfair treatment is usually not accepted. Second, and from an application point of view, overloading certain processes in the system can also lead to loss on the overall performance. In this context, it is important that a participant can influence locally its participation level. For example, in a selective information scheme, the expressed interest can determine the amount of work each participant needs to perform.

In a fair system the contribution of a participant (i.e., the work a participant should contribute in the system) would be a function of the benefit the participant receives from the system, itself dependent on the expressed interest of the participant (cf. Figure 1). In selective information schemes, the benefit is affected for instance, by the number of topics a participant has subscribed to, the number of events a participant has delivered or both.

Clearly, the key to implementing fairness are mechanisms that can link the benefit obtained from the system by a participant (user) to the work performed by the participant (the user's machine(s)) for the community. It is important to notice that we do not claim that for a distributed system to be scalable it should be fair; we believe however that fairness is an important notion that could drive the design of many systems where participants might exhibit selfish behavior. It might also be wise to penalize unstable nodes, as these impose additional work for maintenance on the other participants.

# 4. How Fair Are Existing Approaches?

## 4.1. Structured Approaches

A number of decentralized systems have been proposed to efficiently implement selective information dissemination. However, none of the approaches seem to fairly distribute the workload considering the benefit of processes.

For instance, Scribe [8], which follows a structured approach, has introduced the technique of rendezvous points and application level multicast trees which are constructed with the help of a lookup in a distributed hashtable (DHT). Scribe fully leverages the good properties of the underlying peer-to-peer system, namely Pastry [14], achieving very good performance with respect to network proximity metrics (stress, latency, stretch). Nevertheless, Scribe sacrifices fairness as inner nodes of a multicast participants may well have no interest at all in the given topic they are involved in, thus contributing without benefiting from the system. Disrupting this structure is not desirable as it might prevent benefiting from the network proximity provided by Pastry. In addition, a process with many subscriptions works potentially the same as a process with few subscriptions although it will subject the system to a higher load. Finally, unstable nodes are not treated differently and may impact the system as a whole.

Other approaches like DKS [1] use multiple DHTs to group processes according to their interest and have a special index DHT that allows subscribers to find a correct topic. This allows, when publishing an event, to only involve those processes with a matching subscription. Nevertheless, similar to Scribe some processes in the index DHT which are close to frequently contacted rendezvous nodes will suffer for the same reasons.

## 4.2. Unstructured Approaches

In contrast to structured selective event dissemination systems, unstructured approaches do not rely on lookup mechanism which could be used to associate a specific location with a filter. Still, it is common for unstructured approaches that each peer keeps knowledge about a number of communication partners, forming its view of the system. Usually the view reflects partial knowledge a process has of the system and by exchanging information with its neighbors, it can learn about its neighborhood and select appropriate new communication partners. For instance, in a selective event dissemination system, appropriate could mean that neighbors share similar interests. If the resulting overlay is connected, dissemination can happen simply by forwarding messages to its neighbors.

Unstructured approaches towards selective event dissemination should ensure that peers will not be logically partitioned and processes reliably receive events which are disseminated. Gossip-based dissemination systems gather a class of unstructured approaches which addresses these issues and constitute therefore an interesting candidate to study fairness properties.

Gossiping has shown to be an attractive solution towards failure resilience and reliable dissemination. The appeal of gossiping algorithms is their guaranteed connectivity and convergence in the presence of communication and process failures. For instance, a replicated database can converge to a consistent state using a gossip protocol, despite temporary partitions and process failures [10]. Along the same lines, messages can be broadcast with high reliability despite high loss rates and high process failure probabilities [5].

Most gossip protocols which address event dissemination rely, in some form or another, on a simple push-based algorithm proposed in [5]. Periodically, processes contact a fixed number of communication partners chosen at random and inform them about recently observed events. Since a uniform random selection of communication partners usually requires full knowledge of the system, a lot of literature has dealt with the problem of maintaining well distributed partial views to support random communication partner selection [2, 11, 12, 13, 15].

An important design parameter of gossip protocols is the *fanout* which denotes the number of communication partners. This parameter determines how fast and reliable an event will reach all processes of the system. Interestingly, by having all processes selecting the same fanout, all processes will be expected to perform the same amount of work in terms of forwarding messages.

Similar to structured approaches, only few unstructured approaches seem to address fairness for selective event dissemination. Data aware multicast [3] considers topic hierarchies in order to manage gossip groups. It yields fairness with respect to the dissemination since processes contribute only for messages they deliver. However, the maintenance of grouping assumes that some processes need to subscribe to a supertopic, consequently forced to be interested in all topics of the selective information dissemination system. In this case a peer in the supertopic performs similar to a broker in a client/server architecture.

For expressive event dissemination, existing gossip-based dissemination algorithms are inherently unfair. This is because they do not take into account the interest of processes in the actual propagation of messages. In a sense, the underlying assumption in classical gossip protocols is that every participant is interested in every message. In a general purpose information dissemination scheme, and as we pointed out, a process might have an interest function which determines whether a message is of interest to the process or not.

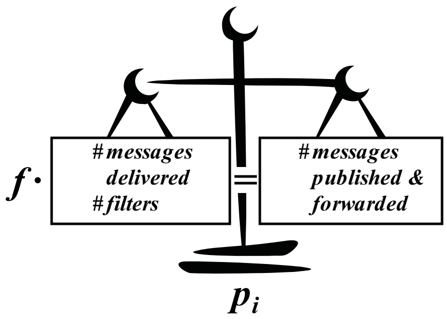

**Figure 2. The contribution is modulated with the number of events published or forwarded, whereas the benefit is computed in terms of number of interested events received, thus delivered and number of subscriptions in the system, here filters.**

## 5. Towards Fairness

Without being exhaustive, the following aspects of fairness can be considered important in a selective event dissemination system, be it structured of unstructured:

1. a process which receives many interesting events will contribute more than a process which receives few interesting events.

2. a process which places many subscriptions (i.e., filters) contributes more than a process which has only a few subscriptions.

Obviously, some of the fairness goals can be contradictory, for instance some processes may have a large number of subscriptions, but only receive few events. Therefore, we believe there must be adaptive approaches which allow to compensate between different fairness goals although some of the properties can be achieved in a structural fashion. We summarize the fairness approach in topic-based event selection in Figure 2.

### 5.1. Topic-Based Event Selection

Topic-based event selection seems to be an appealing candidate to study fairness properties. Since each topic can be associated with a process group, one could conclude that such a grouping in combination with a fair dissemination scheme automatically yields fairness.

However, a fundamental part of work in a selective information dissemination system deals with ongoing subscriptions and unsubscriptions, i.e., we need to perform continuous work in maintaining the infrastructure of the selective information dissemination system and guarantee that a subscriber can perform subscriptions from an arbitrary contact of the system. Where it seems intuitive that processes should work only for topics they have subscribed to during the dissemination, there are limitations for dealing with work performed for subscriptions and unsubscriptions. A subscriber could easily be prevented from subscribing to a topic in the selective information dissemination system because the contact node is not interested in the respective topic.

So one of the main challenges towards fairness seems to be in understanding how much a process should participate in the maintenance of the system or how much process should benefit from a system managing subscriptions and unsubscriptions.

According to the fairness aspects we highlighted, one requirement is to contribute according to the number of subscriptions. Each subscription requires from the system maintenance cost and gives a guarantee that whenever some matching event occurs we should observe it. This suggests that a process which has subscribed to a large number of topics sends more messages than a process which has only a few subscriptions. On the other hand we need to take into account the benefit given by the number of delivered events.

Ideally, a fair solution would adapt the contribution to the current number of events. If almost no interesting events happen in the system, a fair system would consider the cost in terms of subscriptions, whilst if we observe a lot of events in the system the processes which benefit a lot will do most of the maintenance work.

Another challenge for fair subscription management is that not every topic has the same popularity and even the rate at which processes subscribe and unsubscribe can be different for two distinct topics, even if both topics have the same population size. In this case some unlucky processes may be far more often involved in forwarding subscription requests than others.

These challenges highlight the fact that providing fairness for event dissemination systems is not an easy task and require a careful analysis of the expected fairness goals.

### 5.2. Expressive Event Selection

With expressive event selection, it is hardly possible to group processes according to their interest. For this reason it may not be possible to establish fairness by each process forwarding only those events which match its local filter. By doing so we are likely to invalidate the requirement of the selective event dissemination that processes will be informed about all events matching their interest. Instead, we may require that all processes adapt the participation in the event dissemination according to the benefit, namely by the number of interesting events.

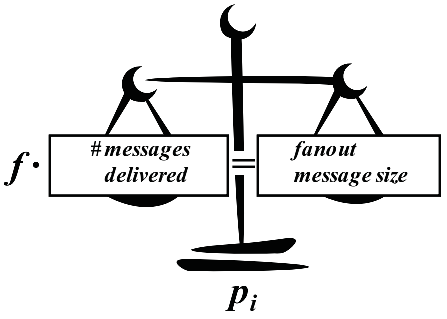

**Figure 3. The contribution is modulated with the fanout and the gossip message size, whereas the benefit is computed in terms of number of interested events received, thus delivered.**

In the context of gossip-based event dissemination, we could adapt the participation level of a process by adapting the fanout. Changing the fanout precisely means changing the contribution of the process: the size of the fanout directly impacts the number of gossip messages that need to be sent by the process in every computation/communication round.

We could alternatively adapt the number of events contained in a gossip message that is forwarded to each neighbor. By selecting more or less messages to forward, the contribution of the sender can also be modulated, we illustrate this concept in Figure 3.

Consider now the basic push gossip-based dissemination algorithm, as illustrated in Figure 4. The algorithm provides an extension to simple gossiping protocols by ensuring that an event is only delivered if it matches the interest of the process. This is achieved by having a function isInterested(e) which can decide whether the event is of interest to a process or not.

Clearly, for a static fanout $F$ and a static size of gossip message $N$ the protocol could only be fair if all processes were interested exactly in the same number of events (i.e., if all processes have a very similar interest function). However, if processes have a measure of their benefit, a process would be able to choose its fanout accordingly and ensure fair dissemination of events. A measure for benefit would be the number of delivered events within a predefined time period. Hence, a fair gossip protocol will need to adapt continuously to the observed number of interesting events. In order to fully devise such a protocol several issues need to be addressed:

```
Initialization:
    delivered ← $\{\}$
    events ← $\{\}$
upon TIMER($t$ time units) at process $p_i$ do
    Neighbors ← SELECTPARTICIPANTS($F$)
    gossip-msg.events ← SELECTEVENTS($N$ in events)
    for all $p \in$ Neighbors do
        P2PSEND($p$, gossip-msg)
    end for
end upon
upon RECEIVE(gossip-msg) do
    for all $e \in$ gossip-msg.events do
        if $e \notin$ delivered then
            events ← events $\cup \{e\}$
            if ISINTERESTED($e$) then
                DELIVER($e$)
                delivered ← delivered $\cup \{e\}$
            end if
        end if
    end for
end upon
```

**Figure 4. A basic push gossip-dissemination algorithm**

- How can the fanout be dynamically adapted to ensure quick convergence to an appropriate fanout?
- How can the gossip message size be dynamically adapted to ensure quick convergence to an appropriate message size?
- Is there any requirement on the size of the fanout?
- Is there any requirement on the gossip message size?
- How can an adaptive algorithm maintain robustness of gossip protocols?
- Can we ensure that a peer does not artificially grow its contribution by biasing the selection of peers (i.e., biasing the fanout) or the selection of events (i.e., biasing the gossip message size)?

In general, there may be alternative ways to control the participation of a process in an expressive selective information processing, which are suited to a specific event dissemination system. In some cases we may also rely on semantic knowledge to bias the participation knowledge and provide grouping according to this semantic knowledge. Similar to the discussion of topic-based event selection we need also to ensure that the work for maintenance of the dissemination system is fairly shared among the participants.

## 6. Summary

The last decade has seen an evolution of computing systems to allow a large number of computing units to interact and share resources. A characteristic of such systems is that the set of units, also called participants, is dynamically changing. Participants disconnect without any notice and typically exhibit a selfish behavior.

In order to support the scalability of such systems and provide failure resilience, there is a trend to move from traditional client-server patterns to decentralized resource management schemes. Such schemes, as considered for instance in the context of grid and peer-to-peer computing, relies on each participant that benefits from the system to provide in turn some services to other participants. While current solutions have succeeded in managing resources in a dynamic setting, they still suffer from an uneven distribution of the workload in the system. In short, some participants with low demand have to perform as much work as other participants with high demand for resources. This is especially important when offering resources is considered to be expensive as for instance in the context of mobile computing because communication adds significant costs in terms of energy consumed.

Load-balancing techniques are usually considered but these do not capture any notion of fairness. Since participants typically act in a selfish manner, an unfair distribution of workload can lead to a high churn in a dynamic system where processes abruptly disconnect whenever they perceive to perform too much work. Such behavior can significantly impact the reliability and scalability of a decentralized system.

Not surprisingly, the success of a scalable resource sharing system is closely coupled to how fair the system is perceived in sharing the workload for managing requested services. Roughly speaking, a fair distributed system will have to provide mechanisms that can link the benefits obtained from the system by a participant (user) to the work performed by the participant (the user's machine(s)) for the community. It is important to notice that we do not claim that for a distributed system to be scalable it should be fair; we believe however that fairness is an important notion that could drive the design of many systems where participants exhibit selfish behavior. In short, a fair system would allocate work to participants as a function of how much they actually benefit from the system. The idea is appealing and rather intuitive. Understanding its ramifications is more challenging however.

A clear challenge in the design of future scalable distributed system is to precisely understand the very notion of fairness in a decentralized environment and to design distributed algorithms that capture to this notion. The context of selective information dissemination is particularly attractive in this context.

## References

[1] L. O. Alima, A. Ghodsi, P. Brand, and S. Haridi. Multicast in DKS(N; k; f) overlay networks. In *Proceedings of the 7th International Conference on Principles of Distributed Systems (OPODIS '03)*, pages 83–95. Springer-Verlag, 2003.

[2] A. Allavena, A. Demers, and J. E. Hopcroft. Correctness of a gossip based membership protocol. In *Proceedings of the 24th ACM symposium on Principles of distributed computing (PODC '05)*, pages 292–301. ACM Press, 2005.

[3] S. Baehni, P. T. Eugster, and R. Guerraoui. Data-aware multicast. In *Proceedings of the 5th IEEE International Conference on Dependable Systems and Networks (DSN '04)*, pages 233–242. IEEE Computer Society, 2004.

[4] D. Bickson, D. Malkhi, and D. Rabinowitz. Efficient large scale content distribution. In *Proceedings of the 6th Workshop on Distributed Data and Structures (WDAS '04)*, 2004.

[5] K. P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems*, 17(2):41–88, May 1999.

[6] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19(3):332–383, 2001.

[7] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: high-bandwidth multicast in cooperative environments. In *Proceedings of the 19th ACM symposium on Operating systems principles (SOSP '03)*, pages 298–313. ACM Press, 2003.

[8] M. Castro, P. Druschel, A.-M. Kermarrec, and A. I. T. Rowstron. SCRIBE: A large-scale and decentralised application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communications*, 20(8):100–110, 2002.

[9] G. Cugola, E. D. Nitto, and A. Fuggetta. The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. *IEEE Transactions on Software Engineering*, 27(9):827–850, 2001.

[10] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic algorithms for replicated database maintenance. In *Proceedings of the 6th Annual ACM Symposium on Principles of Distributed Computing (PODC '87)*, pages 1–12. ACM Press, 1987.

[11] P. T. Eugster, R. Guerraoui, S. B. Handurukande, A.-M. Kermarrec, and P. Kouznetsov. Lightweight probabilistic broadcast. *ACM Transactions on Computer Systems*, 21(4):341 – 374, 2003.

[12] M. Jelasity, R. Guerraoui, A.-M. Kermarrec, and M. van Steen. The peer sampling service: Experimental evaluation of unstructured gossip-based implementations. In *Proceedings of the 5th ACM/IFIP/USENIX International Conference on Middleware (Middleware '04)*, pages 79–98. Springer-Verlag, 2004.

[13] A.-M. Kermarrec, L. Massoulié, and A. J. Ganesh. Probabilistic reliable dissemination in large-scale systems. *IEEE Transactions on Parallel and Distributed Systems*, 14(3):248–258, Mar. 2003.

[14] A. Rowstron and P. Druschel. Pastry: scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware '01)*, pages 329–350. Springer-Verlag, 2001.

[15] S. Voulgaris, D. Gavidia1, and M. van Steen. Cyclon: Inexpensive membership management for unstructured p2p overlays. *Journal of Network and Systems Management*, 13(2):197–217, June 2005.

[16] S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. H. Katz, and J. D. Kubiatowicz. Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination. In *Proceedings of the 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '01)*, pages 11–20. ACM Press, 2001.