Quality-Constrained Routing in Publish/Subscribe Systems

Bogumil Zieba
Thales Nederland B.V,
Innovation Combat Systems,
Haaksbergerstraat 49
Hengelo (O), 7550 GD,
The Netherlands
+31742482661
bogumil.zieba@nl.thalesgroup.com

Marten van Sinderen
University of Twente,
Department of Computer Science,
Centre for Telematics and Information Technology
PO Box 217, 7500 AE, Enschede,
The Netherlands
m.j.vansinderen@utwente.nl

Maarten Wegdam
University of Twente,
Department of Computer Science, Centre for Telematics and Information Technology
PO Box 217, 7500 AE,
Enschede, The Netherlands
m.wegdam@utwente.nl

ABSTRACT
Routing in publish/subscribe (pub/sub) features a communication model where messages are not given explicit destination addresses, but destinations are determined by matching the subscription declared by subscribers. For a dynamic computing environment with applications that have quality demands, this is not sufficient. Routing decision should, in such environments, not only depend on the subscription predicate, but should also take the quality-constraints of applications and characteristics of network paths into account. We identified three abstraction levels of these quality constraints: functional, middleware and network. The main contribution of the paper is the concept of the integration of these constraints into the pub/sub routing. This is done by extending the syntax of pub/sub system and applying four generic, proposed by us, guidelines. The added values of quality-constrained routing concept are: message delivery satisfying quality demands of applications, improvement of system scalability and more optimise use of the network resources. We discuss the use case that shows the practical value of our concept.

Categories and Subject Descriptors
C.2.4 [**COMPUTER-COMMUNICATION NETWORKS**]: Distributed Systems – *Distributed applications*

General Terms
Performance, Design, Reliability

Keywords
Publish/subscribe; quality-constraints; routing; quality-of-service; content-based;

1. INTRODUCTION
Publish/subscribe (pub/sub) systems have recently gained significant attention because their computational model fits well when dealing with real-time, distributed data-centric applications, which exchange [24, 25]. The systems feature a data-centric communication pattern, where applications publish (supply or stream) large amount of "data" samples, which are then available to remote applications that are interested in them. It uses an interaction model that consists of information publishers, which publish events to the system and information subscribers, which subscribe to events of interest within the system. An event can be seen as a special message sent by an information publisher and, implicitly addressed to the set of information subscribers, which issued subscription that matches the event [13]. A participant may simultaneously publish and subscribe to the different events.

The motivation for the research was derived from several uses-cases (one of them is demonstrated in the subsection 3.2). We noticed that pub/sub system, implemented according to DDS specification[1], does not sufficiently support distributed applications with quality demands. Considered use-cases operate in networks, which exhibit high level of dynamism in terms of varying network topology and characteristics of network links. Pub/sub system shall respond to the changes in the computing environment in order to provide a robust supportive infrastructure for applications with quality demands.

We observed redundant messages delivery in the particular use-cases of data-centric systems. This is undesired feature in every distributed system. The pub/sub system shall constrain the communication between distributed applications in order to prevent against the redundant messages delivery.

In this paper, we propose extensions to pub/sub system syntax and four general guidelines for the routing in pub/sub system. They represent the idea that pub/sub routing decision shall be dependent on a subscription declared by subscriber and additionally, on quality-constraints of network, middleware and

[1] DDS (Data Distribution Service for Real-Time Systems Specification) is a modern OMG (Object Management Group) specification for interoperable pub/sub middleware. The purpose of this specification is to offer standardized interfaces and behavior of pub/sub system [15].

application abstraction layers. These guidelines are illustrated by use case from the functional demonstrator.

The rest of the paper is structured as follows. The section 2 introduces the syntax of pub/sub and its network topology. Section 3 presents the motivation for research. Section 4 presents our approach for a quality-constrained routing. Section 5 discusses the guidelines for the quality-constrained routing. In section 6 we describe example that shows the practical value of our concept. Section 7 discusses related work. Finally, conclusions and future work are presented in the section 8.

2. PUBLISH/SUBSCRIBES SYSTEMS

A key aspect of any pub/sub system is how the destination path of the event is evaluated. In the content-based pub/sub systems (with a difference to channel and subject-based systems) a subscriber subscribes to data according to its content. Pub/sub system evaluates the destination path of events, based on a determination by matching the content of the events against the subscription defined by the subscriber. For example, it is possible to subscribe only to those quotes of a certain stock whose price is above a certain limit [21].

The essential feature of the pub/sub systems is the decoupling between publisher and subscriber in terms of:

1. Space decoupling (that captures the fact that interacting parties do not need to known each other)
2. Time decoupling (that captures the fact that parties do not need to be actively participating in the interaction at the same time)
3. Flow decoupling (that captures the asynchrony of the mode) [16].

2.1 The Syntax of the Publish/Subscribe System

Conceptually, a system can be viewed, from the functional point of view, as a black box with an interface [21]. The interface offers a number of operations: *sub(X,e)* - participant *X* subscribes to event *e*, *unsub(X,e)* - participant *X* unsubscribes to event *e*, *pub(X,e)* - participant *X* publishes event *e*, *notify(X,e)* - participant *X* is notified of event *e* (see figure 1 for details).

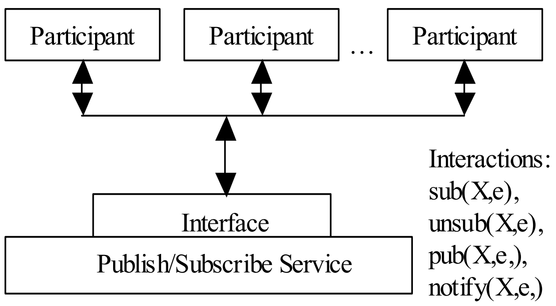


Fig. 1. Black box view of a publish/subscribe system [20].

2.2 The Topology of Publish/Subscribe System

A pub/sub network creates a virtual topology on top of the physical topology. In this sense pub/sub networks are a type of overlay network, consisting of subscribers, publishers and routers connected by communication links. It is given by a graph, which is assumed to be acyclic and connected (see figure 2 for details). Publishers/subscribers have exactly one link and act as senders or receivers of events. Routers are participants with more than one link and act as dispatchers for events that transit through them [7]. Each router manages an exclusive subset of the pub/sub participants. All events transmitted to/from intra-networks go via the routers.

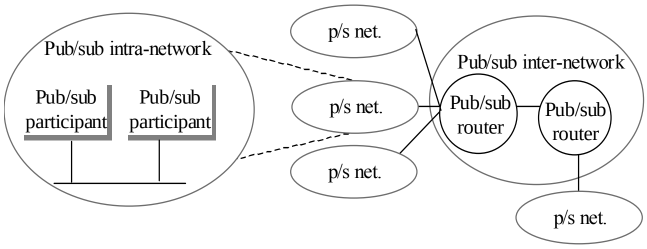


Fig. 2. Pub/sub system topology

Following traditional networking terminology, we say that a router communicates with each neighbour participant through an interface. The routers function consists of two interrelated sub-functions:

1. Routing - evaluates events paths through the inter-network by compiling and positioning local forwarding tables at each participant
2. Forwarding - function processes an incoming event, consults the forwarding table and determines the set of interfaces on which to output the event.

Taken together, the forwarding performed at the router causes events to be routed through the network. Each participant advertises a subscription that denotes events of interest for that participant and, thus, the events that the participant intends to receive [28]. For simplicity, we assume, that routers have the 'global knowledge' of all subscriptions, publishers and inter-network topology, due to propagating the routing table among each other. We are aware of a little suitability of this simple routing algorithm to the large-scale systems and other possible solutions e.g., covering-based routing [5]. However, the subject of research abstracts from a routing algorithm and focuses on quality-constraints.

3. MOTIVATION FOR RESEARCH

Content-based pub/sub systems are a subject of many research efforts [17, 8, 5, 6], also in application for large-scale distributed systems [21]. Those systems shall be highly scalable, which implies that addition or removal of services involve minimal expense, performance degradation and administrative complexity [22]. The need for supporting applications with functional and non-functional qualities - QoS (Quality-of-Service) demands in a dynamic computing environment imposes new requirements on the routing in pub/sub systems. QoS support has been extensively studied over the past decade in network layer [3, 9] and end-host systems [19, 23, 4, 2]. For example, in network layer routing protocols use metrics to evaluate what path is the best for a packet to travel. A metric is a standard of measurement, such as path bandwidth, that is used by routing algorithms to determine the optimal path to a destination [10]. However, the integration of qualities support in pub/sub systems has not been sufficiently addressed (see related work).

3.1 Motivation for the Quality-Constrained Routing

The key point of our paper is the statement that the events routing path determination, based on subscriptions, is not sufficient for and dynamic pub/sub systems with applications that have quality demands. The routing decision shall depend on additional criteria due to the following reasons:

- Quality demands of applications. Applications in some domains are particularly quality demanding of the supporting infrastructure e.g., pub/sub service. They require a certain qualities, which provision is the condition of correct application behaviour. For example, information delivered too late to subscribers is erroneous information. The quality demands of applications can concern functional qualities e.g., accuracy of measurements, or non-functional – QoS e.g., delay, or frame rate.
- Composition of the events flow. In a case, when more than one publisher is a source of a same type of event, pub/sub system shall perform a publisher selection due to undesired redundancy of events delivery. The source selection is considered as the composition of the events flow path. Redundant event delivery may saturate the network or, for example, may well cause processing delays. The selection could be performed based on the following criteria: offered QoS by publisher (for example delay of event production), or functional quality of information encapsulated in an event.
- Dynamic environment. This paper addresses dynamic computing environments. Applications can arbitrarily leave and join the system. The network topology and communication bandwidth can vary at runtime. Pub/sub routing shall react to changes in the environment and modify the routing paths of the events.

3.2 Tracking Scenario as the Problem Illustration

Let us assume the following scenario as an illustration of the motivation for the research. Pub/sub systems are suitable for information-driven tasks (data-centric tasks), for example: multisensor data fusion. The task is to generate tracks on the basis of observation made by more than one sensor. These sensor observations are called plots [20]. The scenario consists of three vessels connected by inter-network by links: L1, L2, L3 (see figure 3 for details). Intra-network connects services e.g., RS (Radar Service), TFS (Tracking Fusion Service) located within one vessel. The TFS on vessel 3 subscribes to event – '*track*' covering the sector '3E' using the following SQL-based subscription expression: "*select * from tracks where sector='3E'*". The RS at vessel 3 does not cover the sector 3E. RS at vessels number 1 and 2 produce needed tracks covering the sector 3E. However, tracks coming only from one vessel shall be delivered, due to undesirable redundancy of information delivery and limited bandwidth of link L3. The pub/sub router at vessel 2 shall select which events to forward to vessel 3 between events from vessel 2 or 3. This selection may be done based on the following metrics:

1. Link bandwidth L2, L1. Note that L2 bandwidth is greater then L1.
2. Delay of event delivery between vessels '1-3' and '2-3'.
3. Quality of tracking. Deliver 'tracks' from the more precise radar.

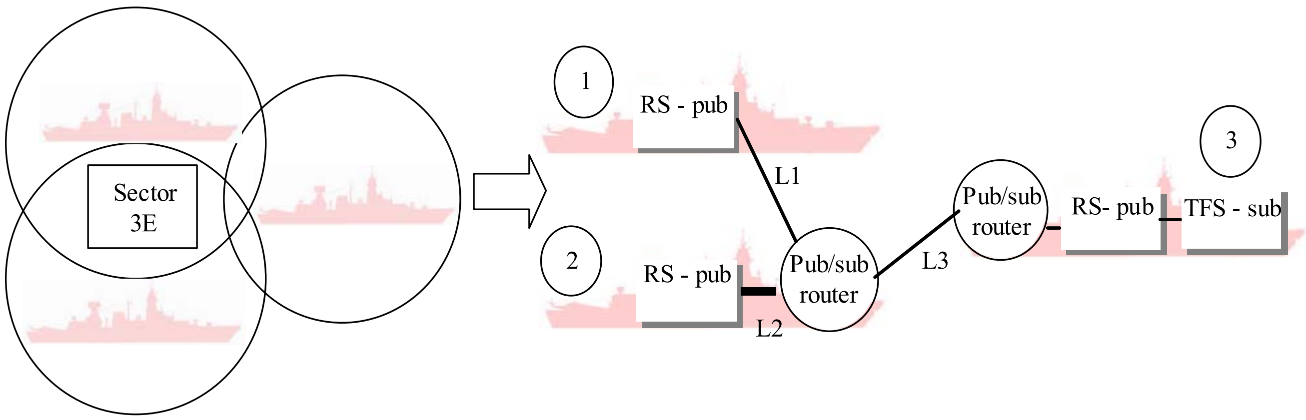


Fig. 3. Scenario of tracking among three vessels

4. APPROACH

In the previous section we pointed out that the pub/sub routing shall determine routing paths of events based on additional, multiple constraints. Constrained-based routing (CBR) denotes a class of routing algorithms that base path selection decisions is based on set of QoS requirements in addition to the destination [27]. Constraints imposed by QoS requirements are referred to as QoS constraints [12]. In this section we discuss qualities types, which shall be incorporated into the CBR in order to match the needs, specified in the motivation of the research.

The important contribution of the work is the identification of relevant quality, on different abstraction levels, which impose constraints on the routing. We identify three quality levels:

- Network QoS (non-functional)
- Middleware QoS (non-functional)
- Functional quality of data, contained in events

4.1 Network QoS

Network QoS are expressed by QoS capabilities (the actual abilities provided by the system configuration [26]) and observations (observed values using measurement functions [27]) of network paths connecting the pub/sub participants and routers. The examples of considered network QoS are:

1. Available link bandwidth - denotes that some percentage of bandwidth is available for data flows.
2. Link propagation delay- denotes the latency encountered on the network link.

4.2 Middleware QoS

Middleware QoS - defines a QoS as a set of characteristics that controls some aspect of the behaviour of the pub/sub service. It is comprised of QoS offers and requirements categories. These policies follow the subscriber-requested, publisher-offered pattern. In this pattern, the subscriber side can specify a "requested" value for a particular QoS policy. The publisher side specifies an "offered" value for that QoS policy. The pub/sub service will then determine whether the value requested by the subscriber side is compatible with what is offered by the publisher side. Communication between publisher and subscriber is established only if the two QoS policies are compatible [15]. The example of middleware QoS is 'Deadline', defined in the DDS specification. The subscriber expects a new sample of data at least once every value of '*deadline_period'* to be produced by the publisher.

4.3 Functional quality

We made the assumption that events contain information. Functional quality refers to a relative value of information quality embedded in an event. Information quality tracks the completeness, correctness, currency, consistency and precision of the data items and information statements. The pub/sub service analyses a content of each event and evaluates information in it according to some predefined criteria. This analysis is domain dependent and requires a-priori knowledge of evaluation criteria. For example, events containing the tracks may additionally include information about accuracy of the radar. Thus, the pub/sub service can relatively compare the events in terms of their functional quality.

In this paper, we avoid giving any implementation details of the presented concept. However, in order to make the concept of functional qualities more clear we show one of the possible implementation techniques. We express criteria of events evaluation and functional qualities using the general quality specification language - QML[2]. Subscriber delivers criteria of evaluation together with subscription. Publisher attaches qualities of information to each publication. Using QML we have built the functional demonstrator. We show the use-case from the demonstrator in section 6.

4.4 Extension of pub/sub syntax

The previously presented syntax of the pub/sub system did not take into account the QoS specifications and functional qualities. The notation is complemented further in this section. We extend operations of pub/sub by adding middleware QoS - $q_{middleware}$ and functional qualities - $q_{functional}$ (see table 1). It follows that a publisher *X* publishes an event *e* offering a certain middleware QoS - $q_{middleware}$ and functional qualities - $q_{functional}$. A subscriber *X* declares not only subscription for event *e*, but also additionally a certain qualities: middleware - QoS and functional qualities. If event's matches a subscription predicate and required qualities matches offered, a subscriber is notified of event *e*.

Table 1. The comparison between standard syntax of pub/sub systems and extended

Pub/sub Operations	Extended Pub/sub Operations
sub(X,e)	sub(X,e, $q_{middleware}$,$q_{functional}$)
unsub(X,e)	unsub(X,e, $q_{middleware}$,$q_{functional}$)
pub(X,e)	pub(X,e, $q_{middleware}$, $q_{functional}$)
notify(X,e)	notify(X,e, $q_{middleware}$, $q_{functional}$)

We notice that the QoS and functional qualities extensions do not bind participants by means of any QoS contract/agreement preserving the decoupling between a publisher and a subscriber.

5. THE GUIDELINES FOR QUALITY-CONSTRAINED ROUTING

In this section we present the guidelines, in the form of four generic rules, which represent the concept of quality-constrained routing. We assumed that the following input data are available to the pub/sub service:

- Subscription, which consists of: a content-based subscription, required middleware QoS and functional qualities.
- Event publication, which consists of: information, offered middleware QoS and functional qualities.
- Network topology with links capabilities and observations.

---

[2] Frølund S., Koistinen J. - Software Technology Laboratory; HPL-98-159; September, 1998; Quality-of-Service Specification in Distributed Object Systems

Application of the four generic rules (presented further in this section), to those input data results in the event path determination satisfying quality-constraints (see figure 4).

5.1 Rule (1): Satisfaction of subscription predicate

The pub/sub service guarantees the delivery of a notification to all interested parties that have subscribed to it. This means that the pub/sub service delivers a notification *e* to an object *X* if only *X* has subscribed to the event *e*. The routers propagate all subscriptions within the scope of the whole inter-network. Using the previously defined operations, we define this guarantee of the delivery as the rule (1).

Table 2. Rule(1): Guarantee of the delivery

IF $sub(X,e,q_{middleware},q_{functional})$
THEN $notify(X,e,\ q_{middleware},\ q_{functional})$.

Rule (1) specifies the basic functionality of the pub/sub system. It does not take into account any additional quality constraints. This rule implies all viable routes connecting all publishers of event *e* *and* subscribers to the *e* in the system.

5.2 Rule (2): Satisfaction of middleware QoS predicate

Whether a notification *e* is delivered to an object *X* that has subscribed to the event *e* depends on the satisfaction of the "subscriber-requested, publisher-offered" pattern. This pattern refers to the middleware QoS.

We introduce the function that evaluates whether this pattern is fulfilled: *satisfaction(sub,pub)*. Its arguments are a subscriber and a publisher of the event *e*. If the pattern is fulfilled the function returns boolean value '*true*', otherwise '*false*'. The pub/sub service delivers the notification *e* to an object *X* that has subscribed to the event *e* satisfying the pattern.

Table 3. Rule (2): Satisfaction of "subscriber-requested, publisher-offered" pattern

IF($satisfaction(\ sub(X,e,q_{middleware},q_{functional}),$
$pub(Y,e,q_{middleware},\ q_{functional})))$
THEN $notify(X,e,\ q_{middleware},\ q_{functional})$

The application of the rule (2) restricts all viable routes connecting all publishers and subscribers of event *e* to only those pairs, which satisfy "subscriber-requested, publisher-offered" pattern.

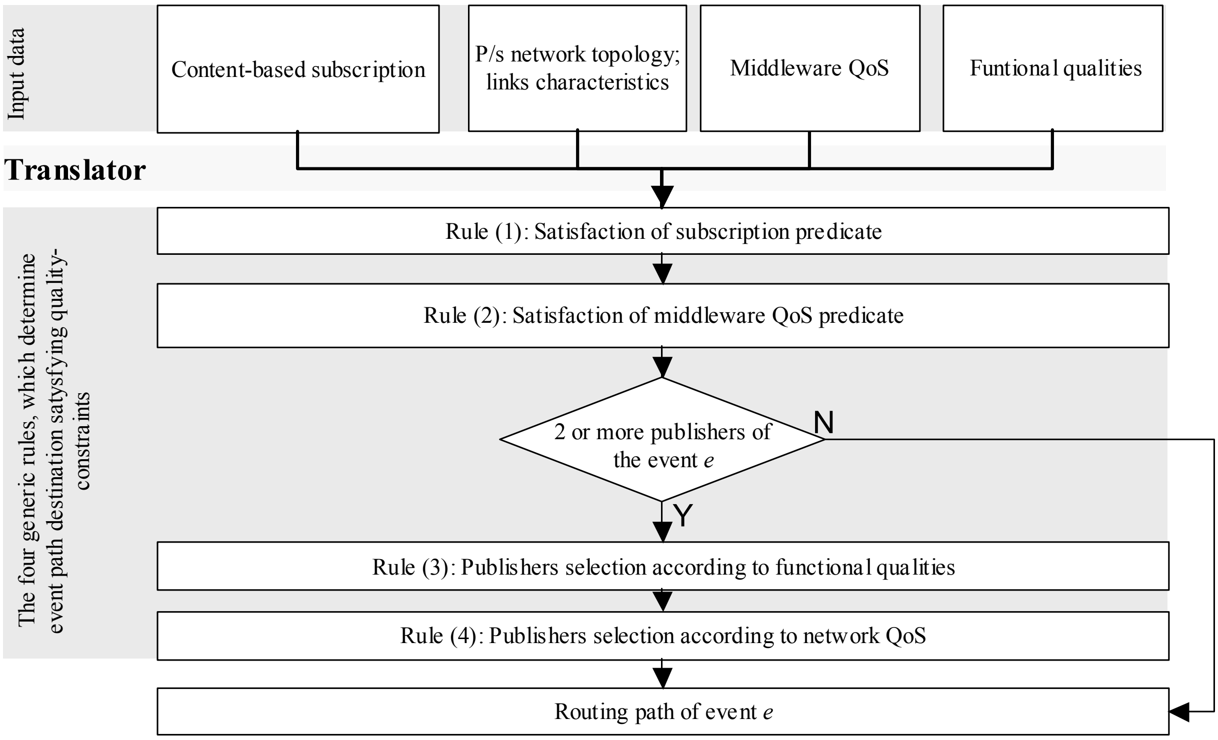


Fig. 4. The Guidelines for Quality-Constrained Routing

5.3 Rule (3): Publishers selection according to functional qualities

This rule is 'fired' only in the situation, where there are more than one publisher of the event *e*. Firstly, pub/sub routers select a publisher and then forward events from the selected publisher, blocking events from the other, unselected publishers. We introduce the function *quality(*pub_1*,*pub_2*),* which compares the functional quality of data produced by two publishers. Its arguments are two different publishers of the event *e* (we denote that pub_1 produces event e_1 and pub_2 produces event e_2; however e_1=e_2). The function returns boolean value '*true*', where publisher- pub_1 provides a better data quality than the publisher - pub_2; otherwise it returns '*false*'.

Table 4. Rule (3): Deliver the best functional quality events

$IF(quality(pub_1(Y_1,e_1,q_{middleware},q_{1functional}),$ $pub_2(Y_2,e_2,q_{middleware},q_{2functional})))$
$THEN\ notify(X,e_1,\ q_{middleware},\ q_{1functional})$
$ELSE\ \ notify(X,e_2,\ q_{middleware},\ q_{2functional\ l})$

For example, two publishers produce tracks with different radar accuracies. Routers forward tracks with a better accuracy.

5.4 Rule (4): Publishers selection according to network QoS

The 'firing' condition of the rule (4) is the same as in the case of the rule (3): more than one publisher of an event. The pub/sub system forwards selected events according to the network QoS. It requires that routers propagate the information about topology of network together with links characteristics. We introduce the function *cost(pub,sub)* that evaluates the cost of the event delivery in terms of the network QoS. Its arguments are the publisher and the subscriber of the event *e*. The rule (4) compares the cost of the event delivery between two pairs of publisher-subscriber. The pair with a lower cost is selected.

Table 5. Rule(4): Deliver the best data quality

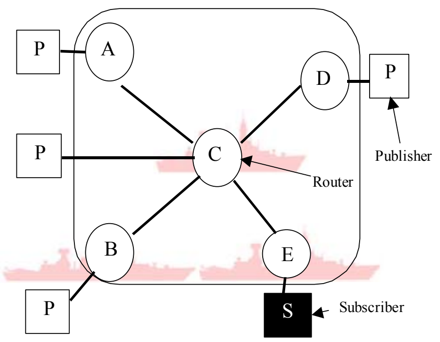
$IF(cost(pub_1(Y_1,e_1,q_{middleware},q_{func}),sub(X,e,q_{middleware},q_{functional}))>$ $cost(pub_2(Y_2,e_2,q_{middleware},q_{func}),sub(X,e,q_{middleware},q_{functional})))$
$THEN\ notify(X,e_2,q_{middleware},q_{functional})$
$ELSE\ notify(X,e_1,q_{middleware},q_{functional})$

For example, the cost of an event delivery can be defined in terms of a weighted combination of an available link bandwidth, link propagation delay etc.

The important remark is that network QoS are highly dynamic and implementation of this rule shall consider oscillation of QoS observation in order to avoid "thundering herd" effect. This is effect, in which the least-loaded link quickly becomes overloaded because of large amount of workload it receives until new load information is gathered [14].

6. THE FUNCTIONAL DEMONSTRATOR

In order to show our concept, we have developed the functional demonstrator of the quality-constrained routing in the pub/sub system. We assumed that one subscriber issues the subscription, which matches all the publications produced by four publishers. The application of the rule (1) determines all viable routes connecting four publishers and one subscriber. For example, in figure 5 all viable routes are: publisher-A-C-E-subscriber; publisher-C-E-subscriber; publisher-B-C-E-subscriber; publisher-D-C-E-subscriber (solid line). In the following step the application of the rule (2) constrains all viable routes only to those, which satisfy middleware QoS. For example, in the figure 6 we show that one publisher's QoS does not match required by subscriber QoS (deadline 55ms>35ms). Thus, the route connecting this pair of publisher-subscriber is constrained (dotted line - route A-C in figure 6). The application of the rule (3) selects the publishers in terms of functional quality of events. In the figure 7 we show that one of the publishers produces worse quality events than others (accuracy='bad'). Hence, the route connecting this publisher with subscriber is constrained (route D-C at figure 6). The figure 8 shows the application of the rule (4). Due to the greater link bandwidth (55Mb/s>2Kb/s), the link B-C has been constrained. Finally, the route-connecting publisher-subscriber is determined: publisher-C-E-subscriber (solid line in figure 8). Events, produced by other three publishers, are filtered (not further forwarded) on the routers B, A, D and. The concept of event's filtering prevents the pub/sub system from redundant events delivery improving system scalability. Hence, it results in a saving subscriber's resource and network bandwidth. The filtering brings in better results when performed as close to the event's source as possible.



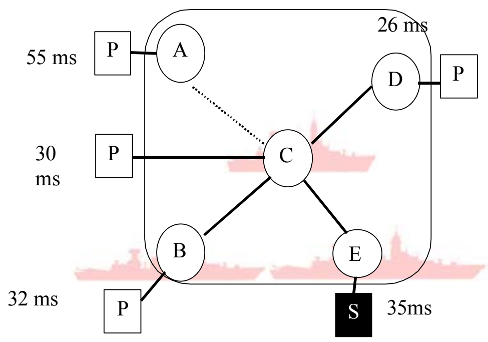
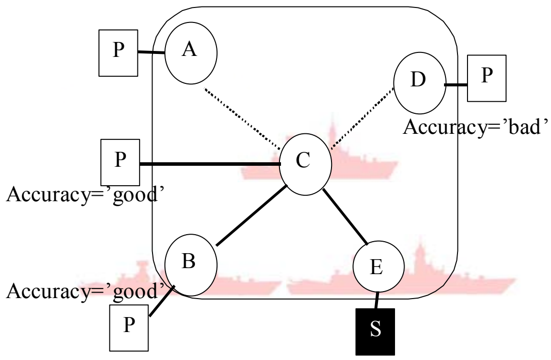


Fig. 5. Application of the rule (1)

Fig. 6. Application of the rule (2)



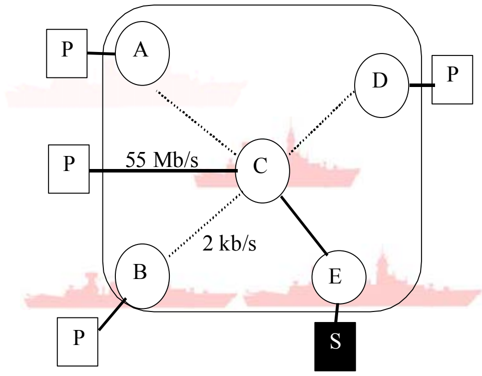


Fig. 7. Application of the rule (3)

Fig. 8. Application of the rule (4)

7. RELATED WORK

QoS support in the pub/sub systems has been considered in [1]. It advocates that QoS parameters should not be embedded on the type or content of the events. It shows the different approach to ours, because we state the QoS shall be embedded in events. The paper [1] proposes a model that supports the decoupling of QoS characterization from the event characterization. The key idea of the paper [1] is similar to ours, to subscribe information not only by subject and content but also according to QoS parameters. It also proposes three-layer architecture: applications, message broker, network. However, we noticed that the following shortcomings of this work: does not address quality support for pub/sub routing; it couples application layer with network layer (applications define QoS parameters like latency or bandwidth) and does not consider any middleware QoS specified in [15].

DDS specification [15] defines the general middleware QoS. However it does not cover any of the following issues, which we deal with in this work: network QoS, functional qualities and pub/sub routing.

The Java Message Service (JMS) API defines a common set of interfaces and associated semantics for messaging-oriented middleware providers. The JMS specification deals with the basic messaging communication. Since this is not sufficient solution for the most of business domains, different vendors propose their own solutions. For example, [11] includes Q-o-S options for message delivery or route optimization, which allows destinations to be reached within the system by message senders regardless of connection and topology changes. Those are proprietary and technical solutions without general notation and meaning. To best of our knowledge there are no related publications that address the quality-constraints integration into a pub/sub routing for the dynamic systems.

8. CONCLUSIONS

In the paper, we argumented that pub/sub routing decision, based on the subscription, is not sufficient solution for dynamic environments with applications that have quality demands. The routing decision shall depend on additional criteria – quality-constraints. We identified three abstraction levels for these quality constraints: middleware, network QoS and functional qualities. In order to support these constraints, we extended the syntax of pub/sub system and proposed four general guidelines for routing path determination. The implementation of these guidelines shall result in the routing paths determination satisfying quality-constrained.

The proposed concept of events path determination brings in the following benefits:

1) Ensures events delivery satisfying application's quality demands.

2) Improves scalability of the system due to selecting a source of events. This filtering method prevents the pub/sub system from redundant events delivery.

3) Optimises use of the network resources due to events routing accordingly to the network QoS and topology.

The presented concept of constrained routing is integrated within pub/sub routers, what gives better results than integrating it in application level. This does not bind participants together by means of any quality-contract/agreement. Thus, it preserves decoupling between publishers and subscribers, which is the major benefit of pub/sub communication model.

Future work is on the formalization of description of the quality-constrained routing. We also plan to develop, using Cnet network simulator[3], the implementation of quality-constrained routing, apply it to the high-load, tracking scenario and obtain quantitative measures.

ACKNOWLEDGMENTS

This work of B. Zieba was partly supported by European Research Programme: Marie Curie Host Fellowship under the contract number: HPMI-CT-2002-00221.

This work of M. Wegdam and M. Sinderen was partly supported by the European Union under the E-Next Project FP6-506869.

[3] Cnet network simulator enables experimentation with various data-link layer, network layer, routing and transport layer networking protocols in networks consisting of any combination of point-to-point links and IEEE 802.3. (http://www.csse.uwa.edu.au/cnet/)

We want to thank colleagues at Thales Nederland for the valuable comments and advises about my research: Maurice Glandrup, Federico Perea, Jose Benito, Wojciech Mlynarczyk and Jacek Skowronek.

REFERENCES

[1] Araujo F., Rodrigues L.- Quality of Service in Indirect Communication Systems - Fourth European Research Seminar on Advances in Distributed Systems (ERSADS'01)

[2] Banga G., Druschel P. and J. C. Mogul. Resource Containers: A New Facility for Resource Management in Server Systems. Proc. of the 3rd USENIX Symposium on Operating Systems Design and Implementation (OSDI), New Orleans, LA, February 1999.

[3] Braden R., Zhang L., Berson S., Herzog S. and Jamin S. - Resource ReSerVation Protocol (RSVP) - Version 1 Functional Speci¯cation. Internet Draft, August 1998.

[4] Campbell A., Coulson G.- A QoS Adaptive Transport System: Design, Implementation and Experience. Forth ACM International Conference on Multimedia (ACM Multimedia 96), November 1996.

[5] Carzaniga A. - Architectures for an Event Notification Service Scalable to Wide-area, PhD Thesis, Politecnico di Milano, 1998

[6] Carzaniga A., Rutherford M.J, Wolf A.L. - A Routing Scheme for Content-Based Networking - In Proc. of IEEE INFOCOM 2004

[7] Carzaniga A., Wolf A.L. - Fast Forwarding for Content-Based Networking – University of Colorado Department of Computer Science, Technical Report CU-CS-922-01 November 2001

[8] Carzaniga A.,. Rosenblum D. R and Wolf A. L.. Challenges for distributed event services: Scalability vs. expressiveness. In Engineering Distributed Objects '99, May 1999.

[9] Casati F. et. al. - Adaptive and Dynamic Service Composition in eFlow. HP technical. Report HPL-2000-39, March 2000.

[10] CISCO Systems - Internetworking Technologies Handbook

[11] Commerce One's solutions website: http://www.commerceone.com/index.html

[12] Crawley E., Nair R., Rajagopalan B. and Sandick H., - A Framework for QoS-based Routing in the Internet, RFC 2386, August 1998.

[13] Cugola G., Jacobsen H.A - Using Publish/Subscribe Middleware for Mobile Systems ACM SIGMOBILE Mobile Computing and Communications Review archive;Volume 6 , Issue 4 (October 2002); Pages: 25 – 33

[14] Dahlin M. -Interpreting Stale Load Information, IEEE Transaction on Parallel and Distributed Systems, Vol 11, No 10, Oct 2000.

[15] Data Distribution Service for Real-Time Systems Specification, ptc/03-07-07

[16] Eugster Th. Elber - The many faces of publish/subscribe. Technical report, Swiss Federal Institute of Technology in Lausanne (EPFL), 2001.

[17] Fiege L., Zeidler A. - Dealing with Uncertainty in Mobile Publish/Subscribe Middleware -proceedings of Middleware 2003, International Workshop on Middleware for Pervasive and Ad-Hoc Computing (MPAC)

[18] Frølund S., Koistinen J.- Quality-of-Service Specification in Distributed Object Systems Software Technology Laboratory HPL-98-159 September, 1998

[19] Gu X.- Spidernet - A Quality-aware service composition middleware. Ph.D thesis, University of Illinois at Urbana-Champaign, 2004.

[20] Houtsma M. - Advanced Situation Assessmant Support in Maritime C4I Systems , NATO SCI-116 Symposium on Multi-Platform Integration of Sensors and Weapon Systems for Maritime Applications, Norfolk, VA, Oct. 2002

[21] Muhl G.- Large-Scale Content-Based Publish/Subscribe Systems – Dissertation Vom Fachbereich Informatik der Technischen Universitat Darmstadt 2002;

[22] Mullender S. – Distributed Systems - @ 1993 the ACM Press.

[23] Nahrstedt K., Smith J. M. Design, Implementation and Experiences with the OMEGA End-point Architecture. IEEE Journal on Selected Areas in Communication, vol. 17, num. 7, pp. 1263-1279, September 1996.

[24] Pardo-Castellote G. –OMG Data Distribution Service: Real-Time Publish/Subscribe Becomes a Standard – RTC Magazine (www.rtcmagazine.com)– January 2005.

[25] Skowronek J., van't Hag - J.H. Evolutionary software development , NATO ESD conference 2001

[26] Working Draft for Open Distributed Processing – Reference Model - Quality of Service, ISO, Report: ISO/IEC JTC 1/SC 21 N 10979 Ed 6.4, Berlin.

[27] Younis O., Fahmy S.- Constraint-Based Routing in the Internet: Basic Principles and Recent Research - IEEE Communications Surveys & Tutorials 5 (2003)