A Practical Verification of Protocol and Data Format Negotiation Methods in ComSS Platform

Łukasz Falas, Patryk Schauer(✉), Radosław Adamkiewicz, and Paweł Świątek

Department of Computer Science and Management,
Wroclaw University of Science and Technology,
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
patryk.schauer@pwr.wroc.pl

Abstract. The main aim of the research presented in this paper was to perform a practical verification of protocol and data format negotiations between streaming services with the use of Future Internet research infrastructure providing network isolation and tools for proper measurement of the key parameters. The paper presents a brief introduction to the ComSS Platform and automated protocol and data format negotiation methods, followed by in depth analysis of the used negotiation scheme and management message exchange protocol efficiency, as well as network traffic analysis related to these communication protocols.

Keywords: Data stream processing · Composite data stream processing services · Communication protocol negotiation · Data format negotiation

1 Introduction

The concept of composite streaming services extends the well-known Service Oriented Architecture paradigm by introducing tools and techniques for data stream processing, which are compliant with the general vision of service orientation. One of most common use cases for such services is the online processing of multimedia streams. For instance, composite streaming service can be used for video and audio data streams (gathered from cameras connected to a surveillance system) processing, aimed at detection of unauthorized access to a certain area or at detection of other incidents requiring reaction. In this example, composite service could consist of a service responsible for gathering data from the cameras, from services responsible for video and audio analysis focusing on anomaly detection, from a decision making service which would raise an alarm or contact system operator on the basis of analysis services output and from data storing service which would save gathered data in the cloud storage.

One of the key requirements for composite streaming services execution is the assurance of proper and compatible communication protocols and data formats used for data stream transfer between different services. In order to ensure the required compatibility, we are introducing tools and mechanisms for automated negotiation between

J. Świątek and J.M. Tomczak (eds.), *Advances in Systems Science*, Advances in Intelligent Systems and Computing 539, DOI 10.1007/978-3-319-48944-5_16

different atomic services, during which the communication between them is established. The negotiations can be used, for instance to choose a common communication protocol and stream bitrate considering the available communication resources.

The main goal of the experiments presented in this article was to evaluate different negotiation methods, necessary during composite streaming services execution, which we have implemented in our solution for Composition of Streaming Services (ComSS) platform. Depending on the amount of available resources different methods of establishing connections between atomic streaming services were applied. The various methods differed in execution time and stream quality parameters like bitrate. Performed tests of different negotiation methods in various network and computing environments (differing in their infrastructure and in the amount of available resources), allowed us to verify resource requirements for each of the tested methods in different environments. Additionally, due to utilization of network traffic generator available in one of the testbeds, we were able to verify the efficiency of our solution and to determine its vulnerability to extreme network conditions. Also, the utilization of distributed environment offered by PLLAB2020 infrastructure allowed us to perform tests in environment which can be configured to mimic real environment conditions for using composite streaming services.

Experimental scenarios which were conducted focused on measuring the time and quality of the negotiation process, executed during establishing connections between atomic services of composite streaming service. The planned scenarios differed in:

- composite service structure (depending on number of serial and parallel sub-structures present in the general composite service structure the quality parameters can differ significantly),
- resources availability:
  - network connections parameters (and characteristic of background traffic),
  - computing resources.

In each scenario different negotiation methods were tested. Additionally influence of background traffic on quality and time of negotiations was also evaluated.

2 Related Works

Such services, especially from multimedia and Internet of Things domain, run constantly basing on processing an ingoing stream of data. Work on the distribution of streaming services can be found in [1], where authors describe methods for processing sensor data or in [2, 3] introducing specialized middleware for data stream processing. The natural distribution of the source of the stream and its destination was extended via introduction of more computing services in between [4, 5], introducing composite stream processing services in eHealth, rehabilitation and recreation fields. Many application from eHealth domain like sportsman and patient monitoring may work as streaming services [6]. Also some examples can be found in computational science and meteorological applications, where there are multiple data sources and multiple recipients interested in the processed data stream [7]. This work is a continuation of our

previous works and presents results concerning the efficiency of ComSS Platform communication [8].

3 ComSS Platform

3.1 Platform Architecture

SOA paradigm assumes the existence of many atomic services, realising defined functionalities. Such services are executed in different locations within the infrastructure, which allows them to communicate and exchange data. It is necessary to use some middleware software to provide composite functionality of processing data streams (presented in Fig. 1) [9]. Such a tool should provide the following functions:

- service structure generation,
- accurate atomic services selection,
- services preparation for execution.

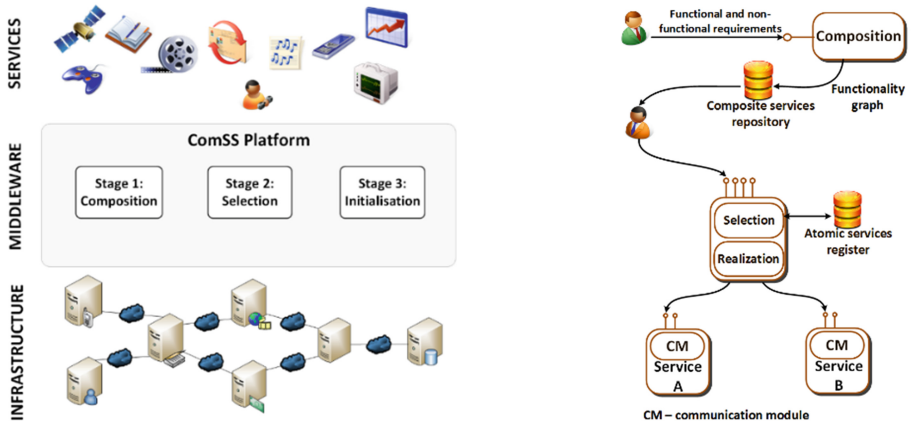

Fig. 1. ComSS Platform environment and process stages and ComSS Platform typical use

All features, mentioned in the previous section, describe a prototype environment called ComSS Platform (COMposition of Streaming Services). Figure 1 shows also a general use of the proposed tool.

Construction of a composite streaming service using discussed platform, consists of three steps, composition, selection and initialisation, which were described in [10]. Each stage corresponds to one function of previously described middleware application. Based on SOA architecture, all of them were implemented as a services [11].

In this article we focused on the last stage of composite streaming designing - service initialisation (also referred to as service set-up). Each atomic service has to be configured and initialised. Because of the distributed environment, this process bases on communication. Procedure prepares all services to realise their functions with other

services as one composite service. The input of this stage is a services graph. Except the connection structure and input/output configuration of each service, the graph contains identifiers (addresses) of each services and formats available on every interface.

Each service class is described by input and output data types, what is needed in the composition stage. Each service, which belongs to the class can operate on different formats of a stream. For instance, data type stream from accelerometer sensors can be represented by different formats, such as Manufacture A Acc Stream or Manufacture B Acc Stream. Both streams contain the same information, but they are represented in a different way. This differences may impact non-functional parameters, especially connected with quality of service. Additionally some services could have functionality of converting one formats into another, but other could not do that. This can result in the failure of negotiation, when two services operate on the same type of a data, but they do not handle common format. During service configuration, data formats should be determined and all possible problems with connections should be solved.

3.2 Automated Communication Protocol and Data Format Negotiation Methods

Service connections parameters, especially formats, can be chosen by two general methods. First is based on auto-negotiation between services. Second assumes, that formats are selected centrally with the use of planning methods. In both cases composite service orchestrator is needed to control the whole initialisation process, which is divided into three steps: resources reservation, service configuration and composite services execution.

We propose two methods of auto-negotiation and two methods of planning [12]. First auto-negotiation method is ad-hoc negotiation. These methods focused only on pairs of services, which have to be connected. In some cases, whole connection graph analysis is necessary because of relations between formats availability. For this reason we propose method of sequential negotiations, in which services communicate with each other step by step.

When using planning based method, parameters selection is realised during the resources reservation. In such case the decision about using formats is taken centrally. Two planning algorithms which base on graph search were implemented.

In our experiment we were focusing on analysis of network communication issues of these methods. Due to the fact, that all of these methods use exactly the same communication protocol, which was presented in our previous works, all of the experiments were conducted for the ad-hoc negotiation method.

4 Experiment Set-up

The experiment was conducted on the PLLAB2020 research infrastructure [13]. In our experiment we have utilized 21 virtual machines (20 machines for services used during tests, 1 machine for our ComSS management software). In order to perform the test and

measure the network traffic in proper isolation we have set up 38 dedicated virtual networks:

- 20 networks were used for our solution's control messages transmissions managing the execution of composite data stream processing services
- 18 networks were used for inter-service communication and communication protocol and data format negotiations

For our experiment we have prepared 4 different composite service workflows which cover majority of communication patterns which we are expecting to encounter in production environments. The aim of the experiment was to test the communication modules of services built on our framework for data stream processing services, which enable automated protocol and data format negotiation and provide interfaces for remote management of execution of composite data stream processing services with our management software (ComSS). For each of the services in the workflow we were measuring the parameters of negotiation message exchange with other services participating in the composite data stream processing workflow and the parameters of management message exchange with the management software responsible for orchestrating the composite service execution.

The schematic overview of the composite service workflows are presented below (Figs. 2, 3, 4 and 5).

As it is depicted on the diagrams, in each of the workflows we selected links on which we were analysing the network traffic with the support of a dedicated analysing device (DAG) and a traffic generator (SPIRENT) to test how background traffic affects network connectivity and message exchange in our solution.

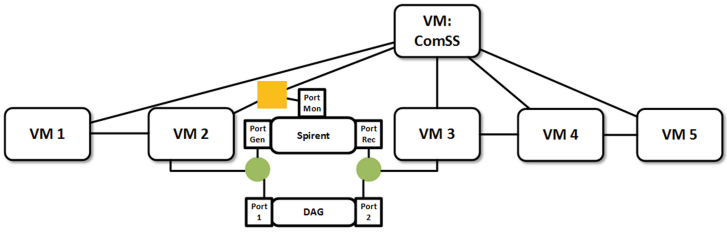

Fig. 2. General schema of the first composite data stream processing workflow

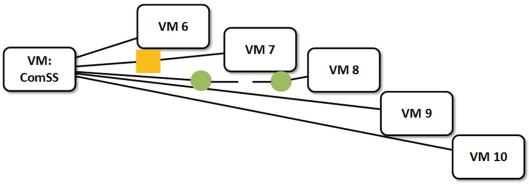

Fig. 3. General schema of the second composite data stream processing workflow

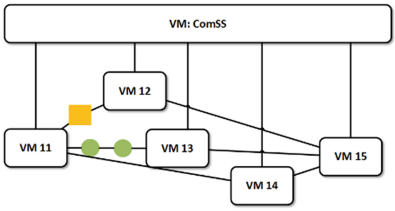

Fig. 4. General schema of the third composite data stream processing workflow

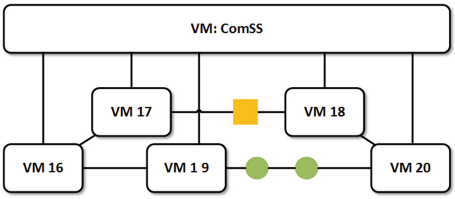

Fig. 5. General schema of the fourth composite data stream processing workflow

In order to thoroughly verify our communication modules and automated protocol and data format negotiation methods we have prepared a set of evaluation tests. The tests differed in following environmental parameters:

- selected composite data stream processing workflow (**g1**, **g2**, **g3**, **g4**)
- number of individual user requests sent during test (**i10**: 10 requests, **i50**: 50 requests, **i100**: 100 requests)
- number of feasible data formats, which were used during negotiations (**f03**: 3 formats, **f14**: 14 formats, **f21**: 21 formats)
- characteristic of background network traffic (**r1**: 0 % of background traffic, **r2**: 95 %, **r3**: 99 %, **r4**: 100 %)

The tests were conducted for all combinations of the described parameters which resulted in a total of 144 different test scenarios. Each of the tests was repeated 5 times (which gives a total number of 720 test runs), and the results from the repeated test runs were averaged.

During the performed tests we have measured the following parameters:

- Management message exchange time at 3 different stages of composite data stream processing service set up and execution process for each connection between services and our management software:
  - resource reservation and negotiation interface setup message exchange (reservation stage),
  - inter-service communication configuration message exchange (configuration stage),
  - composite service execution control message exchange (control stage).
- Protocol and data format negotiation message exchange time between each of the interconnected services.

- Message exchange traffic characteristics (data stream throughput variability) on management links (between management software and services).
- Message exchange traffic characteristics (data stream throughput variability) on negotiation links (between services participating in composite data stream processing workflow).

5 Experiment Results

In this section of the report we have describe the results obtained during the experimentation on PLLAB2020 infrastructure. Due to the large number of tests, which resulted in a relatively big set of test results, we are only presenting a selection of experimental results. Also, during the test we have noticed that the test results from different workflows does not differ considerably, hence in this report we only focused on presenting the results from workflow 1 (which results were similar to results of workflow 3 and 4, probably due to the fact that in all cases management and communication are independent, hence the structure of the workflow does not influence performance of point to point communication between services and between services and management component) and the results of workflow 2. It is also worth noting that in case of workflow 2 we are only analysing the communication with management software, due to the fact that this workflow is completely parallel and, due to this fact, it does not require any inter-service negotiations. In order to prove our solution usefulness in production environments, we assumed that that the communication time between services during the negotiations should, on average, be lower than 600 ms and we expected that the average aggregated time of communication during three phases of composite service initialization (reservation, configuration and control phases) should not exceed 1 s.

Table 1. Aggregated communication times required for composite data stream processing services set-up for a stream of 10, 50 and 100 consecutive user requests.

			Workflow 1.				Workflow 2.			Workflow 1.				Workflow 2.	
			max init	max neg	mean init	mean neg	max init	mean init		max init	max neg	mean init	mean neg	max init	mean init
i50	**f03**	***r1***	16,00	15,37	14,20	13,44	40,15	33,18	**i100**	31,00	29,67	28,70	27,72	72,09	68,14
		r2	32,00	31,08	29,10	28,24	76,58	75,27		57,00	56,56	46,50	45,83	152,45	148,36
		r3	32,50	30,53	28,20	27,37	77,27	74,91		58,00	57,67	55,40	54,79	152,90	121,37
		r4	29,50	28,54	27,90	27,37	77,71	75,78		60,00	58,68	56,60	55,80	151,31	148,93
	f14	***r1***	16,50	15,23	12,90	12,41	36,07	33,49		30,00	29,90	26,90	26,52	79,89	70,34
		r2	30,00	28,83	28,60	28,11	79,12	76,29		56,50	55,65	54,70	53,91	152,59	122,94
		r3	32,00	28,97	28,40	27,14	78,59	77,45		55,50	53,76	54,20	53,49	153,41	122,81
		r4	36,00	35,84	29,70	29,50	79,02	75,93		57,50	55,40	53,90	53,56	151,99	123,83
	f21	***r1***	16,50	15,07	14,40	13,37	35,82	33,05		65,50	60,75	48,50	43,71	70,16	65,59
		r2	6,00	4,98	3,80	3,30	79,68	75,90		9,50	8,78	7,80	7,00	150,85	121,63
		r3	5,00	5,18	3,50	3,32	80,67	76,93		10,00	7,75	7,10	6,34	153,05	149,54
		r4	29,50	29,40	27,80	27,31	77,27	75,87		59,50	57,45	55,20	54,72	149,92	148,18

We have started the analysis of experiment results with an examination of aggregated communication times required for proper set-up of the composite data stream processing services for a stream of 10, 50 and 100 consecutive user requests. All of the times were measured along with the analysis of network traffic characteristics performed by the DAG traffic analyser.

Table 1 shows the mean and maximum time of both, composite data streaming service set-up times and inter-service protocol and data format negotiation communication. The results presented in the table show that, as it was expected, the observed aggregated communication time required for composite data stream processing service set-up increases along with the increase of background traffic. However, it is worth noting that the increase in observed time is less significant than we expected (which is even more evident in case of average processing time for single request presented further). The measured communication time with r2, r3 and r4 background traffic still meets our QoS expectations. Also, there are no significant differences between times measured with r2, r3 and r4 background traffic, which proves that our communication modules and communication methods are reliable and retain required performance even at very high network loads. We can also observe that the increase in the number of format does not impact the performance of network communication significantly, even in case of high network load (Fig. 6).

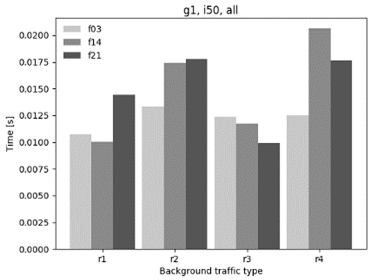

Fig. 6. Average aggregated communication time value of reservation, configuration and control stages times for workflow 1.

After the analysis of aggregated communication times required for proper set-up of the composite data stream processing services for a stream of 10, 50 and 100 consecutive user requests, we have started to analyse average request processing communication times for single request in different stages (reservation, configuration and control) of the set-up process as well as aggregated values of all three of these stages. Figure 7 presents selected times measured during the tests performed on workflow 1, which were measured by our software.

The results show that the measured average communication times for each of the stages are relatively low, as well as the aggregated time of all three stages and the communication time required for composite data stream processing service set-up meets our current requirements. It is also important that values in different cases are similar, hence the variability of network traffic and variety of network formats does not

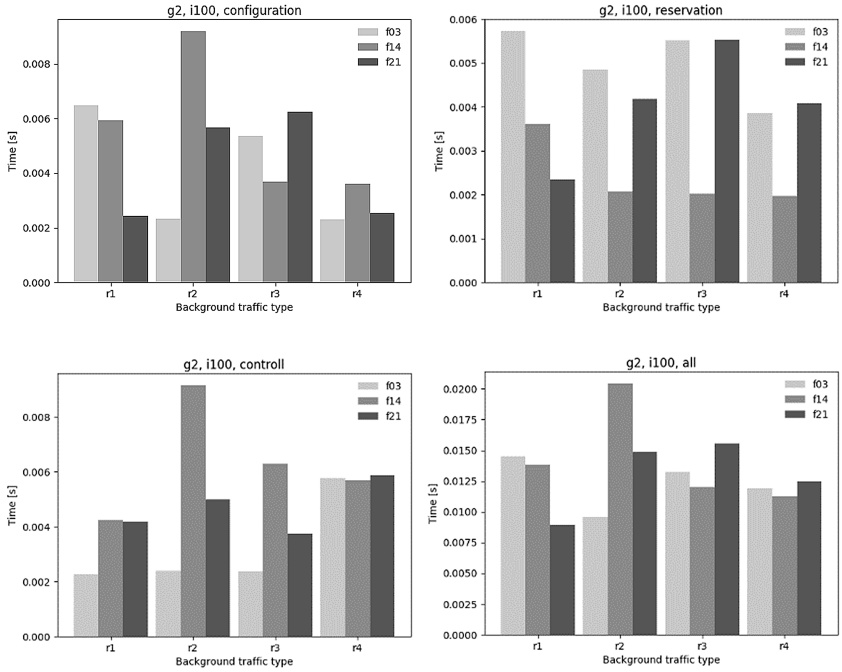

Fig. 7. Average reservation, configuration, control stage communication time and aggregate time of 3 stages for workflow 2.

have a direct performance impact on our solution. Some degree of small scale variability can be observed on the provided charts, however it may be a result of the fact that the testing infrastructure was not fully deterministic. Due to this fact we have encountered some outliers in our measurements, however we have decided that we will not exclude them from the final results.

The results obtained during the analysis of workflow 2 (as well as 3 and 4) were similar to workflow 1. The communication times obtained during the tests of 2 workflow were also relatively low and meet our expectations. Unfortunately, in this case we also encountered some small scale variability, due to the presence of outliers in our measurements, similarly to the workflow 1 (Figs. 8 and 9).

The utilization of SPIRENT traffic generator and DAG analyser allowed us to perform the analysis of traffic characteristics measured in different test cases. The measured characteristics were presented on charts above. Charts related to 1st workflow present the characteristics on both, management and negotiation links. For 2nd workflow measurements were performed only on management link.

In case of reference r1 background traffic (0 %) the maximum observed throughput reached approximately 15 kB/s (120 kb/s). This result is in line with our supposition that we should be able to achieve a relatively low network usage, due to utilization of our dedicated communication modules and communication protocols. As a result of our

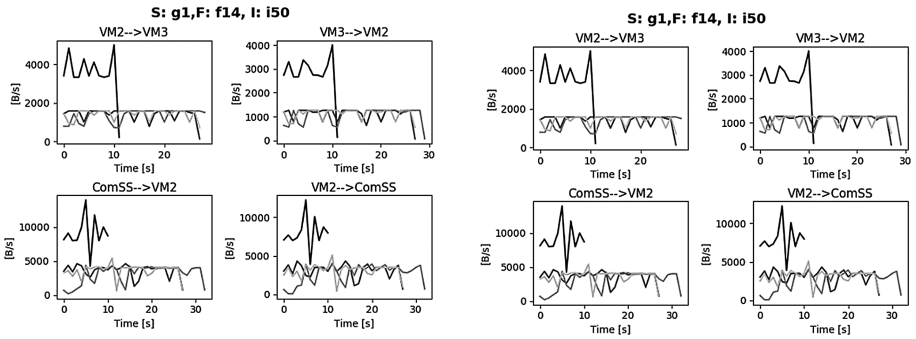

Fig. 8. Network characteristics for negotiation and management message exchange between negotiation modules of services for 14 data formats, 50 and 100 user requests (**black** – r1 background traffic, **dark_grey** – r2, **grey** – r3, **light_gray** – r4).

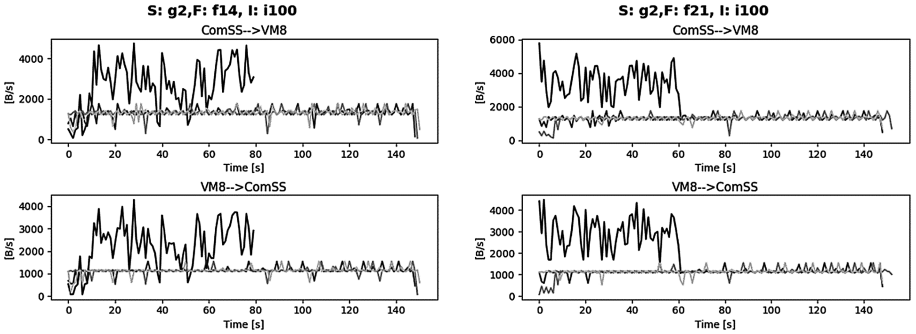

Fig. 9. Network characteristics for management message exchange between negotiation modules of services for 14 and 21 data formats and 100 user requests (**black** – r1 background traffic, **dark_grey** – r2, **grey** – r3, **light_gray** – r4).

optimization, even in case of heavy background traffic (95 %, 99 %, 100 %) observed throughput was similar. In majority of cases it did not exceed 5 kB/s (40 kb/s).

In case of the second workflow the results for management link were relatively similar. The measured throughput on configuration link without background network traffic, on average, did not exceed 6 kB/s (48 kb/s), while with background traffic message were exchanged with an average throughput of 2 kB (16 kb/s). All of these results met our expectations related to the QoS requirements for our solution. An interesting fact related to this experiment, is that in high network load cases the background traffic caused traffic shaping on the link. This result was related to the uniform characteristic of background traffic cause, which smoothed the traffic related to management message exchange.

The key conclusion of the network traffic characteristics analysis is that even in high network load scenarios, when the network links are almost completely reserved, our negotiation modules and protocols retain their required performance and are reliable, enabling efficient service set-up with composite data stream processing.

6 Conclusions and Lessons Learned

The possibility to perform the described experiment enabled us to fully evaluate and verify our protocol and data format negotiations in an isolated and controlled environment which simulates real world use cases. From strictly technical perspective, the conducted tests allowed usto improve our solution and fix some of critical flaws and bugs in the implementation of our methods in communication modules, which were decreasing our methods' performance and which, in some specific cases, resulted in a very high number of packet retransmissions.

The results of the experiments which we have performed after the removal of critical flaws in our software have shown, that the methods we have developed are performing in line with our expectations. The tests have also shown, that the structure of the workflow does not have a considerable impact on the performance of the communication modules and protocol and data format negotiation methods. Following that, the tested protocols and negotiation methods retain their reliability and performance even in case of larger number of communication formats and in case of high network load. Also, the average, as well as maximum, measured times never exceeded to required QoS levels, which proved that the tested methods are reliable. The tests have also shown that the introduction of adaptable protocol and data format negotiation methods does not introduce a considerable delay to the process of composite data stream processing service set-up, hence all of the developed methods can be considered as viable for utilization in production environments.

References

1. Gu, X., Yu, P., Nahrstedt, K.: Optimal component composition for scalable stream processing. In: 2005 Proceedings of the 25th IEEE International Conference on Distributed Computing Systems, ICDCS 2005, pp. 773–782, June 2005
2. Chen, L., Reddy, K., Agrawal, G.: Gates: a grid-based middleware for processing distributed data streams. In: 2004 Proceedings of the 13th IEEE International Symposium on High performance Distributed Computing, pp. 192 – 201, June 2004
3. Schmidt, S., Legler, T., Schaller, D., Lehner, W.: Real-time scheduling for data stream management systems. In: 2005 Proceedings of the 17th Euromicro Conference on Real-Time Systems, (ECRTS 2005), pp. 167–176, July 2005
4. Gu, X., Nahrstedt, K.: On composing stream applications in peer-to-peer environments. IEEE Trans. Parallel Distrib. Syst. **17**(8), 824–837 (2006)
5. Rueda, C., Gertz, M., Ludascher, B., Hamann, B.: An extensible infrastructure for processing distributed geospatial data streams. In: 2006 18th International Conference on Scientific and Statistical Database Management, pp. 285 –290 (2006)
6. Świątek, P., Klukowski, P., Brzostowski, K., Drapała, J.: Application of wearable smart system to support physical activity. In: Advances in Knowledge-based and Intelligent Information and Engineering Systems, pp. 1418–1427. IOS Press (2012)
7. Liu, Y., Vijayakumar, N., Plale, B.: Stream processing in data-driven computational science. In: 7th IEEE/ACM International Conference on Grid Computing, pp. 160–167, September 2006

8. Stelmach, P., Świątek, P., Schauer, P.: Communication protocol negotiation in a composite data stream processing service. In: Swiątek, J., Grzech, A., Swiątek, P., Tomczak Jakub, M. (eds.) AISC, vol. 240, pp. 681–690. Springer, Heidelberg (2014). doi:10.1007/978-3-319-01857-7_65
9. Alamri, A.: Cloud-based e-health multimedia framework for heterogeneous network. In: 2012 IEEE International Conference on Multimedia and Expo Workshops (ICMEW), pp. 447–452 (2012)
10. Świątek, P., Schauer, P., Kokot, A., Demkiewicz, M.: Platform for building eHealth streaming services. In: 2013 IEEE 15th International Conference on e-Health Networking, Applications & Services (Healthcom), Lisbon, pp. 26–30 (2013)
11. Świątek, P., Stelmach, P., Prusiewicz, A., Juszczyszyn, K.: Service composition in knowledge-based SOA systems. New Gener. Comput. **30**, 165–188 (2012)
12. Stelmach, P., Świątek, P., Falas, Ł., Schauer, P., Kokot, A., Demkiewicz, M.: Planning-based method for communication protocol negotiation in a composition of data stream processing services. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2013. CCIS, vol. 370, pp. 531–540. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38865-1_53
13. Binczewski, A., et al.: Infrastruktura PL-LAB2020, Przegląd Telekomunikacyjny, Wiadomości Telekomunikacyjne R, 88(12), s. 1399–1404 (2015)