The DEBS 2012 Grand Challenge

Zbigniew Jerzak*
SAP Research, SAP AG
Chemnitzer Straße 48
01187 Dresden, Germany
Zbigniew.Jerzak@sap.com

Thomas Heinze
SAP Research, SAP AG
Chemnitzer Straße 48
01187 Dresden, Germany
Thomas.Heinze@sap.com

Matthias Fehr
Infineon Technologies
Königsbrücker Straße 180
01099 Dresden, Germany
Matthias.Fehr@infineon.com

Daniel Gröber
Infineon Technologies
Königsbrücker Straße 180
01099 Dresden, Germany
Daniel.Groeber@infineon.com

Raik Hartung
SAP Research, SAP AG
Chemnitzer Straße 48
01187 Dresden, Germany
Raik.Hartung@sap.com

Nenad Stojanovic
FZI Research Center for
Information Technology
Haid-und-Neu-Straße 10-14
76131 Karlsruhe, Germany
nstojano@fzi.de

ABSTRACT

The goal of the DEBS Grand Challenge series is to contribute to the Event Processing Grand Challenge, that serves as a common goal and mechanism for coordinating research focusing on event processing. DEBS Grand Challenge series provides a common ground and evaluation criteria for a competition aimed at both research and industrial event-based systems. The goal of the DEBS Grand Challenge participants is to implement a solution to a specific problem provided by the DEBS Grand Challenge organizers. In this paper we present a description of the DEBS 2012 Grand Challenge problem focusing on the high-tech manufacturing domain. Moreover we provide a set of both: (1) real-life data and (2) queries which can be used by the DEBS 2012 Grand Challenge participants as well as research community at large.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Performance, Experimentation

Keywords

event processing, streaming, cep

1. INTRODUCTION

The overall goal of the DEBS Grand Challenge series is to demonstrate the capability of event processing systems to solve problems arising in the area of event-based data management and analysis. The DEBS 2012 Grand Challenge focuses on a use case which has been developed based on the problems faced in a high-tech manufacturing industry. The DEBS 2012 Grand Challenge problem requires a continuous monitoring of the high-tech manufacturing equipment, based on the data gathered by sensors embedded within the equipment. The goal of the monitoring is to detect and record deviations from the predefined (good) system behavior. The rationale behind the use of event based systems for monitoring of manufacturing equipment is the continuous nature of sensor data and a need for a low latency detection of possible violations. Current state of the art in the manufacturing domain involves the usage of a number of independent, batch oriented systems for monitoring and analysis of sensor data – see Figure 1.

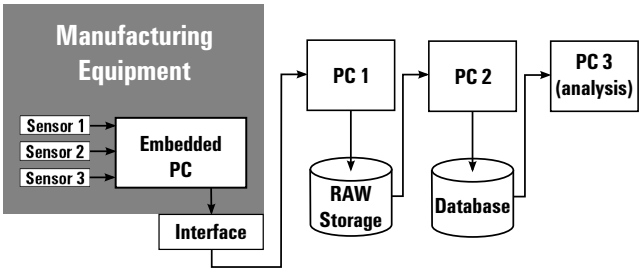

Figure 1: Current system architecture

In a typical, existing setup data is first collected from sensors within the equipment using an embedded PC. Subsequently, it is stored by the PC 1 as raw data in a flat file. This file is periodically integrated into a database by the PC 2. The analysis of data is performed using the PC 3 connected to the database. Only in this last step a possible violation can be detected. It can be observed that such setup results in a delayed response to potential violations of Key Performance Indicators (KPI). The high latency of the response (which currently reaches 30 minutes) is the major factor increasing the severity of the KPI violations and their direct monetary costs.

Therefore, the major goal of the DEBS 2012 Grand Challenge was to investigate the applicability of the event process-

*Corresponding author.

DEBS'12, July 16–20, 2012, Berlin, Germany.

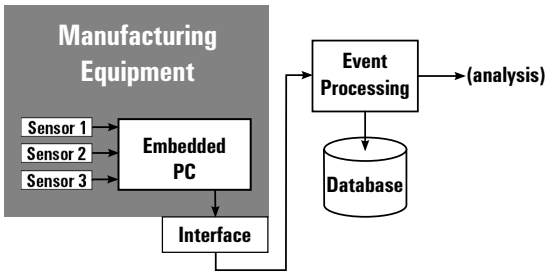

Figure 2: System architecture under evaluation in the DEBS 2012 Grand Challenge

ing systems to bridge the gap between the actual occurrence of the event causing a violation of a predefined KPI and the detection of such a violation – see Figure 2. To that end the DEBS 2012 Grand Challenge organizers have recorded a set of data originating from the real manufacturing equipment. Moreover, a set of rules (queries) has been developed which can be applied on top of the collected data in order to detect violations of the KPIs. This way participants of the DEBS 2012 Grand Challenge can measure their systems against a real-life problem using both real data and queries.

The DEBS 2012 Grand Challenge scenario requires a specific type of data analysis systems to be developed. Systems which can provide: (1) low latency response to continuous queries, as well as (2) the ability to handle large volume of streaming data in a distributed setup. In a typical high-tech manufacturing scenario (a single fabrication plant) an average of 1,000 pieces of geographically distributed manufacturing equipment are operated. This results in a 50 Terabytes of data being collected every day. This, in turn, requires an event processing system which can sustain a continuous workload of 5 Million events per second. Neither classical database systems, nor existing, batch oriented systems, such as [4, 1], were designed to cope with such scenarios. Moreover, the characteristics of the manufacturing environment require that the processing resources are shared between real-time constrained tasks, such as equipment controlling and actuation, and lower priority tasks, such as data collection, filtering and analysis. This requires the event based system to maintain high throughput, while at the same time being adaptive so as to allow for execution of higher priority tasks.

The remainder of this paper is structured as follows: in Section 2 we provide a detailed description of the recorded data as well as the data replay tools. In Section 3 we provide a description of continuous queries to be executed on top of the recorded data. In Section 4 we outline available related work focusing on benchmarking of event driven systems. In Section 5 we describe how the DEBS 2012 Grand Challenge contributes to the Event Processing Grand Challenge. We conclude with Section 6.

The description of the DEBS 2012 Grand Challenge, as well as links to data files, are available under: http://www.csw.inf.fu-berlin.de/debs2012/grandchallenge.html.

2. DATA

The DEBS 2012 Grand Challenge monitoring data originates from the high-tech manufacturing equipment. Provided data set contains 77,576,214 entries (resulting in a total uncompressed file size of 14,218,243,867 bytes) distributed across 18 days, recorded between the 22^{nd} of February 2012 and the 26^{th} of March 2012. The longest consecutive period spans from the 21^{st} of March 2012 until the 26^{th} of March 2012. A single day contains roughly 8.5 million entries, with a single entry having the size of approximately 180 bytes.

```
message CDataPoint {
   required fixed64  ts    = 1;
   required fixed64  index = 2;
   required fixed32  mf01  = 3;
   required fixed32  mf02  = 4;
   required fixed32  mf03  = 5;
   required fixed32  pc13  = 6;
   required fixed32  pc14  = 7;
   required fixed32  pc15  = 8;
   required uint32   pc25  = 9;
   required uint32   pc26  = 10;
   required uint32   pc27  = 11;
   required uint32   res   = 12;
   optional bool     bm05  = 13;
   optional bool     bm06  = 14;
   optional bool     bm07  = 15;
   optional bool     bm08  = 16;
   optional bool     bm09  = 17;
   optional bool     bm10  = 18;
   optional bool     pp01  = 19;
   optional bool     pp02  = 20;
   optional bool     pp03  = 21;
   optional bool     pp04  = 22;
   optional bool     pp05  = 23;
   optional bool     pp06  = 24;
   optional bool     pp07  = 25;
   optional bool     pp08  = 26;
   optional bool     pp09  = 27;
   optional bool     pp10  = 28;
   optional bool     pp11  = 29;
   optional bool     pp12  = 30;
   optional bool     pp13  = 31;
   optional bool     pp14  = 32;
   optional bool     pp15  = 33;
   optional bool     pp16  = 34;
   optional bool     pp17  = 35;
   optional bool     pp18  = 36;
   optional bool     pp19  = 37;
   optional bool     pp20  = 38;
   optional bool     pp21  = 39;
   optional bool     pp31  = 40;
   optional bool     pp32  = 41;
   optional bool     pp33  = 42;
   optional bool     pp34  = 43;
   optional bool     pp35  = 44;
   optional bool     pp36  = 45;
   optional bool     pc01  = 46;
   optional bool     pc02  = 47;
   optional bool     pc03  = 48;
   optional bool     pc04  = 49;
   optional bool     pc05  = 50;
   optional bool     pc06  = 51;
   optional bool     pc19  = 52;
   optional bool     pc20  = 53;
   optional bool     pc21  = 54;
   optional bool     pc22  = 55;
   optional bool     pc23  = 56;
   optional bool     pc24  = 57;
}
```

Listing 1: Event schema expressed using the GPB syntax

The DEBS 2012 Grand Challenge monitoring data has been produced by an array of sensors which are embedded withing the manufacturing equipment – see Figure 1. Every sensor used in the manufacturing equipment is either binary or analogue. Binary sensors produce data with either 0 or 1. The analogue sensors produce data in the range $0 - 2^{15}$. Each sensor produces data with a frequency of either 100Hz or 1000Hz. Data originating from each sensor is collected by a PC embedded withing the manufacturing equipment and

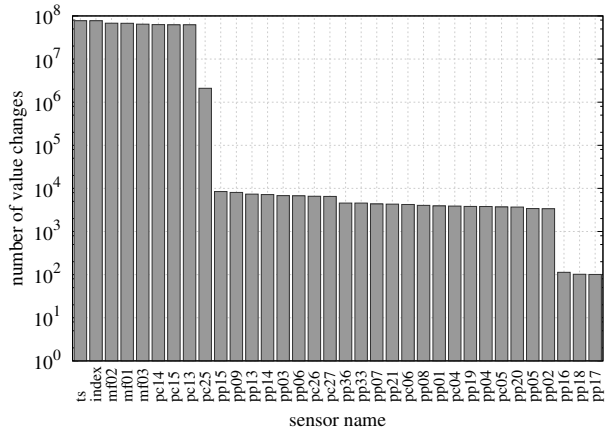

Figure 3: Number of value changes for each of the sensors

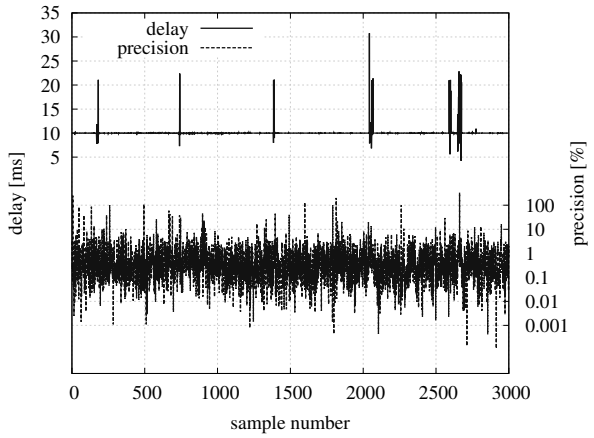

Figure 4: Delay between consecutive events and precision of the provided generator

aggregated into a single event. The embedded PC outputs events with the rate of 100Hz.

All time-related measurements (see Section 3 and Figure 4) are to be regarded with respect to the application time, encoded in the field `ts`. The maximum event rate of 100Hz implies the need for the DEBS 2012 Grand Challenge participants to be able process events with a maximum latency of 10 milliseconds.

The schema of the events produced by the embedded PC is shown in Listing 1. Each event contains a time stamp which is generated by the embedded PC upon the creation of the given event – field `ts`. The time stamp has a resolution of 10^7 Hz, meaning it is possible to differentiate between events which are at least 100 nanoseconds apart. Given the maximum sensor frequency of 1000Hz no two events have the same time stamp. Every event contains a monotonically increasing index (field `index`) which is generated by the embedded PC upon the creation of the given event. The index serves as a unique identifier of an event. Remaining fields, contained within an event, are direct copies of the values recorded by the sensors embedded in the manufacturing equipment.

```
1  ts:      2012-02-22T16:46:29.0570267+00:00
2  index:   2556010
3  mf01:    13045
4  mf02:    14391
5  mf03:    8113
6  pc13:    0065
7  pc14:    0190
8  pc15:    0149
9  pc25:    0000
10 pc26:    0000
11 pc27:    0000
12 res:     0000
```

Listing 2: First twelve fields of an example event

Fields `mf01` till `pc27` (Listing 1 — lines 3 till 12) are analogue values. Fields `mf01` till `pc15` (Listing 1 — lines 3 till 9) can take values in the range from $-(2^{15})$ till $2^{15}-1$. Fields `pc25` till `pc27` (Listing 1 — lines 10 till 12) can take values in the range from 0 till $2^{16}-1$. Field `res` is not used. Fields `bm05` till `pc24` (Listing 1 — lines 14 till 58) are binary and can take a value of either 0 or 1.

Not all fields (sensors) presented in Listing 1 produced values during the data capture period. Figure 3 shows the number of recorded value changes for each of the sensors. The higher the number of changes, the more frequently a given field (sensor) changes (records) its value in the data file. As expected, both `ts` and `index` fields change their values exactly 77,576,214 times, i.e., as often as many unique events are in the data file. Sensors, whose names are not listed along the x axis in Figure 3 are never changing their values.

The embedded PC which is recording the sensor data is not solely devoted to this task. It is also responsible for controlling of the operations of the manufacturing equipment itself. The controlling of the operations has a higher (real-time) priority and it might preempt the data collection tasks — see also Section 1. This in turn leads to the possibility that some events are recorded later than expected — this situation is illustrated in Figure 4. It can be observed that the latency between two consecutive events can be as high as 30 milliseconds, instead of standard 10 milliseconds.

In order to facilitate the use of provided data file, the DEBS 2012 Grand Challenge provides a real-time data generator to its participants. The data generator (32-bit JAR file) consumes the data file and sends the events preserving their relative latency to a specified TCP/IP socket. The precision of the generator is shown in Figure 4, with lower numbers indicating better precision. The provided generator uses Google Protocol Buffers (https://developers.google.com/protocol-buffers) to serialize the events. The DEBS 2012 Grand Challenge provides also a sample receiver code for reading of the events from a TCP/IP socket. Provided generator translates the timestamps from the textual representation shown in Listing 2 to a UNIX-like time stamp format: an unsigned 64 bit integer representing the number of nanoseconds since 1^{st} of January 1970.

The data file as well as the generator and simple receiver code for the DEBS 2012 Grand Challenge is available via HTTP: http://goo.gl/BB0uX and FTP: ftp://ftp.fu-berlin.de/science/computer/debs2012/.

3. QUERIES

In this section we present queries which were developed together with the manufacturing experts. The queries rep-

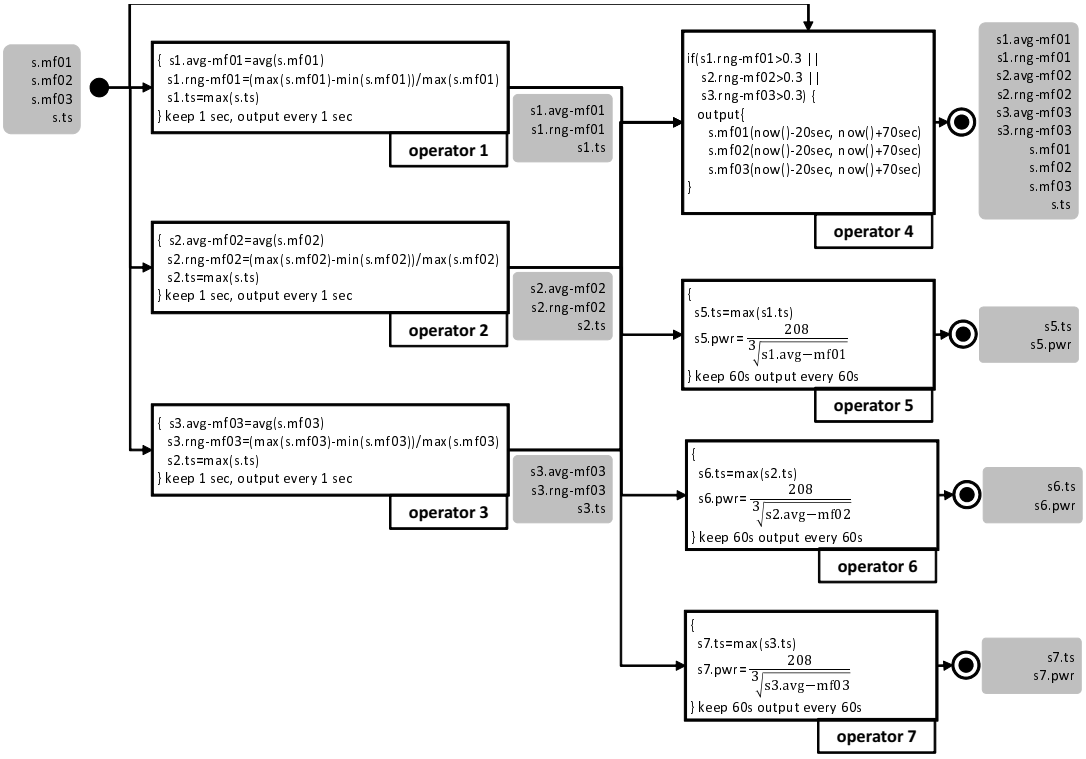

Figure 5: First query — energy monitoring KPI

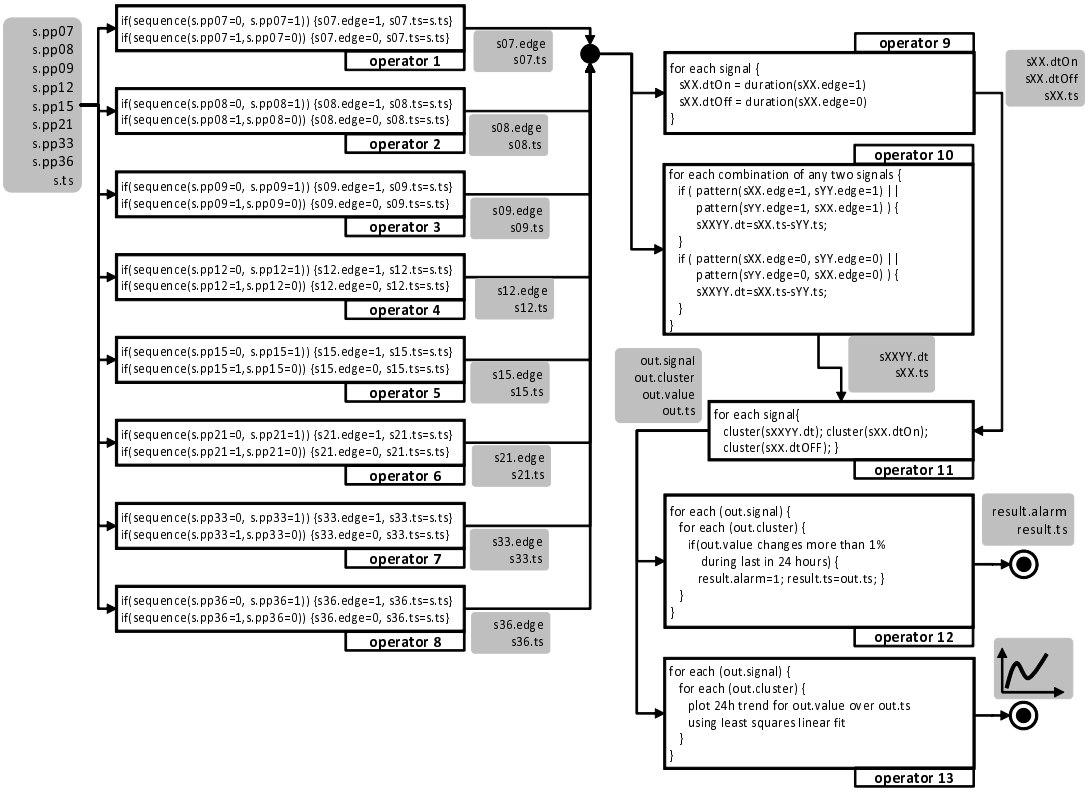

Figure 6: Second query — sensor switching times and dependencies KPI

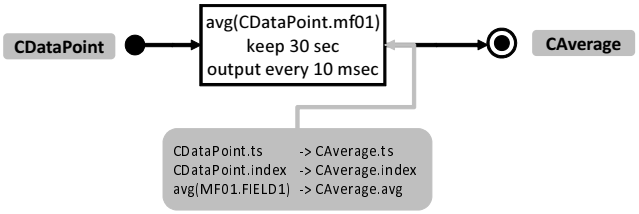

Figure 7: Sample data flow diagram

resent Key Performance Indicators (KPIs) which need to be constantly monitored for violations. Queries are described using simple block diagram notation. In our query description we assume that all queries operate on the same input event schema as defined in Listing 1. Authors would like to advise readers that the implementation of queries presented in following Sections should not to be taken literally. The query description is aimed at providing a comprehensive illustration of how given KPIs should be monitored. It is therefore possible to adjust the way the queries are implemented as long as the given objective of KPI violation monitoring is fulfilled. Specific modifications could, e.g., target query optimization techniques [5] for the sake of achieving a better system performance. Specifically, the intermediate data stream schemes can be freely modified to match the characteristics of the event based system used for implementation.

3.1 Notation

As there are many languages which can be used to express the logic of event-based systems we do not impose any of them on the DEBS 2012 Grand Challenge participants. Instead we provide a detailed description of queries using a block diagram-like approach, with the sole goal of providing a detailed description of how Key Performance Indicators are calculated.

The block diagram notation represents a graph of operators, thus breaking a potentially complex KPI computation into a sequence of simple operations. An operator graph is composed of a set of operators connected by arrows representing message flow. We assume that messages in an operator graph always flow from the left to right and that there are no backward edges. An example operator graph is shown in Figure 7. It is composed of a source with a defined message schema (`CDataPoint` — see Listing 1), an operator, schema conversion step (shaded rectangle), and a sink. Messages with the `CDataPoint` schema are being pushed from the source. The first operator calculates an average for the value of the phase `CDataPoint.mf01`. The average is calculated over a window of 30 seconds (`keep 30 sec`) and the result is being pushed out of the operator every 10 milliseconds (`output every 10 msec`). The calculated average introduces a new value which becomes a part of a new schema `CAverage.avg`. In practice, for the sake of brevity, we omit the definition of schema naming and mappings. Instead we explicitly specify those within operators.

3.2 Query 1

The goal of the first query (see Figure 5) is to monitor the energy consumption of the manufacturing equipment. The energy consumption is recorded by the sensors `mf01`, `mf02`, and `mf03`. The first set of operators (operators 1 till 3) calculates the average values for each of the sensors (`avg-mf01` till `avg-mf03`) as well as the relative variation (`rng-mf01` till `rng-mf03`) in each of the sensors readings. Both average and variation values are calculated over the period of 1 second and are output every second.

The relative variation is used to trigger the recording of the raw values of the sensor readings — see operator 4. Whenever the relative variation on any of the energy measuring sensors exceeds the threshold of 30%, the raw data from each of the sensors (`mf01-mf03`) needs to be recorded. The recording should commence 20 seconds before the occurrence of the threshold violation and end 70 seconds later. If multiple violations occur with the 90 second interval, it needs to be extended so that it always captures 70 seconds of raw data after the occurrence of the last violation and 20 seconds before the occurrence of the first violation. Operators 5 till 7 record the power consumption of the manufacturing equipment within a period of one minute.

3.3 Query 2

The goal of the second query is to monitor the behavior of `pp07`, `pp08`, `pp09`, `pp12`, `pp15`, `pp21`, `pp33`, `pp36` sensors — see Figure 5. The specific goal of the second query is to monitor the dependencies and relations between sensors and their switching times. All input data in this task is boolean. The first operation on the input data is performed by operators 1 till 8. The goal is to detect the change of state (from on, i.e., 1 to off, i.e., 0 as well as from off, i.e., 0 to on, i.e., 1) of each of the input fields `pp07` till `pp36`. The state changes are emitted along with time stamps of the state change occurrence.

The state change events are further processed by two operators: 9 and 10. The goal of the processing performed in the operator 9 is to calculate the duration of each of the states. For each of the sensor the duration of the off (0) and on (1) phase is calculated. Operator 10 calculates the time difference between raising and falling edges of each of the input sensor pairs.

The output of both operators is unified in the operator number 11, which is responsible for the clustering of the duration values. The goal of the clustering is to partition the data so that the subsequent analysis operates only the related data sets. Clustering allows to automate the process of correlation of sensors and allows for an exploration of not obvious relationships. If a sensor exposes an alternating on/off pattern, such as: 1 second on, 3 seconds off, 2 seconds on, 4 seconds off, 1 second on, etc..., clustering would result in four clusters being created, two for the on group (1 second and 2 seconds), and two for the off group (3 seconds and 4 seconds).

The clustered data, output by the operator 11, is consumed in parallel by operators 12 and 13. Operator 12 measures (for each cluster within each signal) whether the difference between the maximum duration and minimum duration of that signal has exceeded 1% within last 24 hours. If this is the case, an alarm is raised. The goal of the operator 13 is to constantly monitor the trend for the calculated cluster values. The trend should be computed using a least squares method for the period of last 24 hours. The trend monitoring can be either visualized or returned as a stream of plot parameters.

It should be assumed that the initial values of all sensors

are unknown, being either 0 or 1. The least squares method should use a linear function for fitting.

4. RELATED WORK

There exists only a few approaches which are aimed at providing a realistic scenario for evaluation of event based system. One of the best known works by Arasu et al. [2] (with over 150 citations as of May 2012) focuses on a toll collection use case. The goal of the benchmark is to simulate a variable tolling system with a variable amount of data (events) to be processed and a fixed amount of queries. Additionally, the Linear Road benchmark specifies constraints on response times which are applied as evaluation criterion. The DEBS 2012 Grand Challenge takes similar approach to Linear Road by specifying a fixed set of queries and data. However, the DEBS 2012 Grand Challenge can be scaled both in terms of number of data and queries which need to be processed. This can be achieved by increasing the number of machines to be monitored. Moreover, unlike the Linear Road benchmark, the DEBS 2012 Grand Challenge does not involve historical data queries, requiring only dynamic window-based persistence of data for further offline analysis.

Mendes et al. [6] provides a generic framework for the evaluation of performance of event-based (Complex Event Processing) system. In contrast to the DEBS 2012 Grand Challenge authors focus only on the evaluation architecture without considering the evaluation problem itself.

5. EVENT PROCESSING CHALLENGE

The aim of the Event Processing Grand Challenge [3] (EPGC) is to identify a single, though broad, challenge that impacts society and which can be used by the community as a basis for measuring progress of the development of event-based systems. Event Processing Grand Challenge is composed of two parts: (1) a decentralized, global, Internet-like infrastructure, so-called Event Processing Fabric, built upon widely-accepted open standards; and (2) the design, development, deployment and management of life-changing, or society changing applications that utilize the Event Processing Fabric. The DEBS 2012 Grand Challenge contributes to the Event Processing Grand Challenge through its Internet of Things (IoT) orientation, which supports the realization of the Event Processing Fabric. Data produced in a manufacturing context is used for the early detection of problems that can be resolved in a timely way. Such an approach can also open the possibilities for proactive handling, i.e., the reactions not only to potential conflicts, but also the prediction of issues and new business opportunities.

6. SUMMARY

In this paper we present the DEBS 2012 Grand Challenge problem. We present the rationale for the DEBS 2012 Grand Challenge, originating in the manufacturing industry, along with a comprehensive set of data as well as KPI monitoring queries. The goal of this paper is to provide a long living reference for all willing to use the data and queries for benchmarking and evaluation of event based systems beyond the scope of the DEBS 2012 Grand Challenge. To that end the corresponding authors explicitly welcome comments or questions regarding both the data as well as queries specified in this paper.

7. ACKNOWLEDGMENTS

This work is partially sponsored by European Commission's Seventh Framework Program under grant agreement No. 257843 and grant agreement No. 260111 — projects SRT-15 (http://srt-15.eu) and KAP (http://kap-project.eu), respectively.

8. REFERENCES

[1] A. Alexandrov, D. Battré, S. Ewen, M. Heimel, F. Hueske, O. Kao, V. Markl, E. Nijkamp, and D. Warneke. Massively parallel data analysis with pacts on nephele. *PVLDB*, 3(2):1625–1628, 2010.

[2] A. Arasu, M. Cherniack, E. F. Galvez, D. Maier, A. Maskey, E. Ryvkina, M. Stonebraker, and R. Tibbetts. Linear road: A stream data management benchmark. In *VLDB*, pages 480–491. Morgan Kaufmann, 2004.

[3] M. K. Chandy, O. Etzion, and R. von Ammon. 10201 executive summary and manifesto – event processing. In *Event Processing*, number 10201 in Dagstuhl Seminar Proceedings. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, Dagstuhl, Germany, 2011.

[4] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, 2008.

[5] C. Jin and J. Carbonell. Predicate indexing for incremental multi-query optimization. In *17th International Symposium on Foundations of Intelligent Systems*, volume 4994, pages 339–350, Toronto, Canada, May 2008. Springer.

[6] M. R. N. Mendes, P. Bizarro, and P. Marques. A framework for performance evaluation of complex event processing systems. In *DEBS 2008: Proceedings of the Second International Conference on Distributed Event-Based Systems*, volume 332 of *ACM International Conference Proceeding Series*, pages 313–316, Rome, Italy, July 2008. ACM.