Demo: Dynamic Generation of Adaptive Real-time Dashboards for Continuous Data Stream Processing

Timo Michelsen, Marco Grawunder, Dennis Geesen and H.-Jürgen Appelrath

University of Oldenburg, Germany
`{timo.michelsen|marco.grawunder|dennis.geesen|appelrath}@uni-ol.de`

Abstract. Conventional database management systems are usually not capable to deal with continuous processing of potentially infinite data streams. Therefore, special data stream management systems and frameworks are developed. They use continuous queries which produce also streams as results, so that static visualization is not feasible, since the results are changing constantly. To handle this, we developed a dashboard concept, which we want to propose in this demonstration. A dashboard can be considered as an control or monitoring panel for real time data stream results. The user is free to define and configure individual dashboard parts. Each part is connected to a (user defined) continuous query, whose results are received and visualized in real-time. In this demonstration, we provide different data stream sources, continuous queries and dashboard parts. With them, the user can compose his own individual presentation of his processing results.

Key words: data stream, continuous queries, continuous visualization, framework, Odysseus

1 Introduction

Active data sources like sensors continuously producing data and send them to processing systems. The resulting data streams are potentially infinite and cannot be persistently stored at once. Conventional database management systems (DBMS) are not capable to deal with these type of sources. It is inevitable that each incoming data element has to be processed immediately. For this purpose, data stream management systems (DSMS) are developed. Instead of queries, which are executed once at a given time, DSMS use continuous queries. Continuous queries are installed into the DSMS and are running infinitely, producing streams of results [4].

Many DSMS are hard-coded, specified for one given task. Others provide entire frameworks with basic components to develop application-specific DSMS at a higher level. Odysseus is such a framework. It uses the OSGi Service Platform to provide basic components, which can be easily replaced or extended to meet specific requirements. For instance, Odysseus provides CQL, a SQL-like query language. A developer can implement an additional, more specific language, if needed. The graphical front-end of Odysseus uses the Eclipse Rich Client Platform (RCP). RCP is build on top of OSGi, providing the same extensibility as Odysseus itself [2].

2 Dashboards

The continuous processing of data streams prohibits a static visualization of query results. The results have to be presented in real-time. So the visualization component of a data stream management system has to update its visualization each time a new result is calculated. Additionally, the system has to deliver the user the tools to individualize the visualizations. For instance, if data streams come from special sources or the results of multiple queries should be shown in one picture. The most suitable solution allows changes even during the continuous data stream processing.

To meet these challenges, we developed a framework for visualizing multiple results in one huge visual surface and implemented it in Odysseus (see figure 1). Therefore, we split the visualization into smaller parts which can be handled

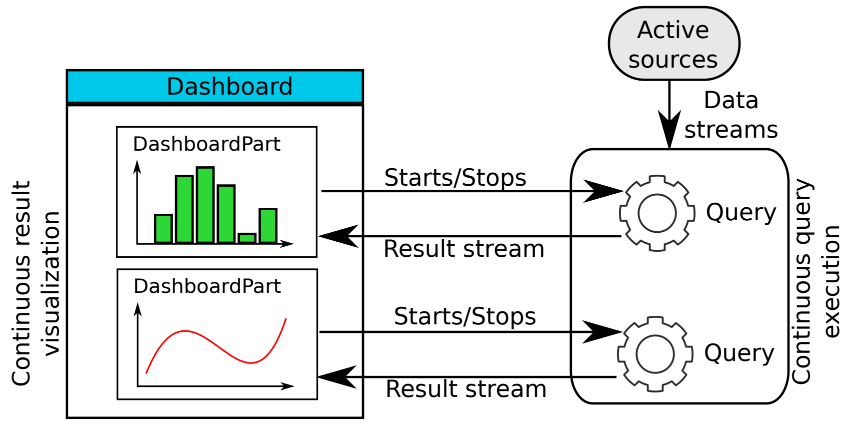


Fig. 1. Correlation between dashboards and continuous query execution in Odysseus.

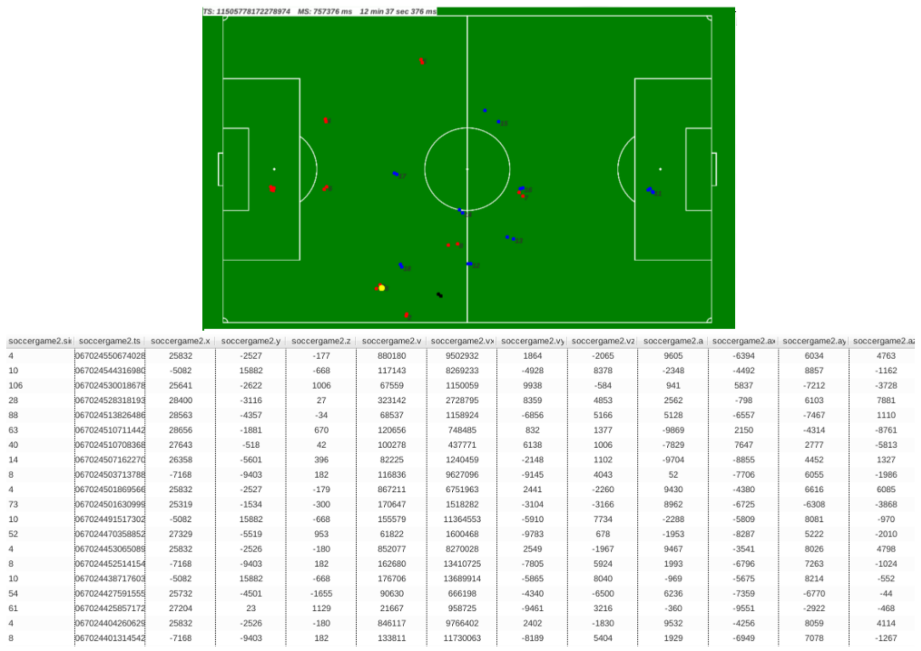
easier. If a user defines a continuous query, he can connect its (future) results to so-called *dashboard parts*. Dashboard parts are responsible for the execution of the queries: if one part is shown on screen, the underlying query will be installed, immediately started and the results are send to the presenting part. Since the queries are defined independently from the dashboard parts, the queries can be as complex as needed (with complex results and visualizations). Additionally, the dashboard part is highly configurable: First, the user can decide which type of part he wants to show (e. g. classical line diagrams or scatter plots). A developer can easily add additional (application specific) types, if needed. Second, each part can be individually configured in more detail (e. g. which exact attributes of one data stream should be visualized).

With the underlying RCP-Framework it is possible to save dashboard parts persistently and send them to other users who want to work with them. Next, a set of dashboard parts is composed to one complete visualization. In our concept, we call them *dashboards*, which usually occupy the whole screen. A dashboard is a central control and monitoring panel, which enables the user to see the processing data (and its results) graphically pleasant in real-time. A dashboard is responsible for the positions and sizes of the containing dashboard parts. Beginning with an empty dashboard, the user can include his predefined parts to it (e. g. by drag-and-drop), and move or resize each part inside the dashboard. At the same time, the results of the underlying continuous queries are already shown in real-time. Most likely, all parts are placed in a way, that each part is shown completely. In many situations, the user is only interested in showing his dashboard parts (and does not care about the exact positions and sizes). Therefore, we provide a function to layout all parts automatically (so-called *layouter*). In its simplest form, it positions all parts side-by-side in a grid-pattern. A developer can implement additional layouter for special requirements.

3 Demonstration contents

In this demonstration, we want to show our concept of the dashboards: the dynamic generation and real-time presentation of continuous data streams. Some predefined active data sources continuously send data stream elements to Odysseus. We propose a number of predefined queries which are running in the background. Of course the user can define his own continuous queries if he wants to. For generic data streams, we provide the classical set of diagrams as dashboard parts: line graphs, scatter plots, pie charts, etc. These are suitable for fast and easy visualizations. We also provide some specialized dashboard parts, which are implemented for one specific data stream and their corresponding queries. This shows the high adaptability of our concept. In this demo, we provide two data sources. First, the NREL dataset, containing data from about 32,000 wind power stations. Each station sends a data element each ten minutes which describes the actual amount of generated power, the speed of the wind and several other attributes [3]. Specialized dashboard parts (and dashboards) visualize these data in real-time to give an overview of the current status of the wind power stations. The second data source is a data set provided by the DEBS Grand Challenge 2013, containing position coordinates and moving data from a recorded soccer game [1].

Finally, the user can create an empty dashboard and drag some of our dashboard parts into it. During this process, the continuous queries connected in the background are installed and executed. The user can actually see the results in real-time. If the user removes one part, the corresponding query will be stopped and removed from Odysseus. Figure 2 shows a exemplary picture of a dashboard. It contains two different visualizations/dashboard parts: A specialized



TS: 11505778172278974 MS: 757376 ms 12 min 37 sec 376 ms

soccergame2.si	soccergame2.ts	soccergame2.x	soccergame2.y	soccergame2.z	soccergame2.v	soccergame2.vx	soccergame2.vy	soccergame2.vz	soccergame2.a	soccergame2.ax	soccergame2.ay	soccergame2.az
4	067024550674028	25832	-2527	-177	880180	9502932	1864	-2065	9605	-6394	6034	4763
10	067024544316980	-5082	15882	-668	117143	8269233	-4928	8378	-2348	-4492	8857	-1162
106	067024530018678	25641	-2622	1006	67559	1150059	9938	-584	941	5837	-7212	-3728
28	067024528318193	28400	-3116	27	323142	2728795	8359	4853	2562	-798	6103	7881
88	067024513826486	28563	-4357	-34	68537	1158924	-6856	5166	5128	-6557	-7467	1110
63	067024510711442	28656	-1881	670	120656	748485	832	1377	-9869	2150	-4314	-8761
40	067024510708368	27643	-518	42	100278	437771	6138	1006	-7829	7647	2777	-5813
14	067024507162270	26358	-5601	396	82225	1240459	-2148	1102	-9704	-8855	4452	1327
8	067024503713788	-7168	-9403	182	116836	9627096	-9145	4043	52	-7706	6055	-1986
4	067024501869566	25832	-2527	-179	867211	6751963	2441	-2260	9430	-4380	6616	6085
73	067024501630999	25319	-1534	-300	170647	1518282	-3104	-3166	8962	-6725	-6308	-3868
10	067024491517302	-5082	15882	-668	155579	11364553	-5910	7734	-2288	-5809	8081	-970
52	067024470358852	27329	-5519	953	61822	1600468	-9783	678	-1953	-8287	5222	-2010
4	067024453065089	25832	-2526	-180	852077	8270028	2549	-1967	9467	-3541	8026	4798
8	067024452514154	-7168	-9403	182	162680	13410725	-7805	5924	1993	-6796	7263	-1024
10	067024438717603	-5082	15882	-668	176706	13689914	-5865	8040	-969	-5675	8214	-552
54	067024427591555	25732	-4501	-1655	90630	666198	-4340	-6500	6236	-7359	-6770	-44
61	067024425857172	27204	23	1129	21667	958725	-9461	3216	-360	-9551	-2922	-468
4	067024404260629	25832	-2526	-180	846117	9766402	2402	-1830	9532	-4256	8059	4114
8	067024401314542	-7168	-9403	182	133811	11730063	-8189	5404	1929	-6949	7078	-1267

Fig. 2. Screenshot of a dashboard containing two dashboard parts for a soccer game.

part, showing the current position of each player, and a classical table, showing the raw data (e. g. useful for debugging purposes). These parts are rendered during the data stream processing in real-time, which is indicated in a static picture here. It shows, how our concept can be used to build a dynamic, component based real-time visualization of streaming data.

References

1. Debs grand challenge 2013. `http://www.orgs.ttu.edu/debs2013/index.php?goto=cfchallengedetails` (Last visit: 06/12/13), 2013.
2. H.-J. Appelrath, D. Geesen, M. Grawunder, T. Michelsen, and D. Nicklas. Odysseus: a highly customizable framework for creating efficient event stream management systems. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, DEBS '12, pages 367–368, New York, NY, USA, 2012. ACM.
3. S. Hammond. Challenges and opportunities in renewable energy and energy efficiency. In *Proceedings of the international conference on Supercomputing*, ICS '11, pages 151–151, New York, NY, USA, 2011. ACM.
4. J. Krämer. Continuous queries over data streams - semantics and implementation. *PhD thesis, University of Marburg*, 2007.