The OSIRIS-SE (Stream-Enabled) Infrastructure for Reliable Data Stream Management on Mobile Devices*

Gert Brettlecker, Heiko Schuldt
University of Basel, Department of Computer Science, Basel, Switzerland
gert.brettlecker, heiko.schuldt@unibas.ch

ABSTRACT

The proliferation of software and hardware sensors which continuously create large amounts of data has significantly facilitated novel types of applications such as healthcare telemonitoring or roadside traffic management. All these applications demand new mechanisms for online processing and analysis of relevant data coming from multiple data streams. Especially telemonitoring applications in healthcare require a high degree of reliability and must be able to be deployed in a distributed environment. We present OSIRIS-SE, an information management infrastructure for reliable data stream management in a failure-prone distributed setting including resource-limited mobile devices. OSIRIS-SE supports the combination of different data stream operators into stream processes and offers efficient coordinated operator checkpointing for the execution of these stream processes. In order to support mobile devices, OSIRIS-SE is able to deal with multiple failures, offers fine-grained reliability at operator level, and supports decentralized stream process orchestration in a peer-to-peer fashion. Moreover, OSIRIS-SE is fully implemented in Java and thus can be run on different platforms. We show the reliable execution of stream processes in a health monitoring application including a wearable ECG sensor, a Bluetooth enabled blood pressure sensor, and a web cam as data sources. Operators are hosted at mobile devices (PDAs, smart phones) of a patient and at a laptop computer which also acts as base station. An important feature of OSIRIS-SE is to show that sensor data can losslessly be processed by seamlessly migrating stream processing to other devices in the network even in case of multiple failures.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; J.3 [**Computer Applications**]: Life and Medical Sciences

General Terms

Management, Reliability, Experimentation

*The work presented in this paper has been done in the DELOS Network of Excellence on Digital Libraries, funded by the Swiss State Secretariat for Education and Research (SER) under contract No. SBF 03.0546-3.

SIGMOD'07, June 11–14, 2007, Beijing, China.
ACM 978-1-59593-686-8/07/0006.

Keywords

Data Stream Management, Healthcare, Telemonitoring, Checkpointing, Information Management Infrastructure

1. INTRODUCTION

An important new area of data stream management (DSM) applications are applications for telemonitoring in healthcare. Due to the aging society, chronic ailments such as cardiovascular diseases, hypertension, and diabetes affect a significant number of the population [1]. Telemonitoring applications enable healthcare institutions to take care of their patients while they are out of hospital, which is especially useful for managing various chronic diseases and is also more comfortable for the patients. In addition, also the effects of treatments under real-life conditions can be remotely measured. Emerging telemonitoring applications utilize new sensor technologies, wireless communication standards, powerful mobile devices, and wearable computers. Recent trends in ubiquitous and pervasive computing strongly support this novel applications.

We present our novel information management infrastructure OSIRIS-SE which fully takes advantage of these new developments. OSIRIS-SE supports distributed processes and DSM with a high degree of reliability. This infrastructure is particulary focused on supporting telemonitoring applications in healthcare, but not limited to this domain. Similar issues arise in other areas as well, e.g., environmental monitoring or traffic control systems where continuous sensor data has to be processed in a reliable way.

We pursue the following telemonitoring application scenario: Fred, aged 71, has already been suffering from congestive heart failure for 1.5 years. Due to this fact, he has a high risk of heart failure. Therefore, Fred's caregiver decides to equip him with a wearable telemonitoring system consisting of a wearable ECG sensor and a Bluetooth enabled blood pressure sensor. Future systems will also oxygen saturation sensors. The wearable sensors are connected to distributed OSIRIS-SE infrastructure including PDAs and smartphones carried by patients and physicians, base stations at patient's homes and servers at the caregiver's site. In order to interpret Fred's vital signs appropriate additional context information is needed, e.g., ECG signals vary if Fred is running or sleeping. Inbuilt sensors in his smart home environment (e.g., a web cam) is providing this additional context information. Moreover, the caregiver is able to define individual data stream processing for Fred in a graphical "boxes and arrows" approach by combining basic building blocks with a graphical process design tool.

Fred's vital sensors and context information is processed by the OSIRIS-SE infrastructure on several nodes in a distributed setup. On these nodes, the data accumulated is analyzed, extracted and relevant information is forwarded to the care provider in charge. The increased number of components, devices, and platforms in this scenario leads to an increased failure probability. Reliability is of utmost importance in telemonitoring applications. OSIRIS-SE provides a flexible platform for different kinds of monitoring applications and guarantees a high level of reliability. In case of relevant changes of Fred's health condition, his physician in charge is automatically informed, and she is able to retrieve all medically relevant data. Also, critical events (e.g., heart attack) can be detected by analyzing the data streams. These require immediate intervention, and reliable processes for calling the emergency service are invoked.

2. OSIRIS-SE INFRASTRUCTURE

OSIRIS-SE [2, 3, 4] is a novel information management infrastructure for reliable and efficient DSM and process management. OSIRIS-SE is based on the *Open Service Infrastructure for Reliable and Integrated process Support (OSIRIS)* [6, 7], which is a prototype of a hyperdatabase [5] infrastructure, that has been developed at ETH Zurich and extended at UMIT and University of Basel. OSIRIS-SE (OSIRIS-Stream Enabled) extends OSIRIS by adding DSM functionality. The OSIRIS-SE architectures consists of two parts. Firstly, a local software layer which is running on each participating node, called *OSIRIS-Layer*. Secondly, *core services* are repositories for metadata, which offer the replication of all locally needed information for distributed data stream processing and process execution.

OSIRIS-SE supports the execution of continuous queries over data streams, which we call *stream processes*. A stream process is a well-defined set of logically linked *data stream operators* which continuously process input data streams. In doing so, they produce results (output data streams) and might have side effects. Data stream operators, short *operators*, keep and aggregate an *operator state* during their life-time of processing. The OSIRIS-SE infrastructure offers the ability to activate, run, and stop stream processes. Activating and stopping stream processes is performed by the execution of traditional (non-stream) processes which individually activate or stop the operator instances needed and establish the connecting data streams between them properly. During the running phase of the stream process the infrastructure is controlling operator execution and data stream flow. In case of failures, the OSIRIS-SE infrastructure is able to migrate the stateful data stream operators to an unaffected node and redirect data streams in order to continue data stream processing without losing results. The emphasis of failure handling in OSIRIS-SE is offering lossless data stream processing, even in case of multiple failures while not degrading performance. According to the application scenario, we assume failures are frequent and may happen in all phases of the stream process' life-time. Additionally, loss of data is not tolerable because this may have severe consequences for the monitored patients.

3. RELIABILITY OF OSIRIS-SE

OSIRIS-SE is able to handle two types of failures. Firstly, *transparent failure handling* is completely done by the infrastructure without any manual intervention or interaction with the application. Secondly, if transparent failure handling is not possible, OSIRIS-SE offers *application-defined failure handling*.

Application-defined failure handling is available in two flavors. Firstly, *alternative processing branches* are defined in the stream process specification, for example to continue data stream processing even in the case an operator instance is not available. This can be done by executing an alternative processing branch which is bypassing the missing operator. Secondly, *invocation of traditional processes* may handle failure situations where the proper execution of the stream process is not possible. These traditional processes are defined by the process designer (e.g., the caregiver) and they describe how to cope with the failure. For example, patient Fred, who is leaving the house, can be informed by an alarm process to return in connection range of his monitoring system. The alarm process may send an alarm SMS to Fred by using his cellular phone.

The first-class solution in OSIRIS-SE is to handle failures transparently. The demonstration described in Chapter 4 will emphasize on transparent failure handling. Transparent failures are further distinguished between *temporary* (handled within a timeout period) and *permanent* (these failures cannot be handled within the timeout).

All kinds of temporary failures, e.g., a temporary network disconnection (which results in loss of messages) or a temporary failure of a service provider which recovers within the given time, are compensated by buffering. After recovery, the buffered data stream elements are resent. In order to prevent from long delays and huge buffer sizes, a temporary failure becomes a permanent failure after exceeding a timeout. Permanent failures require to migrate the affected operator instance with its aggregated operator state from the affected provider to another suitable provider. This task is called *operator migration*. In particular, ensuring consistency between operator states even in case of failures is a crucial constraint for seamless continuation of data stream processing. The demonstration shows the handling of failure scenarios by operator migration. Since the provider that hosted the operator is no longer reachable, the infrastructure needs a recent backup of the operator state and all queued (not yet processed) stream elements to allow for correct operator migration. If no operator migration is possible user-defined failure handling is applied.

An important prerequisite of operator migration is to ensure the availability of a recent operator state backup on an other node (the *backup node*) in the OSIRIS-SE infrastructure. If multiple backup nodes are used OSIRIS-SE can also prevent from backup node failures. In order to achieve consistent recovery, which means that operator migration will not lead to loss of data, OSIRIS-SE applies an efficient operator checkpoint strategy. Main focus of this strategy is to reduce the necessary network overhead of checkpointing. For mobile devices, network overhead is usually expensive due to low bandwidth and high energy consumption. The main idea of our reliability approach is firstly to offer fine-grained reliability at operator level. Secondly, in order to reduce the size of checkpoint messages, checkpoints are applied by coordination between via data streams connected operator instances. For details on our coordinated checkpoint algorithm see [4].

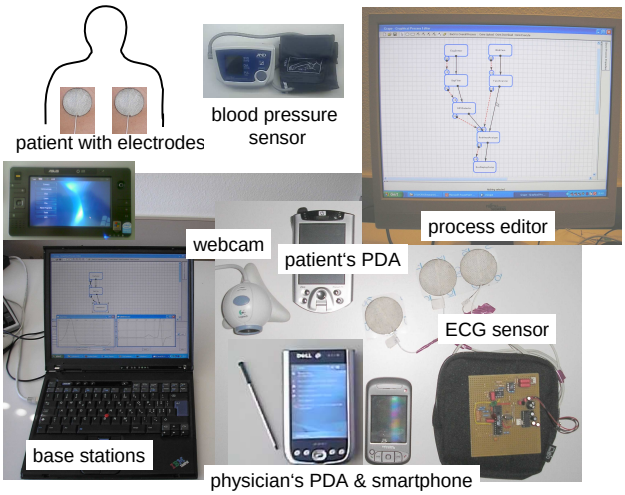

Figure 1: Demo Setup

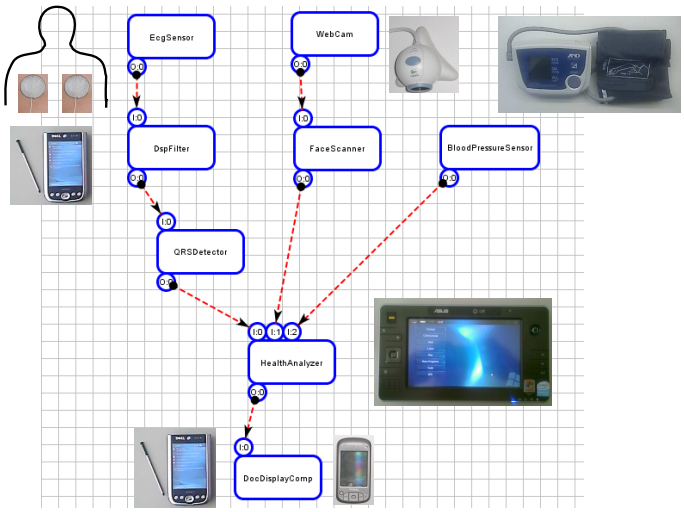

Figure 2: Demo Process

4. DEMONSTRATION

The demonstration executes the presented health monitoring scenario (see Chapter 1) based on our OSIRIS-SE infrastructure. Figure 1 illustrates the devices included in the demo setup. The smart ECG sensor is continuously acquiring the current heart activity (as ECG signal) of the patient at a rate of 100 samples per second and sending them to an OSIRIS-SE enabled PDA. Additionally, a web cam attached to the OSIRIS-SE enabled laptop is used as a second sensor source producing an image data stream. Moreover, regularly taken measurements with the blood pressure sensor device are also forwarded to an OSIRIS-SE enabled PDA. Laptops and ultra-mobile PCs are used for the base station at the patient's home and the caregiver's server. Each of the different OSIRIS-SE enabled devices has the same OSIRIS-SE infrastructure software running in a Java virtual machine, which nicely demonstrates the platform independence of this approach. The stream process used by this application is illustrated in Fig. 2. This stream process will acquire the ECG signal by the *ECGSensor* operator. Noise is removed from this data stream by applying the *DSPFilter* operator. Medical relevant information about the peaks of the heart activity (so called QRS complexes) is derived by the *QRSDetector* operator. The medical relevant information of heart activity and blood pressure is combined with additional context information derived from the web cam data stream. Therefore, the *WebCam* operator is acquiring images of a web cam observing the patient, which are evaluated by the *FaceScanner* operator. This operator detects faces in the image stream and calculates the average red value of the face area. The *HealthAnalyzer* operator generates alarm events whenever the red value of the face, the heart activity or the blood pressure received from the *BloodPressureSensor* operator exceed thresholds given by the caregiver. These alarm events are sent to the *DocDisplayComp* operator running at the smartphone of the physician in charge. This example –which can seamlessly be extended by other sensor data sources– nicely illustrates the combination of physiological data streams with contextual data streams. We show that even if the smartphone fails because of battery problems or wireless coverage, the OSIRIS-SE infrastructure is able to seamlessly migrate the *DocDisplayComp* operator to another PDA device without loss of data. This means, each alarm event that has not been shown to the doctor before the failure is shown after migration at the new device. Additionally, we present that even multiple failure situations, e.g., also the patient's base station fails at the same time, are handled by OSISIS-SE. Furthermore, we show how to easily define and adapt stream processes with our graphical process editor tool O'GRAPE [8].

5. REFERENCES

[1] American Heart Association. *2002 Heart and Stroke Statistical Update*, 2001.

[2] G. Brettlecker, H. Schuldt, and R. Schatz. Hyperdatabases for Peer–to–Peer Data Stream Processing. In *Proc. of ICWS Conf.*, pages 358–366, San Diego, CA, USA, 2004.

[3] G. Brettlecker, H. Schuldt, and H.-J. Schek. Towards Reliable Data Stream Processing with OSIRIS-SE. In *Proc. of BTW Conf.*, pages 405–414, Karlsruhe, Germany, March 2005.

[4] G. Brettlecker, H. Schuldt, and H.-J. Schek. Efficient and coordinated checkpointing for reliable distributed data stream management. In *Proc. of ADBIS Conf.*, pages 296–312, Thessaloniki, Greece, September 2006.

[5] H.-J. Schek, H. Schuldt, and R. Weber. Hyperdatabases – Infrastructure for the Information Space. In *Proc. of VDB Conf.*, pages 1–15, Brisbane, Australia, 2002.

[6] C. Schuler, R. Weber, H. Schuldt, and H.-J. Schek. Peer–to–Peer Process Execution with OSIRIS. In *Proc. of ICSOC Conf.*, pages 483–498, Trento, Italy, 2003.

[7] C. Schuler, R. Weber, H. Schuldt, and H.-J. Schek. Scalable Peer–to–Peer Process Management – The OSIRIS Approach. In *Proc. of ICWS Conf.*, pages 26–34, San Diego, CA, USA, 2004.

[8] R. Weber, C. Schuler, P. Neukomm, H. Schuldt, and H.-J. Schek. Web Service Composition with OGrape and OSIRIS. In *Proc. of VLDB Conf.*, Berlin, Germany, 2003.