Dautov *et al. J Big Data* (2019) 6:19
https://doi.org/10.1186/s40537-019-0183-6

Journal of Big Data

RESEARCH Open Access

Hierarchical data fusion for Smart Healthcare

Rustem Dautov[1*†], Salvatore Distefano[1,2†] and Rajkumaar Buyya[3†]

*Correspondence:
rdautov@it.kfu.ru
†Rustem Dautov, Salvatore Distefano and Rajkumaar Buyya equally contributed to the paper
[1] Kazan Federal University, Kazan, Russia
Full list of author information is available at the end of the article

Abstract

The Internet of Things (IoT) facilitates creation of smart spaces by converting existing environments into sensor-rich data-centric cyber-physical systems with an increasing degree of automation, giving rise to Industry 4.0. When adopted in commercial/industrial contexts, this trend is revolutionising many aspects of our everyday life, including the way people access and receive healthcare services. As we move towards Healthcare Industry 4.0, the underlying IoT systems of Smart Healthcare spaces are growing in size and complexity, making it important to ensure that extreme amounts of collected data are properly processed to provide valuable insights and decisions according to requirements in place. This paper focuses on the Smart Healthcare domain and addresses the issue of data fusion in the context of IoT networks, consisting of edge devices, network and communications units, and Cloud platforms. We propose a distributed hierarchical data fusion architecture, in which different data sources are combined at each level of the IoT taxonomy to produce timely and accurate results. This way, mission-critical decisions, as demonstrated by the presented Smart Healthcare scenario, are taken with minimum time delay, as soon as necessary information is generated and collected. The proposed approach was implemented using the Complex Event Processing technology, which natively supports the hierarchical processing model and specifically focuses on handling streaming data 'on the fly'—a key requirement for storage-limited IoT devices and time-critical application domains. Initial experiments demonstrate that the proposed approach enables fine-grained decision taking at different data fusion levels and, as a result, improves the overall performance and reaction time of public healthcare services, thus promoting the adoption of the IoT technologies in Healthcare Industry 4.0.

Keywords: Smart Healthcare, Industry 4.0, Data fusion, Complex Event Processing, Distributed architecture, Internet of Things, Edge computing, Cloud computing

Introduction

Healthcare is one of the many domains, continuously improved by the pervasive penetration of IoT technologies, which are used to support core functions of healthcare institutions. This way, traditional hospitals are converted into next-generation smart digital environments extensively making use of interconnected sensor systems and (Big) data collection/processing techniques. From this perspective, Smart Healthcare can be seen as a complex ecosystem of smart spaces (e.g. hospital rooms, ambulances, pharmacies, etc.), supported by a powerful infrastructure stack (including edge devices and sensors, wired and wireless networks, Cloud platforms, etc.) and driven by innovative business models and legislation enabling the Healthcare Industry 4.0.

The foundation of Smart Healthcare is built on intelligent, low-power, wirelessly connected medical devices. These devices continuously measure, process, capture and protect the biometric data collected by sensors, including body position, weight and movements, sleep quality, blood pressure, blood oxygen saturation, body temperature, heart rhythm and rate, blood oxygen, fatigue levels, respiration rate, etc. [3, 33]. This gave rise to the Internet of Medical Things (IoMT) [12, 20]. By embedding sensors into internal spaces and hospital equipment, such as beds and wheelchairs thus becoming "medical things", hospital staff are given an opportunity to receive and view valuable biometric information on their computers and/or mobile devices remotely via a wireless connection, while performing their daily duties. This way, doctors are able to take immediate decisions, thus potentially saving people's lives. In most cases, however, the collected information, albeit timely and precise, is only intended to be displayed to the hospital personnel, who remain responsible for taking final decisions based on this received information. From this perspective, finding potential correlations between various human body indicators—i.e. 'fusing' data from multiple sources—is undertaken manually by a doctor, who applies his/her knowledge and experience to detect/prevent a potentially critical situation.

Indeed, often a single source of information is not sufficient to take reliable and justified decisions. Therefore, there is a need for a combination of several various data sources, be it streaming or static data. Establishing a diagnosis by a doctor is a representative example in this context—i.e. a healthcare professional can only diagnose an illness/disease based on several examinations (e.g. blood and urine tests, blood pressure, blood temperature, heart rate, etc.), not just one. Admittedly, detecting a deviation in a single health indicator (e.g. body temperature) is not enough to diagnose an illness, as there may be plenty of reasons to an increased temperature. To this end, doctors typically base their decisions on several examinations, and by performing manual data fusion over these several sources (typically, the more the better), are able to minimise the uncertainty and, eventually, come to a precise, valid diagnosis. Same principles of combining multiple sources also apply to automated data fusion, especially in the Smart Healthcare contexts where (huge) streams of (Big) data, coming from different spatially-distributed locations, have to be 'fused', often under real-time constraints.

Furthermore, there may be more complex cases, when a single doctor does not have sufficient information and/or experience to establish a diagnosis. In these circumstances, to diagnose a problem, a medical board of healthcare professionals with different specialisations needs to discuss and correlate their individual findings, after which the head of the board is able to take the final decision, based on the inputs provided by individual board members. From this perspective, this medical board resembles a hierarchical organisation, where lower-level members provide information to a higher-level decision maker.

However, with these promising opportunities come emerging challenges as to how to process and manage avalanches of data, continuously generated by millions of sensing devices. A particularly pressing concern in these circumstances is to enable smooth integration of multiple data sets into a consistent, accurate, and useful representation—that is, to perform *data fusion* [29]. More formally, the goal of data fusion is to combine

relevant information from multiple data sources into a single one to provide a more accurate description than any of the individual data sources alone and minimise uncertainty [19].

The existing approaches to enable IoT data fusion (including the Smart Healthcare domain) and processing have primarily adopted a *Cloud-centric* model, where raw data collected by edge devices are pushed to a Cloud that is seen as the primary processing location [28]. Such a *vertical* off-loading model, however, tends to neglect or undervalue (growing) computational resources of edge devices to support data analytics and processing, including data fusion [10]. As a result, some relatively simple data fusion tasks instead of being accomplished immediately on the spot, are pushed to a Cloud server through a potentially congested public network, thereby suffering from several issues. First, by not processing simpler tasks locally, the overall reaction time considerably increases due to network latency and limited bandwidth. Second, pushing potentially sensitive data through a public network is associated with an increased security risk, whereas applying additional data encryption techniques as a way of addressing this challenge introduces unnecessary computational and network workload. Third, by not utilising computational resources of edge devices, this model requires the Cloud to cope with an increased (Big Data) workload, which might also reflect in the overall cost of renting and running the Cloud service.

Accordingly, taking the above considerations as a reference, this paper presents a hierarchical automated data fusion architecture for Smart Healthcare ecosystems, where individual elements perform data fusion over multiple data sources with respect to their background knowledge and processing capabilities. This way, lower-level elements carry out relatively limited data fusion and transfer aggregated information to higher-level elements, which, in their turn, after fusing the received information, may transfer newly aggregated information further up to the upper elements in the hierarchy. The proposed architecture is implemented using the Complex Event Processing (CEP) technology, which aims at detecting complex event patterns in a stream of atomic events, and represents a potential way of implementing data fusion in distributed IoT systems [8, 9], such as smart hospitals. Deployed as close to the source of sensor data, it aims at utilising local processing capabilities wherever possible, or off-load tasks to Fog/Cloud computing otherwise—thereby paving the way for a multi-layered, hierarchical data fusion approach. As it will be further discussed and demonstrated by a healthcare use case scenario, this approach enables fine-grained decision taking at different data fusion levels, and, as a result, improves the overall performance and reaction time. Thus, the contribution of this paper is threefold: (i) a fine-grained hierarchical data fusion approach for timely decision taking in digital healthcare, (ii) a prototype implementation of the proposed system using the CEP technology, and (iii) evaluation and benchmarking of the proof-of-concept implementation against an existing centralised approach.

Accordingly, the rest of the paper is organised as follows. In "Background" section, IoT ecosystems and data fusion approaches are studied to identify main processing architectures and data fusion patterns. Next, in "Methods" section, these patterns and IoT processing models are mapped into a three-tier hierarchical solution for the Smart Healthcare domain. This section also provides an example of applying such a solution to Smart Healthcare services and demonstrates the feasibility of the proposed hierarchical

data fusion approach with a proof-of-concept implementation. "Results and discussion" section describes experimental evaluation of the proposed approach and discusses the results. "Conclusion" section concludes the paper with some final remarks and outlines directions for future work.

Background

The innovative Industry 4.0 is supported by several technological pillars, including Cloud computing, Big Data analytics, Cyber Physical Systems and the IoT [24]. Supported by the recent advancements in the networking, mobile and embedded technologies, they serve to enable highly automated data-centric service-oriented business processes by ubiquitous insertion of smart sensors for data acquisition. As a result, numerous devices, equipped with a wide range of sensing resources, as well as relatively advanced computing, storage and communication capabilities (thus earning the *smart* attribute), are ubiquitously present in people's life aiming to improve it. Such sensor-enabled smart objects are Internet-connected, thereby creating a global network of remotely accessible data sources, ready to be discovered and integrated into complex cyber-physical systems.

As a result, this new paradigm paved the way for an increased level of industrial automation and previously unknown business models. Moreover, the Industry 4.0 principles are also revolutionising other public and social domains (not necessarily related to production and manufacturing), including the healthcare system, thus giving rise to the alike concepts of *Health 4.0*, *Healthcare Industry 4.0*, and *Smart Healthcare*. The aim of these emerging trends "is to allow for progressive virtualization in order to enable the personalization of health and care next to real time for patients, professionals and formal and informal carers" [36]. Although typically associated with the concept of a *smart hospital* (which still remain the fulcrum of a healthcare system), Healthcare Industry 4.0 and Smart Healthcare go far beyond the boundaries of a single hospital spanning across multiple healthcare institutions, as well as public administrations, offices and government and public health services. In this light, it is important to ensure that Health 4.0 is realised not via vertical non-sustainable solutions aiming to 'smarten' individual, isolated clinics and hospitals, but rather via an all-encompassing Smart Healthcare solution, able to cover much wider scenarios, involving multiple organisations and stakeholders. From this perspective, further implementation of the (Smart) Healthcare Industry 4.0 vision requires a global approach and depends on a technological convergence among various ICT domains, including IoT, Big Data analytics, and Cloud Computing.

At present, a common solution adopted for data management and processing in the context of Smart Healthcare is to offload related tasks to remote servers, typically located in datacenters and Cloud platforms, which collect, store and process data, thus pushing for the interaction between the IoT and the Cloud paradigms [11]. In this regard, Smart Healthcare has primarily adopted a 'vertical' offloading paradigm, in which raw sensor data are collected by edge devices and transferred over the network to a central processing location via several network links. This inevitably implies that raw biometric data are sent out immediately upon generation, thus putting strict dependency on the underlying network bandwidth. As a consequence, this requirement may often appear not affordable or unsustainable due to restrictions of the underlying (wireless) networks and related (3G/4G) providers, which

Table 1 IoT resources categorisation

Category	Property					
	Sensing actuation	Internet facilities	Processing	Memory, storage	Advanced connectivity	Service, resource provisioning
Device	X	–	–	*	–	–
Mote—basic gateway	*	X	–	*	–	–
Smart object—smart gateway	*	X	X	*	–	–
Communication and processing unit	–	X	X	X	X	–
Datacenter	–	X	X	X	*	X

impacts the overall processing latency. Despite this limitation, (resource-constrained) edge devices are still not expected to perform data processing themselves, but rather have to push collected data through the network topology, albeit they may already have enough processing, storage and computing resources on-board, which can be exploited in sensor fusion (e.g. filtering, pre-processing, local analytics) with benefits in terms of performance and network latency.

Moreover, network communication and processing units, such as mobile edge cloud (MEC) servers and cloudlets [37], are usually powerful enough to support such computations. This pattern was proposed by the Edge/Fog computing paradigm, aiming at pushing intelligence to the edge of the network topology [17]. In Edge/Fog computing, network devices, IoT gateways, switches, routers, servers, and cloudlets, are widely used to support computational tasks, incoming from edge nodes. This complements the traditional Cloud offloading and overcomes bandwidth constraints, mitigating network latency and delays. Therefore, a solution able to minimise costs and latency, while taking into account edge devices' resource constraints, is possible by exploiting and combining local resources with those provided by the network (i.e. Edge/Fog computing) and remote (i.e. Cloud computing) processing nodes. To this end, relevant concepts and insights on infrastructure and software/data management solutions are discussed below to provide a baseline for an effective converging Smart Healthcare solution.

Infrastructure: IoT, Edge, Fog and Cloud

From an infrastructure perspective, a Smart Healthcare ecosystem is composed of multiple interconnected edge devices, network nodes, and (virtual/physical) servers. Connected into an IoT network, these elements differ in their capabilities, and can be classified into four categories, as summarised in Table 1. In this taxonomy, six features have been identified based on the resources provided by a specific node. The *Sensing/Actuation* parameter refers to the presence of sensors and actuators that provide corresponding capabilities. Then, basic network connectivity features are referred to as *Internet Facilities*. Other features are related to *Processing* and *Memory/Storage* capabilities. *Advanced Connectivity* features refer to the forwarding and/or routing functionality, as well as support for software-defined and virtualised networking. Finally, the *Service/Resource Provisioning* capabilities allow to provide resources and services to third parties on demand in a service-oriented fashion.

Table 2 Data fusion taxonomy

Category	Taxonomy		
	Low level	**Middle level**	**High level**
Processing stage [22]	Low	Intermediate	High
Levels of abstraction [30]	Low	Medium	High
Data granularity [32]	Raw data processing	Feature level	Decision level
Operation domains [39]	Temporal	Spatial	Temporal and spatial
Semantics [16]	Knowledge base construction	Pattern matching	Inference
Source relationships [13]	Complementary	Redundant	Cooperative
User requirements [39]	Local/single node	Region	Global/overall network

Based on this set of features, a *Device* can be considered as a simple node with sensing and actuation capabilities, but without any network facilities, thus requiring to be physically connected to an access point/gateway to send acquired data. It can have some (limited) storage capabilities to aggregate several measurements into batches before sending for minimising energy consumption (which is important for battery-powered devices). Examples are any kind of biometric sensors and actuators, as well as multiple sensor boards/shields.

A *mote-basic gateway* is a node responsible for transferring data from edge devices to the Internet (i.e. collecting sensor values), or vice versa (i.e. sending actuating commands). It can be equipped with own sensors and actuators, or these can be connected through a specific (wired) interface to its pins (e.g. GPIO pins). Storage facilities can be present in a gateway, which typically has limited or no processing capabilities on-board. Examples are boards, only equipped with a micro-controller, such as Arduino Uno and similar.

A *smart object* and a *smart gateway* are more powerful units, equipped with processing facilities, thus allowing on-board (pre-)processing of probed data, coming from own or pin-connected sensors. Examples are smart boards and single-board computers with microprocessors, such as Raspberry Pi and Arduino Yun.

Communication and processing units are network nodes and servers with advanced connectivity and processing capabilities, but usually without any sensing/actuation facilities. They are able to support data processing tasks (filtering, analytics, data fusion, etc.) off-loaded by smart objects and gateways according to the Fog computing principles. Examples are routers, MEC servers, base stations, and cloudlets.

Datacenters provide processing, storage and networking resources to support IoT healthcare applications, possibly through a Cloud-based provisioning model.

According to this categorisation, data processing in the IoT-Cloud ecosystems can be performed at three different levels, as summarised in Table 2: on-board (i.e. 'Mist' or Edge computing), on communication and processing servers and cloudlets (i.e. Fog computing), and/or in remote datacenters (i.e. Cloud computing).

Software: data fusion

Data fusion is a broad 'umbrella' term for several techniques and approaches, aiming to combine data, information and knowledge with a goal to improve data quality,

reduce uncertainty, extract/mine advanced (emerging) features, provide statistics/analytics, and so on. Several different data fusion patterns have been identified based on multiple parameters. From a data processing perspective, the primary broad classification distinguishes between standalone and distributed data fusion approaches. As demonstrated by the relevant literature, real case studies, and applications, the latter are widely adopted for IoT data fusion, especially in the presence of the Big Data challenges. Other relevant data fusion taxonomies and classifications [1] are briefly summarised in Table 2.

By looking at the taxonomy in Table 2, it is possible to identify a clear multi-layered architectural pattern, where individual levels build one on top of the other in a hierarchical manner. More specifically, a three-level model prevails in the surveyed taxonomies (albeit with a possibility to be extended to a greater number of layers). Accordingly, the conceptual three-level data fusion model can be summarised as follows:

I. *Low level* includes data fusion features usually applied to raw data coming from sources, implementing a first stage of processing or at a low level of abstraction, performing local operations such as those in a temporal domain aiming at knowledge base construction or cooperating with other nodes on complementary activities.
II. *Middle level* includes features at a higher level, which could be performed on preprocessed information, i.e. the results of the previous processing stage, local computation or level of abstraction, for example to obtain spatial domain parameter estimations on a given area, or implementing feature extraction, pattern matching or redundant computations.
III. *High level* implements the last processing stage at the highest level of abstraction on a global domain, performing inference or complex reasoning and decision making on data, coming from the lower layers or implementing temporal–spatial fusion by, for example, exploiting cooperative patterns.

Speaking of specific technologies aligned with the presented hierarchical data processing model, a potentially promising data fusion approach, that can be potentially applied in the Smart Healthcare context, is Complex Event Processing (CEP) [27]. As opposed to the traditional Stream Processing, CEP goes beyond simple data querying/transformation and aims to detect complex event patterns, themselves consisting of simpler atomic events, within a data stream. Accordingly, from CEP's point of view, constantly arriving tuples can be seen as notifications of events happening in the external world—e.g. an increase in body temperature or a sudden drop of blood pressure. The focus of this perspective is on detecting occurrences of particular patterns of lower-level events that represent higher-level events. A standing CEP query fetches results if and only if a corresponding pattern of lower-level events is detected. For example, a common task addressed by CEP systems is detection of situation patterns, where one atomic event happened strictly after another. To achieve this, CEP systems rely on event timestamps; they extend existing query languages with sequential operators, which allow specifying the chronological order of events or, simply put, whether one event happens before or after another in time.

CEP's capabilities to enable data fusion over incoming streams have been utilised in scenarios, where data are continuously pushed from a distributed network to be dynamically analysed as soon as they arrive. More specifically, a number of network intrusion detection systems (IDSs) [5, 14, 23] were implemented using CEP. These IDSs aim to collect and correlate all network events in one place to detect critical situations, which are represented as pre-defined CEP rules. There is also evidence of utilising CEP techniques in the context of IoT systems. Similarly, the motivation is to enable run-time monitoring and data fusion in the context of distributed IoT networks [6, 15, 18, 38]. Admittedly, similar challenges in terms of timely detection and reaction to collected sensor measurements are present in the Smart Healthcare domain.

There are two main aspects, however, which are not addressed by the literature yet. First, existing approaches tend to implement CEP only at the highest level of Cloud computing, thus neglecting the possibility of introducing intermediate data fusion at the levels of networking and edge devices. Second, there is little evidence of the bi-directional communication—that is, existing approaches only focus on data collection, and do not consider coordination of lower-level devices by modifying data fusion policies in a top–down manner.

Big Data for Smart Healthcare

The rise of Smart Healthcare has triggered an exponential growth of biomedical data sets continuously generated by millions of embedded sensors, introducing new technological challenges along with innovation/business opportunities [12, 34].

The main biomedical data sources include Electronic Medical Records (EMRs) and Electronic Health Records (EHRs) from healthcare providers, as well as clinical trials and data from pharmaceutical companies [31]. To cope with these overwhelming amounts, academia and industry aim to offer efficient solutions to collect, transfer, storage, aggregate, and analyse data from multiple sources (i.e. data fusion).

While storing Big Healthcare Data is relatively successfully addressed by general-purpose storage technologies (e.g. relational and NoSQL databases supported by horizontal scaling and data replication), processing and analysis of healthcare data sets is domain-specific. In particular, the following three disjoint areas of Big Healthcare Data analytics can be distinguished [4]:

- *Image processing* medical images (e.g. tomography, magnetic resonance imaging, radiography, etc.) are an important source of data extensively used for diagnosis, therapy assessment and treatment planning. Such high-resolution images require large storage capacities if persisted for long term. As far as processing is concerned, medical images also demand fast and accurate interpretation algorithms, especially when images serve to assist healthcare personnel in manual decision taking. Furthermore, in cases when such image-based analytics involves other sources of information, the challenge of providing cohesive storage and processing support requires advanced solutions. In recent years, this challenge has been widely addressed by applying artificial intelligence techniques and neural networks [25].
- *Genomics* analysing genome-scale data has been extensively investigated by the computational biology research that aims to develop actionable recommendations for a

clinical setting. Genomic data sets are characterised by extremely high density which makes their exploration, discovery, and clinical translation an attractive are for applying novel Big Data analytics approaches [21, 35].

- *Signal processing* signals, generated by medical sensors, are characterised by their volume and velocity aspects. This is especially challenging in the context of the Internet of Medical Things (IoMT), where a multitude of monitors connected to each patient generates a continuous stream of multi-dimensional sensor readings [2]. Furthermore, in addition to the volume and velocity issues, the spatio–temporal nature of physiological signals needs to be addressed to make analysis of sensor signals more valuable by taking into consideration the situational context. Such context awareness needs to be embedded into continuous monitoring and predictive IoMT systems to ensure that current sensor observations are correlated with the context. Traditionally, existing healthcare systems tend to focus on single sources of information while lacking the remaining context of the patients' true physiological conditions from a broader and more comprehensive viewpoint. Therefore, there is a need to develop improved and more comprehensive approaches for combining multi-modal clinical time series data [2]—a challenge which is currently beyond manual capabilities of humans. As it is further explained in this paper, to address these issues we introduce a data fusion approach based on CEP.

Methods

Proposed approach

A data fusion architecture for a Smart Healthcare ecosystem can be naturally identified by matching IoT infrastructure capabilities within healthcare organisations with the existing data fusion models. The three-level data fusion model, inspired by the taxonomies in Table 2, indeed, naturally fits the classification of IoT processing elements in Table 1, pairing and deploying the low level into edge devices (i.e. Edge computing), the medium one into communication and processing units (i.e. Fog computing) and/or the high one into remote datacenters (i.e. Cloud computing). On this premise, it is therefore natural to think on how to exploit edge devices and network nodes in supporting data fusion tasks. Given the different types of devices and their locations within an IoT network topology, as well as data fusion patterns, in this paper we propose a hierarchical multi-level architecture for data fusion in IoT healthcare systems. According to our approach, data should be first processed on-board (i.e. locally on Edge objects) whenever possible, or pushed to communication and processing units and/or, finally, to remote datacenters following the three-level data fusion pattern in Table 2. As depicted in Fig. 1, the proposed architecture thus includes three conceptual levels, which can also be aligned with geographical areas, from which IoT healthcare data are collected:

I. *Low level data fusion (LDF)* is supposed to take place on smart objects, which collect data, coming from edge devices via gateways, or smart devices themselves—the amount of data is relatively small, and data fusion can be performed on-board. The main goal of LDF is to detect critical health conditions of individual patients. By combining streaming sensor data (i.e. *sensor fusion*) with patients' personal his-

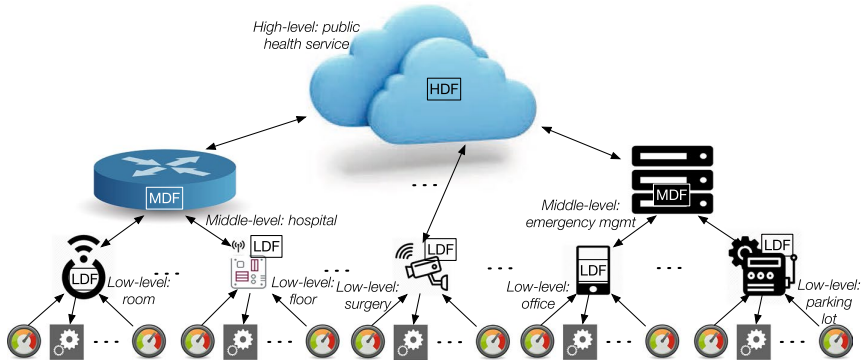

Fig. 1 Overview of a multi-level hierarchical data fusion on a Smart Healthcare scenario. This figure illustrates the proposed approach at a high level. It shows what elements constitute each of the levels of the data fusion hierarchy

toric health records, it is possible to detect potentially critical health conditions, send a corresponding alert to a higher level, and notify hospital staff members in charge. For example, by collecting seemingly 'healthy' sensor readings, the LDF engine is able to match them with personal records and see that for this specific patient the observed values actually might indicate deteriorating health conditions. This way, data fusion—i.e. integration of sensor readings and static background information—takes place as close to the source of data as possible, thus only filtered/aggregated values are transferred to an upper-level processing node, whereas hospital staff is immediately provided with a possible diagnosis.

II. *Middle level data fusion (MDF)* refers to performing more intensive analytics over information pushed from a wider network of devices. In the IoT ecosystems, MDF is supposed to be performed on communication and processing units following principles of Fog computing. The middle-level MDF performs data fusion over a larger area of interest (i.e. a clinic or a hospital), aiming to achieve two main goals. First, MDF aims to detect/prevent potential disease outbreaks, by mapping critical values coming from bed sensor devices to physical locations, where these devices are actually placed. This way, MDF becomes aware of hospital rooms and floors, where there is a potential disease spread. Second, by identifying rooms and floors within the hospital with an increased healthcare demand (including epidemic diseases), MDF aims to organise and manage the staff more efficiently (e.g. by allocating more nurses to a specific floor, where currently an increased demand is being observed). For these purposes, MDF needs to rely on background knowledge with a mapping for individual sensor devices to rooms/floors, where they are placed. By performing data fusion over these multiple data sources, the system is not only able to detect a spread of a sudden infectious disease, but also to efficiently manage and direct hospital personnel by allocating them according to the current demand.

III. *High level data fusion (HDF)* refers to the highest level of data fusion, which provides a global view on the whole managed system of edge devices and networking nodes. This involves processing of large amounts of data, and therefore is expected to be implemented in a remote datacenter or a Cloud. HDF collects data from all

hospitals within a city and performs data fusion over the whole managed urban area. First, this way, the HDF engine is able to provide a global real-time view on the current utilisation of individual hospitals and other healthcare institutions, and manage emergency vehicles accordingly. More specifically, it can combine the streaming information from hospitals and current GPS positions of emergency vehicles with static information on traffic routes and hospital locations nearby. As a result, an emergency vehicle might be directed to a hospital, which is not necessarily the closest to the vehicle at the moment, but which has more available staff to provide first aid in time. Second, the incoming information from multiple hospitals may be correlated to identify and prevent an outbreak of an epidemic disease on the regional scale in a timely manner—a mission-critical requirement in the light of the recent occurrences of the Ebola virus disease and Avian influenza.

It is worth noting that modern IoT systems, especially the ones related to Smart Healthcare, tend to go beyond the notion of the 'bottom–up' monitoring—i.e. raw data are collected by edge devices and transferred through the network- and also implement 'top–down' feedback communication between managed and managing devices (e.g. for actuating commands). As part of such bi-directional communication, the proposed approach introduces the notion of *coordination*, which is responsible for communication between the layers just described. More specifically, coordination is a twofold functionality. On the one hand, it receives data from lower-level devices, collecting and dispatching them to data fusion processors and engines, as well as to the higher level, if required. On the other, it uploads new and modifies/deletes existing processing rules, according to changing requirements.

As a result, the described high-level conceptual data fusion architecture is realised through deploying and running instances of the data fusion logic on devices, constituting multi-level IoT systems. It implements main two functions—namely, (i) actual data fusion, and (ii) communication with devices, located at lower levels of the IoT network topology, and orchestration of data processing tasks distributed across lower-level nodes. The reference architecture in Fig. 2 depicts this conceptual separation of concerns—all three levels are equipped with dedicated data fusion (DF) engine instances (i.e. low-level DF—LDF, middle-level DF—MDF and high-level DF—HDF Engines, respectively), whereas the upper two levels also include coordination components (LDF and MDF Coordinators), responsible for bi-directional communication between lower- and higher-level nodes and management of offloading requests incoming from lower-level nodes.

Interactions among such components are depicted by the workflow in Fig. 3. Initially, data generated by a smart object are gathered and sent to the LDF Engine which first evaluates if the related data fusion task can be processed on-board and, if so, processes it. Otherwise, it splits the task into $n \geq 1$ subtasks which can then be redistributed internally, and/or forwards to Edge nodes, interacting with its middle-level counterpart, i.e. the LDF Coordinator. Similarly, this module assesses the (sub-)task and sends it to the MDF Engine, if processable. Otherwise, it decomposes and redistributes the (sub-)task internally, or forwards the latter to the high-level MDF Coordinator running on the remote datacenter and/or back to the LDF Engine. In its turn, the MDF Coordinator

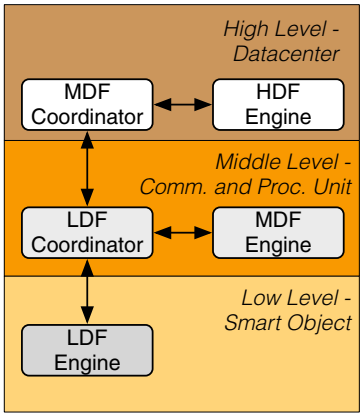

Fig. 2 Conceptual architecture of the multi-level hierarchical data fusion for digital healthcare ecosystems. This figure illustrates the proposed approach at a conceptual level and depicts how data fusion levels interact with each other

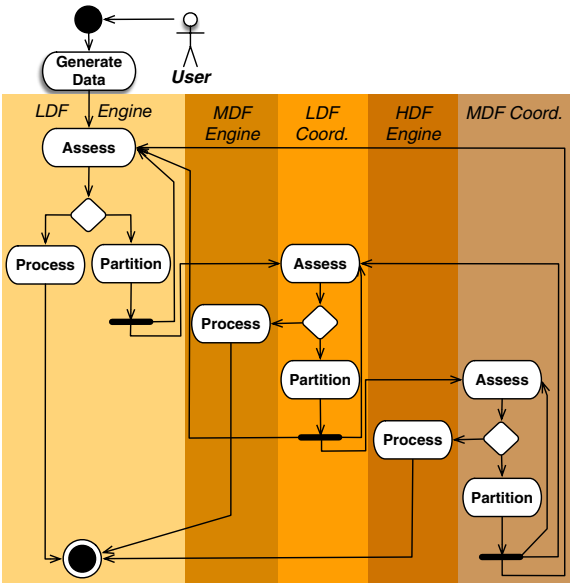

Fig. 3 IoT hierarchical data fusion workflow. This figure depicts the general data fusion workflow where data generated at a low level is processed and transformed as it is pushed upwards through the hierarchy

implements a similar process—i.e. it first evaluates the incoming request and processes it by means of the HDF Engine if feasible, or partitions the request into simpler requests to be re-submitted to the HDF level, and back to the lower MDF/LDF levels through the MDF Coordinator.

It is worth noting that the presence of one of the two upper layers in the hierarchy is optional, since an LDF Engine could directly interact with a MDF Coordinator, or a LDF Coordinator can assign requests only to MDF and LDF engines. Potentially, the approach is flexible in the number of levels in the hierarchy—i.e. it can also accommodate more than three levels, if necessary, by deploying two or more engines and coordinators at higher levels, such as communication and processing units and datacenters.

Case study

Some of the main goals for automated data fusion in the Smart Healthcare domain are (i) to perform situation assessment and minimise the uncertainty by bringing multiple data sources together, (ii) to support timely automated decision taking, and (iii) to reduce security risks associated with sending data remotely over a public network. A potentially promising solution to address such requirements is to use processing capabilities of smart objects, ubiquitously present in IoT networks. However, data fusion in distributed IoT networks is a multi-faceted problem that needs to be addressed from multiple perspectives. Taking the taxonomy in Table 2 as a reference, we specifically consider the following three types of data fusion present in the Smart Health domain, and explain how they can be addressed using CEP.

Temporal data fusion refers to aggregation of data coming from the same data source, but at different points in time. It looks into the chronological order of distinct values, trying to find temporal correlation among them, usually within a specific time frame. For instance, temporal fusion needs to be applied to body temperature sensor readings to detect rapid increases within a short period of time, which are very likely to represent a fever or similar types of health deterioration. Using CEP, temporal data fusion can be implemented by sequential (i.e. to detect the exact chronological order of events) and iterative (i.e. to detect changes with respect to previously arrived values) operators. For example, the former operators can detect critical situations when an increase in body temperature strictly follows a drop in blood pressure, whereas the latter can identify fever symptoms, such as sudden elevations and drops in body temperature.

Spatial data fusion refers to aggregation of multiple physically and/or logically distributed sources into a single common representation. In Smart Healthcare, spatial fusion may need to be performed, for example, over data coming from health sensors placed in different rooms within a hospital with a goal to identify an epidemic pattern. CEP supports spatial data fusion by applying conditions to individual patterns. More specifically, it is possible to check whether events originate from the same location, or compare the source of events with a specific pre-defined location. This way, it is possible to group incoming sensor values with respect to the hospital rooms/floors they are coming from, or extract only a subset of values coming from a specific (i.e. matching) area of interest.

Semantic data fusion refers to the heterogeneous nature of various data sources (i.e. physical and virtual sensors) present in the IoT healthcare domain, which need to be uniformly represented and aggregated in order to enable further analysis and pattern detection. An example of semantic data fusion can be the detection and diagnosis of an illness based on several measured body indicators, such as temperature, blood pressure, heart rate, etc. CEP on its own cannot address the heterogeneity issue of multiple data sources—this task is delegated to an actual CEP implementation, which enables the common data representation and format as part of its programming model. That is, in order to be able to handle, for example, body temperature, blood pressure, or heart rate within a single CEP pattern, it is required to define them as programming classes (possibly as subclasses of the same super-class) with corresponding sets of properties. Once corresponding sensor readings (possibly in an semi-structured form, such as JSON or XML) arrive in the system, they are marshalled into corresponding CEP classes and streamed to the actual CEP engine for processing.

Table 3 Sample data fusion use cases at different levels in healthcare

Data fusion level	Goals	Sample data fusion use cases	Data fusion type	CEP operators to be used
LDF	To detect sudden health deterioration and inform healthcare staff To transfer biometric information about individual patients to higher data fusion levels, specifying the detected symptoms and the location (i.e. room and floor)	Detecting a monotonic increase of a single body health indicator (e.g. body temperature), where every next value is higher than the previous within a given time frame Detecting a rapid simultaneous increase in multiple body health indicators (e.g. body temperature, blood pressure, heart rate) within a given time frame	Temporal semantics	Sequential operators to capture the chronological order of events Iterative operators to compare current values with respect to the previous ones Condition checking to map streaming sensor values to background knowledge on patients' health records
MDF	To allocate more hospital staff on floors with increased demand To detect a potential disease outbreak within the hospital	Detecting identical health deterioration symptoms coming from the same room/floor	Temporal spatial semantics	Condition checking to map streaming values with locations of rooms and floors
HDF	To navigate ambulances to less overloaded healthcare institutions To detect a potential disease outbreak within a metropolitan area	Detecting identical health deterioration symptoms coming from hospitals located in the same area	Temporal spatial semantics	Condition checking to map hospitals and ambulances with their geographical locations

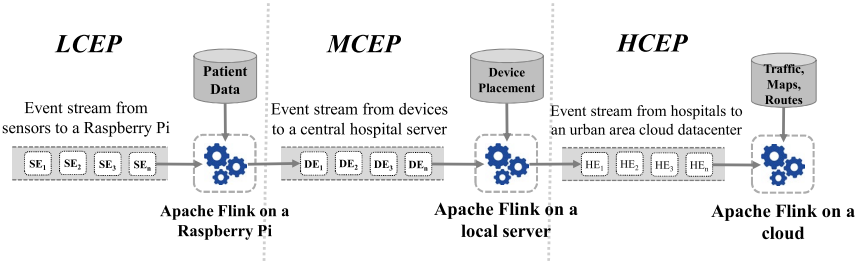

Fig. 4 Implementation of three-level hierarchical data fusion for Smart Healthcare. This figure depicts the implementation details of the pilot prototype. It also shows what CEP framework was used and which events are generated and transferred in the system

Table 3 summarises data fusion levels, goals, and types, as well as potential CEP patterns to be used at each level. As it already follows from the example above, in practice, these three data fusion types usually appear in combinations, meaning that corresponding CEP techniques also need to be combined to address complex data fusion requirements. To this end, we consider several possible data fusion patterns that can be addressed using CEP. These patterns reflect the sample Smart Healthcare scenario, where sensor-rich smart hospital beds continuously collect multiple heterogeneous data related to a patient's health state to be analysed and acted upon at different data fusion levels.

Automation of manual healthcare procedures, such as data collection and fusion, is expected to shorten the time required for establishing a diagnosis. Moreover, timely detection of critical health conditions of individual patients has the potential to contribute to mission-critical activities, such as efficient management of hospital staff and ambulances, and timely detection of disease outbreaks within a single hospital or a whole metropolitan area. The lowest level of the case study is constituted by smart hospital beds, equipped with multiple biometric sensors, among which blood pressure, body temperature and heart rate are primarily taken into account. Please note that the use case scenario is only intended to demonstrate the viability of the presented approach, and is therefore correspondingly simplified.

As a first step towards validating the proposed hierarchical data fusion approach, an initial proof-of-concept prototype was implemented. The prototype utilises Apache Flink[1] as the underlying CEP middleware. Being just one of the pluggable modules of the larger modular platform, Flink CEP is a light-weight open-source implementation, which combines CEP expressiveness (i.e. temporal reasoning over events and sliding windows of interest), relatively low resource requirements, and well-maintained client libraries. As a result, the Flink CEP components were deployed at three levels of the hierarchical Smart Healthcare architecture (as depicted in Fig. 4).

Thus, adopting the proposed approach and terminology, and taking CEP as an underlying data fusion implementation technology, we identify the following three-level CEP hierarchy.

[1] https://flink.apache.org/.

Low-level CEP (LCEP) is deployed on several Raspberry Pi boards, each of which corresponds to a single smart bed and is, therefore, responsible for data fusion concerning a single bed/patient. Accordingly, in the presented scenario, Flink CEP can detect a complex pattern of *sensor events* (*SE*), where all three key indicators (i.e. temperature, blood pressure, and heart rate) are exceeding critical thresholds. It also checks with personal health records whether these increased values are indeed critical for a given patient, provides an initial diagnosis based on these records to a dedicated member of staff, and sends a higher-level *device event* (*DE*) to the next CEP level.

Middle-level CEP (MCEP) is deployed on a server, responsible for collecting data from multiple Raspberry Pi devices, installed across the whole hospital, and performing data fusion over a larger area of interest. It takes as input a continuous stream of device events, containing patient diagnoses and device IDs, and performs CEP by combining streaming data with two types of background knowledge. First, MCEP is able to identify a disease spread, based on previously stored epidemic patterns. Second, it is able to map individual sensor devices to hospital rooms, where they are placed. This way, the system is not only able to detect a spread of a sudden infectious disease, but also to efficiently manage and direct hospital personnel by allocating them according to the current demand. Finally, MCEP sends these aggregated values, containing information about potential diseases within a hospital and current utilisation of the staff, as *hospital events* (*HE*) to the next CEP level.

High-level CEP (HCEP) is deployed on the Amazon EC2[2] Cloud. It collects data from managed hospitals within a city and performs data fusion over the whole metropolitan area. By combining incoming hospital events with a static knowledge base of hospital locations, maps, and traffic routes, HCEP is able to (i) detect an epidemic outbreak on a regional scale, and (ii) provide online navigation instructions to emergency vehicles, directing them to less overloaded hospitals.

More specifically, the case study implements a three-level hierarchical CEP ranging from hospital beds to urban areas. To describe the three CEP level logic formally the complex event logic (CEL) [7] is adopted. The CEL is, to the best of our knowledge, the first and yet not exhaustive attempt to provide a formal logic for the CEP domain. Advanced CEP options and features, such as correlation and time windows, are not yet implemented indeed, so we will try to frame our example into the current CEL core framework. The case study assumes that all devices are globally synchronised, and the events occurring in the environment are time stamped.

To formally specify the Smart Healthcare case study, we first need to define the CEP events involved in this scenario at three levels.

Definition 1 The LCEP sensor event *SE* is defined by the triple

$$SE = \{T, ts, id\}$$

where T is the temperature sample value; ts is the sample timestamp; id is the device id, uniquely identifying the LCEP device generating the sample.

[2] https://aws.amazon.com/ec2/.

This way, the LCEP logic, running on an IoT Smart object-gateway (as defined in Table 3) can be described by the CEL formula below (Eq. 1)

$$\begin{aligned}\varphi_{LCEP} = [&SE \texttt{ AS } x;\ (SE \texttt{ AS } y)+;\ SE \texttt{ AS } z]\\ &\texttt{FILTER}\\ &(x.T <= 37 \wedge z.T > 38 \wedge y.T > x.T\ \wedge (z.ts - x.ts =< 60))\end{aligned} \tag{1}$$

that captures a sudden increase in body temperature, where the temperature of a patient without fever ($x.T <= 37$) starts suddenly raising up ($y.T > x.T$) to fever ($z.T > 38$), i.e. within a minute ($(z.ts - x.ts) =< 60$).

Definition 2 The MCEP device event *DE* is defined by the pair

$$DE = \{\mathbf{SE}, rid\}$$

where **SE** is the set of sensor events triggered by the LCEP according to Eq. (1); *rid* is the hospital room id.

The MCEP logic, running on Fog nodes or servers, is specified by the following CEL formula (Eq. 2)

$$\begin{aligned}\varphi_{MCEP} = [&DE \texttt{ AS } x;\ (DE \texttt{ AS } y)+;\ DE \texttt{ AS } z]\\ &\texttt{FILTER}\\ &(x.SE[i].T >= 38 \wedge y.SE[j].T >= 38 \wedge y.SE[j].id! = x.SE[i].id \wedge\\ &\wedge x.rid == y.rid \wedge (z.SE[k].ts - x.SE[i].ts) =< 60)\end{aligned} \tag{2}$$

that detects similar health deterioration patterns ($x.SE[i] >= 38$, $y.SE[j].T >= 38$) on at least two patients ($y.SE[j].id! = x.SE[i].id$) in the same room ($x.rid == y.rid$) within a minute ($(z.SE[k].ts - x.SE[i].ts) =< 60$).

Definition 3 The HCEP hospital event *HE* is defined by the triple

$$HE = \{\mathbf{DE}, sy, aid\}$$

where **DE** is the set of device events triggered by the MCEP according to Eq. (2); *sy* identifies the symptom; *aid* is the area id.

The HCEP logic, running on Cloud nodes or servers, is specified by the following CEL formula (Eq. 2)

$$\begin{aligned}\varphi_{HCEP} = [&HE \texttt{ AS } x;\ (HE \texttt{ AS } y)+;\ HE \texttt{ AS } z]\\ &\texttt{FILTER}\\ &(x.sy == y.sy == \text{“}Ebola\text{”} \wedge y.DE[i].SE[j].id! = x.DE[k].SE[h].id \wedge\\ &\wedge x.aid == y.aid \wedge z.DE[p].SE[q].id! = x.DE[k].SE[h].id \wedge\\ &\wedge (z.DE[p].SE[q].ts - x.DE[k].SE[h].ts) =< 60)\end{aligned} \tag{3}$$

that detects similar health deterioration symptoms ($x.sy == y.sy == \text{“}Ebola\text{”}$) on at least two patients ($y.DE[i].SE[j].id! = x.DE[k].SE[h].id$, $z.DE[p].SE[q].id! = x.DE[k].SE[h].id$) in an urban area ($x.aid == y.aid$) within a minute ($(z.DE[p].SE[q].ts - x.DE[k].SE[h].ts) =< 60$).

As far as implementation is concerned, below we present these three sample CEP patterns expressed in Java as part of the Apache Flink deployment. In simple terms, the code in Listing 1 defines the LCEP pattern of Eq. (1), where an initial event that carries some physiological measurements, including body temperature, is followed by other events, so that every next temperature value is greater than the previous one, within a time frame of 60 s. This way, it is possible to detect a rapid increase in a patient's body temperature reaching the 38 °C threshold.

Listing 1: CEP pattern for detecting a sharp increase in body temperature.

```
Pattern<LCEPEvent, ?> lcepPattern = Pattern.<LCEPEvent>begin("Initial event")
    .next("Consequent event")
    .oneOrMore()
    .where(new IterativeCondition<LCEPEvent>() {
        public boolean filter(LCEPEvent nextEvent, Context<LCEPEvent> ctx) {
            LCEPEvent currentEvent = ctx.getEventsForPattern("Initial
                 event").last();
            return nextEvent.getTemperature() > currentEvent.getTemperature();
        }
    })
    .within(Time.seconds(60));
```

Listing 2 defines the MCEP pattern specified in Eq. (2) where an initial event that carries a body temperature value greater than 38, is followed by at least one other event, which also exhibits an increased temperature level and originates from the same hospital room, within the last 60 s. This way, it is possible to detect and localise some sort of rapid health deterioration within a hospital.

Listing 2: CEP pattern for detecting identical health deterioration symptoms in the same room.

```
Pattern<MCEPEvent, ?> mcepPattern = Pattern.<MCEPEvent>begin("Initial event")
    .where(new SimpleCondition<MCEPEvent>() {

      public boolean filter(MCEPEvent event) throws Exception {
         return event.getTemperature() >= 38.0;
      }
    .followedBy("Consequent event")
    .oneOrMore()
    .where(new IterativeCondition<MCEPEvent>() {

        public boolean filter(MCEPEvent nextEvent, Context<MCEPEvent> ctx) {
            MCEPEvent currentEvent = ctx.getEventsForPattern("Initial
                 event").last();
            return ((nextEvent.getTemperature() >= 38.0) &&
                 (nextEvent.getLocation() == currentEvent.getLocation()));
        }
    })
    .within(Time.seconds(60));
```

Similarly, Listing 3 defines the HCEP pattern of Eq. (3) where an initial event from a hospital indicates symptoms of the Ebola virus disease, and is followed by at least one other event, which also exhibits same symptoms and originate from the same urban area (but not the same hospital), within the last 60 s. This way, it is possible to detect and localise a rapid disease outbreak within a city.

Listing 3: CEP pattern for identical health deterioration symptoms in the same urban area/region.

```
Pattern<HCEPEvent, ?> hcepPattern = Pattern.<HCEPEvent>begin("Initial event")
     .where(new SimpleCondition<HCEPEvent>() {

      public boolean filter(HCEPEvent event) throws Exception {
         return event.getSymptom().equals("Ebola");
      }
    .followedBy("Consequent event")
    .oneOrMore()
    .where(new IterativeCondition<MCEPEvent>() {

       public boolean filter(HCEPEvent nextEvent, Context<HCEPEvent> ctx) {
           HCEPEvent currentEvent = ctx.getEventsForPattern("Initial
                event").last();
           return ((event.getSymptom().equals("Ebola")) &&
                (nextEvent.getLocation() == currentEvent.getLocation()) &&
                (nextEvent.getSourceHospital() !=
                currentEvent.getSourceHospital()));
       }
    })

    .within(Time.seconds(60));
```

Results and discussion

The proposed hierarchical data fusion approach was compared to the traditional established practice of sending all low-level sensor readings to a central cloud-based CEP component for analysis. This means that multiple sensors were configured to send raw data directly to the Cloud over the public Internet connection. The main benchmarking metric in the experiments was *time delay*—i.e. time difference between the moment when sensor data are first generated and the moment when valuable insights are drawn based on these data. In both cases, communication between individual nodes was implemented using the Message Queue Telemetry Transport (MQTT) protocol. Communication took place either on a local area network (for LCEP and MCEP), or on the public Internet (for HCEP). An average size of an MQTT message, containing several sensor readings, was about 1 kB.

The configuration of all three testbeds is summarised in Table 4, which covers both hardware and network specification. To minimise latency, the physical location of the GCC instance was set to Western Europe. Admittedly, the limited hardware capabilities of the LDF and MDF setups is compensated by the increased throughput and low latency of the local network connection, as well as the relatively small amount of data to be processed. On contrary, the elastic resources of the cloud HDF setup suffers from the increased latency of the public network. Same applies to the traditional centralised approach for healthcare analytics.

Table 4 Testbed hardware specs and network speed test

	Hardware	Uplink, Mbit/s	Downlink, Mbit/s	Round trip time, ms
LDF	Raspberry Pi 3 (1.2 GHz ARM Cortex-A53, 1 GB RAM)	16.58	26.9	144
MDF	PC (2.00 GHz Intel Core i7-4510U CPU, 16 GB RAM)	16.58	26.9	144
HDF	Google Compute Engine *n1-standard-1* (located in EU, 2.52 GHz vCPU, 3.75 GB RAM)	1.24	1.63	482
Centralised approach	Google Compute Engine *n1-standard-1* (located in EU, 2.52 GHz vCPU, 3.75 GB RAM)	1.24	1.63	482

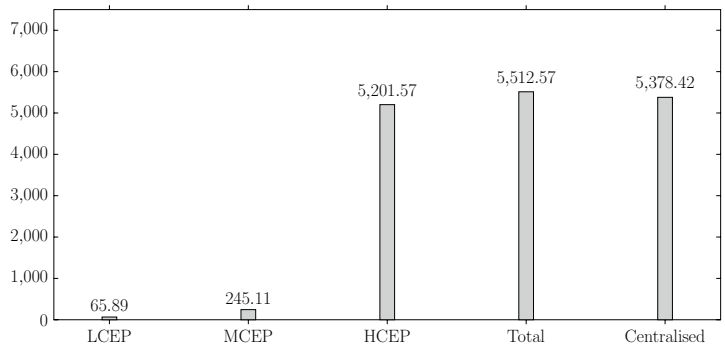

Fig. 5 Processing time benchmarking results, ms. This figure contains experimental results that demonstrate how much time is required at each level to process data and take a decision using the proposed approach, as well as using the existing vertical solution

The results of this comparison are summarised in Fig. 5. The main observation to be drawn from the diagram is the fact that the proposed approach facilitates timely and fine-grained decision taking, as opposed to the centralised approach. That is, decisions are made at each level of the data fusion hierarchy as soon as relevant data are collected. More specifically, decisions about (i) health conditions of individual patients can be taken within 66 ms, (ii) disease spreads in a hospital and staff management—within 311 ms, and (iii) disease outbreaks on an urban scale and emergency vehicle management—within 5513 ms. The latter value is slightly higher than the one corresponding to the centralised approach. The major difference, however, is that with the centralised approach time delay is fixed for all types of problem detection and decision taking procedures. In other words, even the simplest diagnosis task might require more than 5 s to be performed, and another 5 s to be spent on propagating the results back to the hospital staff.

It is worth noting that the presented hypothetical use case scenario is somewhat simplified to demonstrate the general viability of the approach. In practice, the benchmarked time delays are expected to be higher due to network bandwidth, size of network packets being transferred, number of sensors, beds, and hospitals, etc., thus potentially resulting in situations, when health-critical decisions are taken with a fixed time delay of up to several minutes. As demonstrated by the presented approach, it is possible to enable more timely operation by implementing a hierarchical data fusion (i.e. decision taking) architecture. This way, lower-level data fusion is

performed within a LAN, and, therefore, does not suffer from network congestions and increased latency.

Conclusion

The emerging Healthcare Industry 4.0 relies on the ubiquitous presence of smart sensing devices to enable timely data collection and decision. To address the time constraints and deal with the increasing number of heterogeneous data sources, this paper proposed a distributed hierarchical data fusion approach, utilising the processing capabilities of individual nodes of the Smart Healthcare ecosystem, thus splitting data fusion tasks between smart edge objects (i.e. Edge computing), communication and processing units (i.e. Fog computing), and remote datacenters (i.e. Cloud computing). This three-level processing model naturally follows from and is aligned with the existing data fusion taxonomies, where a hierarchical pattern is frequently applied to enable data fusion at various levels. The CEP technology, which natively supports the hierarchical processing of streaming data, was applied as an enabling technology to implement data fusion. Feasibility and effectiveness of the proposed approach has been demonstrated through a Smart Healthcare case study, in which the whole IoT network topology, including sensor-enabled hospital beds, LAN processing units, and Cloud datacenters, was equipped with CEP functionality, thus resulting in a three-level hierarchical CEP framework. The results, albeit demonstrating a greater latency at the highest HCEP level when compared to a centralised stand-alone Cloud-based CEP solution, indicate the effectiveness of the approach at lower levels, where decisions can be taken 20×–90× times faster—a promising achievement given the life-critical nature of healthcare-related decisions.

To make the proposed approach even more flexible, we are eager to extend its functionality with a bi-directional coordination channel. That is, it will be possible to modify the exiting CEP policies at each level in a top–down manner through a Cloud-based management interface. This way, and changes in the CEP rule base (i.e. addition, modification, deletion) will be propagated down the managed IoT topology to reflect emerging requirements. In these circumstances, it is important to ensure that the CEP functionality on participating devices is remotely accessible and configurable—a requirement that can be addressed using the existing containerisation technology, which supports remote dynamic 'injection' of system updates over the network in a seamless and transparent manner [26].

Finally, apart from Smart Healthcare, the ideas and concepts proposed in this paper can be potentially applied to a wider range of smart scenarios, such as Intelligent Transportation Systems, Smart Factories, or Smart Surveillance. In all these systems, sensor-rich edge devices constituting cyber-physical systems continuously generate data streams to be processed and analysed. Accordingly, to meet stringent time constraints and enable near-real time decision taking, it is possible to implement data fusion at different levels using the CEP technology, thereby minimising the impact of network latency associated with remote (Cloud-based) data processing.

Abbreviations

CEP: Complex Event Processing; DE: device event; GPIO: General Purpose Input Output; GPS: Global Positioning System; HCEP: high-level Complex Event Processing; HDF: high level data fusion; HE: hospital event; ICT: Information and Communication Technologies; IDS: Intrusion Detection System; IoMT: Internet of Medical Things; IoT: Internet of Things; JSON:

JavaScript Object Notation; LAN: local area network; LCEP: low-level Complex Event Processing; LDF: low level data fusion; MCEP: middle-level Complex Event Processing; MDF: middle level data fusion; MEC: mobile edge cloud; MQTT: Message Queue Telemetry Transport; SE: sensor event; XML: Extensible Mark-up Language.

Authors' contributions
RD carried out the experiments, drafted the manuscript, and designed figures. SD drafted the manuscript and designed figures. RB contributed to the final version of the manuscript. All authors provided critical feedback and helped shape the research, analysis and manuscript. All authors read and approved the final manuscript.

Author details
[1] Kazan Federal University, Kazan, Russia. [2] University of Messina, Messina, Italy. [3] University of Melbourne, Melbourne, Australia.

Acknowledgements
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Availability of data and materials
Not applicable.

Funding
The authors affirm that there are no sources of funding for the research to be declared.

Publisher's Note

Received: 3 December 2018 Accepted: 13 February 2019
Published online: 25 February 2019

References

1. Almasri M, Elleithy K. Data fusion in WSNs: architecture, taxonomy, evaluation of techniques, and challenges. Int J Sci Eng Res. 2015;6(4):1620–36.
2. Apiletti D, Baralis E, Bruno G, Cerquitelli T. Real-time analysis of physiological data to support medical applications. IEEE Trans Inform Technol Biomed. 2009;13(3):313–21.
3. Appelboom G, Camacho E, Abraham ME, Bruce SS, Dumont EL, Zacharia BE, D'Amico R, Slomian J, Reginster JY, Bruyère O, et al. Smart wearable body sensors for patient self-assessment and monitoring. Arch Public Health. 2014;72(1):28.
4. Belle A, Thiagarajan R, Soroushmehr S, Navidi F, Beard DA, Najarian K. Big data analytics in healthcare. BioMed Res Int. 2015;2015:370194.
5. Bhargavi R, Vaidehi V, Bhuvaneswari P, Balamuralidhar P, Chandra MG. Complex event processing for object tracking and intrusion detection in wireless sensor networks. In: 11th international conference on control automation robotics & vision (ICARCV). 2010. p. 848–53.
6. Brunelli D, Gallo G, Benini L. Sensormind: virtual sensing and complex event detection for Internet of Things. In: international conference on applications in electronics pervading industry, environment and society. Berlin: Springer; 2016. p. 75–83.
7. Bucchi M, Grez A, Riveros C, Ugarte M. Foundations of complex event processing. https://arxiv.org/pdf/1709.05369.pdf.
8. Dautov R, Distefano S. Distributed Data Fusion for the Internet of Things. In: International conference on parallel computing technologies. Berlin: Springer; 2017a. p. 427–32.
9. Dautov R, Distefano S. Three-level hierarchical data fusion through the IoT, edge, and cloud computing. In: Proceedings of the 1st international conference on Internet of Things and machine learning. New York: ACM; 2017b. p. 1.
10. Dautov R, Distefano S, Bruneo D, Longo F, Merlino G, Puliafito A. Pushing intelligence to the edge with a stream processing architecture. In: 2017 IEEE international conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). 2017. p. 792–99.
11. Díaz M, Martín C, Rubio B. State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing. J Netw Comput Appl. 2016;67:99–117.
12. Dimitrov DV. Medical Internet of Things and Big Data in healthcare. Healthcare Inform Res. 2016;22(3):156–63.
13. Durrant-Whyte HF. Sensor models and multisensor integration. Int J Robot Res. 1988;7(6):97–113.
14. Ficco M, Romano L. A generic intrusion detection and diagnoser system based on complex event processing. In: First international conference on data compression, communications and processing. 2011. p. 275–84.
15. Fonseca J, Ferraz C, Gama K. A policy-based coordination architecture for distributed complex event processing in the internet of things: doctoral symposium. In: Proceedings of the 10th ACM international conference on distributed and event-based systems. New York: ACM; 2016. p. 418–21.
16. Friedlander D, Phoha S. Semantic information fusion for coordinated signal processing in mobile sensor networks. Int J High Perform Comput Appl. 2002;16(3):235–41.
17. Garcia Lopez P, Montresor A, Epema D, Datta A, Higashino T, Iamnitchi A, Barcellos M, Felber P, Riviere E. Edge-centric computing: vision and challenges. SIGCOMM Comput Commun Rev. 2015;45(5):37–42.

18. Guo Q, Huang J. A complex event processing based approach of multi-sensor data fusion in IoT sensing systems. In: 4th international conference on computer science and network technology (ICCSNT), vol 1. 2015. p. 548–51.
19. Haghighat M, Abdel-Mottaleb M, Alhalabi W. Discriminant correlation analysis: real-time feature level fusion for multimodal biometric recognition. IEEE Trans Inform Foren Sec. 2016;11(9):1984–96.
20. Joyia GJ, Liaqat RM, Farooq A, Rehman S. Internet of medical things (IOMT): applications, benefits and future challenges in healthcare domain. J Commun. 2017;12(4):240–7.
21. Khoury MJ, Ioannidis JP. Big data meets public health. Science. 2014;346(6213):1054–5.
22. Klein LA. Sensor and data fusion: a tool for information assessment and decision making, vol. 138. Langen: Spie Press; 2004.
23. Krügel C, Toth T, Kerer C. Decentralized event correlation for intrusion detection. In: International conference on information security and cryptology. Berlin: Springer. 2001. p. 114–31.
24. Lee J, Kao HA, Yang S. Service innovation and smart analytics for industry 4.0 and big data environment. Procedia CIRP. 2014;16:3–8.
25. Litjens G, Kooi T, Bejnordi BE, Setio AAA, Ciompi F, Ghafoorian M, Van Der Laak JA, Van Ginneken B, Sánchez CI. A survey on deep learning in medical image analysis. Med Image Anal. 2017;42:60–88.
26. Longo F, Bruneo D, Distefano S, Merlino G, Puliafito A. Stack4Things: a sensing-and-actuation-as-a-service framework for IoT and cloud integration. Ann Telecommun. 2017;72(1–2):53–70.
27. Luckham D. The power of events, vol. 204. Boston: Addison-Wesley Reading; 2002.
28. Mahmud R, Koch FL, Buyya R. Cloud-Fog Interoperability in IoT-enabled Healthcare Solutions. In: Proceedings of the 19th international conference on distributed computing and networking (ICDCN 2018). New York: ACM. 2018. p. 1–10.
29. McAfee A, Brynjolfsson E, et al. Big Data: the management revolution. Harvard Bus Rev. 2012;90(10):60–8.
30. Nakamura EF, Loureiro AAF, Frery AC. Information fusion for wireless sensor networks: methods, models, and classifications. ACM Comput Surv. 2007;39:3.
31. Nambiar R, Bhardwaj R, Sethi A, Vargheese R. A look at challenges and opportunities of big data analytics in healthcare. In: 2013 IEEE international conference on Big Data. 2013. p. 17–22.
32. Neves PA, Rodrigues JJ, Lin K. Data fusion on wireless sensor and actuator networks powered by the ZenSens system. IET Commun. 2011;5(12):1661–8.
33. Pantelopoulos A, Bourbakis NG. A survey on wearable sensor-based systems for health monitoring and prognosis. IEEE Trans Syst Man Cybern C. 2010;40(1):1–12.
34. Raghupathi W, Raghupathi V. Big Data analytics in healthcare: promise and potential. Health Inform Sci Syst. 2014;2(1):3.
35. Stephens ZD, Lee SY, Faghri F, Campbell RH, Zhai C, Efron MJ, Iyer R, Schatz MC, Sinha S, Robinson GE. Big Data: astronomical or genomical? PLoS Biol. 2015;13(7):e1002195.
36. Thuemmler C, Bai C. Health 4.0: application of industry 4.0 design principles in future asthma management. In: Health 4.0: how virtualization and Big Data are revolutionizing healthcare. Berlin: Springer. 2017. p. 23–37.
37. Verbelen T, Simoens P, De Turck F, Dhoedt B. Cloudlets: bringing the cloud to the mobile user. In: Proceedings of the third ACM workshop on mobile cloud computing and services. New York: ACM. 2012. p. 29–36.
38. Wang Y, Cao KA. Proactive complex event processing method for large-scale transportation Internet of Things. Int J Distrib Sensor Netw. 2014;10:3.
39. Zhao C, Wang YA. New classification method on information fusion of wireless sensor networks. In: IEEE 2008 international conference on embedded software and systems symposia. 2008. p. 231–36.