Event-Driven Video Services for Monitoring in Edge-Centric Internet of Things Environments

Nikita Bazhenov, Dmitry Korzun
Petrozavodsk State University (PetrSU)
Petrozavodsk, Russia
{bazhenov, dkorzun}@cs.karelia.ru

***Abstract*—Many existing video services are based on transferring the data stream from a video capture device to some powerful data processing center (e.g., cloud-based). In this paper, we consider video services constructed in an edge-centric Internet of Things (IoT) environment. Several edge video capture devices are available. Each data stream is subject to local processing for event detection. Complex events can be also locally detected based on the observed context situation in the IoT environment, forming the semantics for video-analytics. We present a concept model of an edge-centric service-oriented system and its event-driven model for local video data processing. In our demo implementation, we consider a video service of personnel recognition around a production equipment unit, which is used in digital monitoring the smart manufacturing process.**

I. Introduction

The Internet of Things (IoT) technology enables development of advanced digital services [1]. A promising way to construct such a service is provided by Edge-Centric and Fog computing [2]. Video services cover a wide class of IoT-enabled services, where one or many IoT devices in the IoT environment can capture video data, and several video streams are mined with known analytics methods [3] and possibly in real-time mode [4].

As in basic video services, an individual data stream from its capturing device is subject to local processing for event detection based on well-elaborated video data mining algorithms such as human face recognition [5]. Then complex events can be detected based on semantic linking the events observed in several individual video streams as well as using observed context information in the IoT environment. In this paper, we continue our previous work on video serviced development as IoT applications [6]. A particular example is video surveillance when events in video stream of smartphone camera are augmented with related events detected from other surrounding cameras [7].

The analytics can be implemented locally, following the edge computing [8]. One way is making a camera smart, where the video-analytics is implemented by software agent [9]. If the camera capacity is low then the data processing can be delegated to other devices (agents) nearby [10]. As a result of this multi-agent cooperation, the shared semantics are formed and used for video service construction [11].

We study an advanced class of video services deployable in edge-centric IoT environment. The key features of the studied service class are (a) microservice-based system [12] for operating with multiple data streams, (b) event-driven model [13] for service construction, and (c) context-based mechanisms [14] for use in video-analytics. We present a concept model of an edge-centric microservice-oriented system and its event-driven context-based model for local video data processing. In our demo, we implemented a video service of personnel recognition around a production equipment unit, which is used in digital monitoring the smart manufacturing process.

The rest of the paper is organized as follows. Section II shows related work identify the features of the studied class of video services. Section III considers IoT enablers for our video service development to show how advanced properties can be accounted in video services. Section IV introduces our model for event detection and context construction. Section V describes our proof-of-the-concept demo implementation. Finally, Section VI concludes the paper.

II. Related Work

Video services define a topical area for research and development since IoT environments include various video capturing devices. This section considers existing methods of video data processing in services to identify possible extends for the services. Our reference example is video surveillance and video surveillance services and their possible extensions [4] in various application areas.

In the basic version, video surveillance uses a single video stream from a recording device (camera) to a user (client). That kind of video surveillance does not provide interactivity, multithreading, a large number of clients, and ability to process big data and non-video data (e.g., recommendations in video-based systems). In particular, work [4] considers an advanced video surveillance system, where video analytics and edge computing algorithms are combined to create the low-cost and high-precision environment for analyzing the movement of different objects. We expect that many advanced video surveillance services can be implemented primarily on the edges of Internet.

The first feature is Microservice-based system [12]. The feature applies the shortest delivery path for a service. Many existing video service delivery solutions are insufficient, often video service developers forget about using which algorithms the services will be delivered, the main insufficient parameters may be: lack of understanding how many clients will use the system and therefore load it, inaccurate definition of the class

of video services that focuses on consumer, and therefore to the end user, too narrow or too broad a definition of subject areas within which a service-oriented system will be described. Moreover, services are not delivered to customers properly, taking into account all the necessary requirements (real-time computing, high quality image transmission, a large number of customers). The most effective and reliable solution may be the use of microservices, where delivery to customers occurs using a large number of small self-contained modules. In general, microservice based architecture can be useful in our case to create smart video services.

The second feature is Event-driven model [13]. The feature applies the object-oriented concept based on current events within the framework of the unified modeling language UML. Designing a service-oriented system based on events that occur during the extraction of video data may be a useful case for developers of video applications. The created layout of the presentation and interaction between the components of the video system may not be clear and understandable when the customer is the developer of the video application. In this case, the implementation of the concept or video model can be achieved by constructing descriptive diagrams and models. Also, such a model can be represented as a collection of events that are recorded and monitored by a video camera. Moreover, the events in the video data can be complex and versatile, which will allow the most detailed study of the received data for the service.

The third feature is Context-based mechanisms [14]. The feature applies micro and web services in various system conditions. The following aspects of QoS are accented: reliability, availability, response time, including taking into account the overall load on the server and other components of the system by selecting special instances for the microservice. As a part of the creation of a video model, it is necessary, among other things, to focus on requests from end users who will eventually interact with the camera interface (for example, receiving a video stream in real time, interacting by controlling multiple cameras, automatically tracking events in large amounts of video data). Context-oriented information systems allow you to organize additional intelligence due to significant calculations in the framework of the most complex, multi-structured video data. There is a need to provide the most appropriate contextual information for the user in the case when the current presentation of the data for one reason or another does not suit him (for example, some results after analyzing the video data are presented in the form of graphs, when the user wants to see them in the form of histograms).

The identified features describe the class of services that need to be created as a part of video surveillance tasks. Existing solutions describe a specific algorithm or method that implements a certain advantage in the framework of services, including those related to video data. There is a need for a class of services that would provide end users with the necessary services related to video. We consider the video surveillance services that include the basic components: cameras and video surveillance objects, a server that analyzes video data based on available methods and algorithms, a database that stores basic information about video cameras and events, clients (users) who will use these video services with the following features.

The identified features support implementation of an advanced video surveillance system. We focus on the following reference examples (already existed in many implementation variants).

First example is “Recognizing the activities of workers and equipment in order to identify deviations” within the tasks of production monitoring [10]. Applied area describes industrial enterprises using robotic production equipment (e.g. machine tools), industrial categories enterprises: metalworking, foundry, food production, microprocessors for consumer electronics. In such area class of video services is designed to reduce the cost of repairs and additional costs for the purchase of materials for machinery equipment, increase the efficiency of production equipment, reduce the human factor in the efficiency of equipment and the quality of finished products through the use of industrial Internet technologies to improve monitoring processes, further planning, boards of maintenance, repair and operation of equipment based on methods for analyzing large heterogeneous data using embedded and mobile technologies, as well as video surveillance cameras.

Second example is “Real-time patient assistance” within the At-Home Lab with Smartphone help for patients with degenerative diseases of the motor system (Parkinson’s disease, PD) [15]. A smart video camera installed at the patient’s home can record video and send it to the attending doctor. In this case, after analyzing the patient’s behavior and gait, microservices can be represented in the form of delivery of various recommendation systems and tips from the doctor to patients to mobile smartphones remotely. The events in this case will be various activities of the patient: “woke up and got out of bed”, “had breakfast”, “carried out physical exercises”. The context will be the simplest activity of the patient as a result of forming something more complex, for example, “had breakfast” consists of “took a spoon”, “put food on a spoon”, “put a spoon in the mouth”.

Third example is “Detection of collaborative activities of people” within the MICE (Meetings, incentives, conferencing, exhibitions)-systems [7]. In this case, microservices will be various objects of the surrounding world that provide their micro-help as part of the common activities of several people (for example, using a smart board to draw several people at the same time). The event will be directly any joint activity: “Mary and James jointly edit the image.” A service can provide context-sensitive relationships in the form of similar images or types of durability depending on what a group of people is doing now.

Fourth example is “Details recognition on the conveyor line using smart mini-board computer” within the Industrial automation and automatization [10]. In the tasks of industrial automation, microservices will be the recognition of various details. Events in this case will be simple, they will mean that some part was recognized, in this case, such a model can be complicated by analyzing the chain of events (for example, 5 consecutive details of such and such a company were recognized). The context for enterprise workers who are users of the system will detail the description of the part, its shape, color, thickness, manufacturer, and so on. Thus, the class of video services that was created with using of that features should be useful to both developers of video systems and users of certain application areas.

The related work above showed the advantages of various systems in which the authors made an attempt to describe the features given above. These features are implemented depending on the application area. Examples of implementations were also given above. The advantage of using these features in their entirety, that is, they will be implemented in the system as separate properties, but at the same time retaining their consistency and be presented as a whole.

III. IoT-enabled Video Services

Based on the features identified in the previous section, as well as on the basis of the functionality and other video surveillance activities within the IoT technology, this section shows the studied class of video services. First, we would like to give an example of some papers from other authors that are available in the field of video surveillance in IoT at the current moment and papers in IoT as a concept generally. Then we determine what information the given features provide and highlight the corresponding IoT properties. After that we show the functional roles of agents in IoT and present them as entities (software components). As a result, we present these software components in the form of a layered architecture in order to show how these components interact with each other being at certain levels.

Nowadays the popularity of video surveillance is increasing due to the ever-growing need for tracking security objects. There are some new paradigms of universal computing in the field of the IoT. An example of modeling a unified solution for a video surveillance system based on the ubiquitous eIoT is shown in [16]. The traditional video surveillance scheme shows how video components are arranged at a basic level. It can be used to show on what ”basic” video surveillance is based on (video camera - video server - client), improving the ”traditional” scheme to ”advanced” scheme using additional streams on the video camera and using a broker (SIB, Semantic Information Broker) that supports the interaction of agents among themselves through the exchange of information and its semantics.

Modern service delivery varies in its architectural type: IaaS, PaaS, SaaS (Infrastructure, Platform, Software as a Service). The main differences between these architectures are the difference in the number of managed components. For example, PaaS allows to create and manage with working applications and data, SaaS provides ready-made applications within which a service developer implements the application. An example of IaaS type that implements a reliable and comprehensive video surveillance system for Smart City using mobile devices is given in [17]. It can be used as a final solution to provide the infrastructure where we video surveillance services can subsequently developed while using our proposal conditions and software components as a part of IoT.

As an example, we can consider a production site where it is necessary to monitor the workers and their actions at the workplace in order to automate production and help to eliminate errors. When the equipment is in operation, the operator monitors the progress of the work and responds to emergency situations. Therefore, it is important to have an operator at work place during operation of the equipment. It tracks the location of the operator and warn for interested persons, if he is absent from the workplace or if he performs actions for which he is not entitled (opening the electric cabinet, penetrating the machine during work). Implementation requires that one or more cameras will be installed generating the images from which it is supposed to track the location of the operator relative to the machine and other machine facilities.

TABLE I. Video services features

Features	Provision of information	IoT-properties	Advantages
Microservice-based	General background information	Multiple cameras with multiple threads and data	High productivity, flexibility, sustainability, scalability
Event-driven	Analytical ”advanced” information	Mobile and heterogeneous activities	Easy to understand, easy to detect and use, intuitive for end-users, deployable
Context-based instances	Recommendations and advices	Heterogeneous data and calculations	Intellectuality, interactivity, interoperability

Consider the case when several cameras are located in the workplace (workshop) to track various activities between workers and the equipment they use. It is necessary to provide various types of services:

- Informational: informing the worker about the event (for example an incident that happened due to incorrect broken machine work);
- Analytical: the opportunity to review and analyze the data leading to the onset of some event;
- Recommendational: predict the occurrence of an event, which allows to recommend to the user make some actions with the equipment

This can be implemented using the general features of video surveillance, which can be expressed as the following IoT properties, presented in Table I. It outlines the features of video development, what information they provide, what activities they should complete and what benefits can be obtained using these features. A big amount of information comes from several cameras in the form of video streams. Each camera can be assigned its own micro service, which receives and processes these video streams. Then, the information must be analyzed and at the output receive the analyzed information in the form of simple and complex events. Employees can be assisted in the event of equipment malfunction or in the event of an emergency. This can be achieved through context-based instances.

The systematic concept and formalization of the terminology of video data in the framework of the Internet of things are not well described, and the above features and their advantages are not fully disclosed and in certain circumstances require specification. We propose introducing our functional roles for video surveillance within the framework of IoT environments, reflecting their activities in the form of a table and showing how they can be implemented within the framework of a specific video surveillance architecture.

The basic architectural models for smart spaces are described in [18], [19], with focus on generic roles of agents

TABLE II. FUNCTIONAL ROLES OF ENTITIES IN IOT VIDEO SERVICES

Role	Description	Activity	Implementation
Client (User)	Represents the end user of services, a human user	Provide information to Broker about the user and deliver services to the User	Personal mobile device, end-user (PC, smartphone)
Video sensor	The source of video information received from video cameras and other devices that can capture video	Provide information in Broker of the current state of the context of video sensors from IoT environments and their context	Smartphone camera, web-camera, street camera
Sensor	The source of information received from sensors, sensors and other low power IoT-devices	Provide reflection in Broker of the current state of the context of the sensors from IoT environments and their context	Temperature sensor, accelerometer, gyroscope
Sink	Accumulates content from Sensors and Generators, performs analysis of accumulated information content	Extract knowledge from the available information and send it further through the communication Nodes	Module
Node	The point of transmission of the information component	Transfer of knowledge from available information, including the publication of actions in Broker	Data transmitter
Gateway	Represents a finite entity that connects to the Internet	Provide information to Broker from external sources (entities) and vice versa, transmitting the information component of this entity	External network device (router)
Representative	Software implementation of a range of services focused on specific subject area (s)	Retrieve information about processes taking place in Broker in time	Service
Monitor	Monitors information in the broker and transfers it further through the communication nodes	Provide Broker with monitored content	Simple events monitor
Generator	Generates information in the broker and transfers it further through the communication nodes	Provide Broker with generated content	Complex events generator
Interpreter	Represents an intermediary between agent and another software entity	Provides Broker with the necessary information from the service and vice versa: performs interpretation of the transmitted information	Video server, database

(knowledge processors). In the case of video data, the functional roles of agents in terms of the Internet of Things should be discussed and described. Entities inside table may be too abstract, but at the same time, these entities can describe the interaction of video data with the IoT.

Due to the large number of IoT properties, it is necessary to highlight the functional roles that will be performed within the IoT environment based on our example. Table II is based on the following definitions: role, agent description, activity, implementation. A large number of roles are information agents that generate information and exchange it within edge-computing (for example, video sensor, sensor, sink). This role classification shows the complexity and heterogeneity of the information that video cameras and sensors generate, which is necessary for further interpretation. The interpreter in this case represents a video server and a database where video streams are analyzed and the necessary information is extracted.

The next step is to build a multi-level (multi-layer) architecture in order to show on which layers the functional roles of agents and their functional tasks are used. The construction of a video service is organized at the expense of a multi-

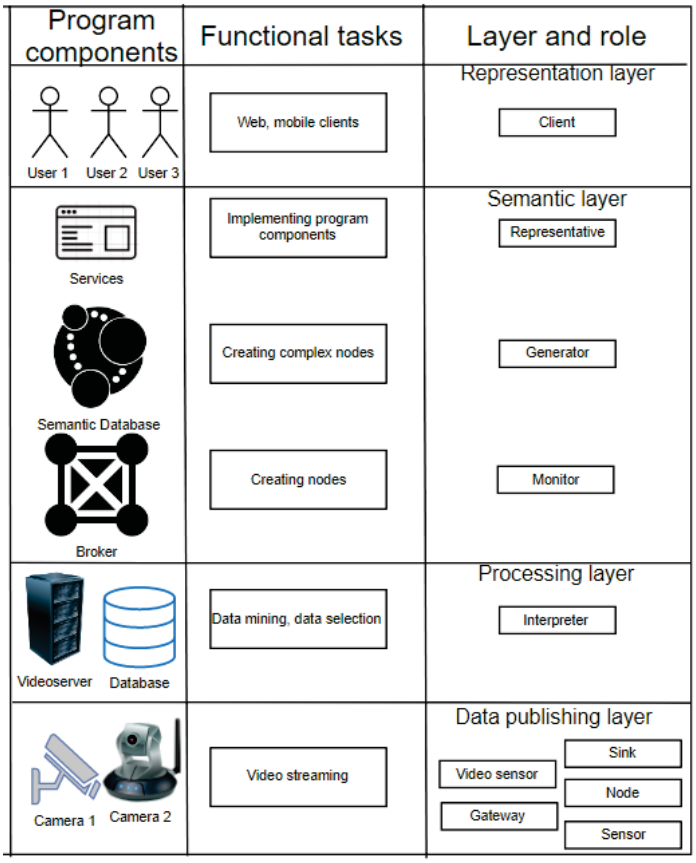

Fig. 1. IoT-enabled multi-layered architecture of video services

layered architecture. Information within the system should be distributed among the participants.

The IoT-enabled multi-layered architecture of video services is shown in Fig. 1. The data publishing layer is represented by devices generating data (sensors), in this example, visual sensors (video cameras) generating a video data stream are shown. The processing layer is represented by a video server for data analysis (application of video image analytics algorithms depending on the choice of implementation method) and a database for loading / unloading data. This layer may include various data storages, neural networks, as well as any components that analyze or save the stream of video images from cameras. The semantic layer is represented by several components. A semantic broker creates information-dependent relationships between video components (for example, linking semantically similar images). The semantic database creates complex multi-component relationships, and the service implements semantics in the form of software components (web, interfaces, mobile clients). Users represent the level of presentation and that they are service end-users.

The architecture is implemented in the form of layers and represents a certain set of interconnected entities, reflected at a certain network level, similar to OSI model (The Open Systems Interconnection model). The top layer (service layer) in general terms represents the service that organizes relationship with external sources of information (in this case, cameras). Semantic the layer implements Broker, it is presented as a semantic

broker, which is a generator of information rich in semantics. The processing layer is represented by a video server, storing information related to video analytics; database collecting any other information. The consumer layer is user-submitted. consuming content in the form of ”finished” information (graphs, charts, images, text).

The core service (Service 0) at the software level implements the connection to the cameras of the IoT environment. Service is the initial component of the system and directly organizes interaction with video sources. Particular services from [15], [7], [10] are examples for end-users. The services in total implement the required set of video analytics algorithms. Each service is presented as a software component (module). Thus, the environment can be expanded and customized for specific tasks (e.g., smart home). The following particular services are considered.

1) Ambient Assistant Living: At-Home Lab with Smartphone help for patients with Parkinson Disease (PD) [15];
2) Public services and conferences: Face detection and recognition, collaborative work detection [7];
3) Industrial automation and automatization: Details recognition on the conveyor line [10].

The main processing of video data is performed on a video server (usually a local network computer). In its file system, you can save video streams from cameras. To store information retrieved from video data, the video server uses a non-relational database. In particular, our current implementation uses MongoDB. The chains of events detected in individual video streams are saved. For each event, a set of attributes describing the event is stored. The semantic layer that was represented in the model, implemented with the help of a broker of the relationship between components and the ontology of the state of the environment. The broker collects the necessary information from the services. Ontology allows you to build and maintain a semantic network. To store the semantic network, the Neo4j semantic database can be used as an example, which is implemented as an add-on on the MongoDB database. The semantic network connects many events from individual video streams, allowing you to detect complex events (e.g., recognition of the same person from multiple cameras). In this case, it is possible to calculate and save a set of attributes for each relationship within a composite event.

In summary, we highlight the following steps in video system developemnt.

1) selection and description of the application area, conditions and tasks of creating the system;
2) highlighting the main components and definitions of the system;
3) establishing relationships between components, a description of the interaction.

IoT-enabled video services have all the advantages represented in the form of the following points: providing heterogeneous information depending on the needs of users for communication of video services, attracting a large number of devices (including mobile devices) to a common computer network, variability and diversity of the implementation of video services depending on the application area, automating various processes in human life and in industry using a large number of video devices.

IV. Event-Driven Video-Analytics Model

Next-generation Internet of Things systems have new problems associated with video surveillance: high CPU and RAM costs, fast and voluminous accumulation of big data on the hard disk, progressive delay depending on connected clients. The proposed solution in [20] provides video surveillance in the Internet of Things (IoVT) using several layers of information processing (edge, fog, cloud), exploring the behavioral information of video data. In this section, we describe a conceptual model of video data based on the IoT-enabled components architecture from the previous section.

The solution to the problem of high server load and high load on hard drives can be the classification of large amounts of incoming video information and the selection of necessary information. Thus, a particular solution to this video surveillance problem may be to search for contextual information. For example, at a manufacturing enterprise with the help of several cameras it is necessary to solve the problem of finding a person on the image in the current frame. The first problem can be a large amount of data, even when using the most efficient data transfer and analysis algorithms. Another problem is the complexity of the data itself, its multi-structured representation. We offer a solution based on a service-oriented event system model.

The advantages of event-driven features have been shown previously. Let us describe this as part of the application: monitoring production equipment in order to detect deviations using a video surveillance system. The phenomena that occur with machines during production or when people interact are easiest to describe in the form of simple events - a one-time discrete phenomenon that occurs in a certain point of time and represents one of the characteristics (beginning, end, tick of the T-Tn timer). So the beginning of an event is a discrete moment of time, meaning the beginning of an event (0 to 1); the end of an event is a discrete moment of time, meaning the end of an event (1 to 0); a long-term event is an interval phenomenon that occurs in a certain period of time, the beginning and end of a long-term event are simple events; the tick of the T-Tn event timer is a discrete time instant that is triggered after a certain equal period of time after the start of the event; the event truth criterion is a means of verifying the truth of the beginning / end of an event, calculated on the basis of a large amount of data; compound event - an phenomenon that includes a chain of several consecutive or simultaneous events that form one complex (can contains several simple and/or long-term events); layer is a symbol of a set of components interacting with each other.

Similar solutions are being investigated by other scientists. In particular, work [21] introduces the Collaborative Internet of Things (C-IoT) paradigm that is aimed at involving multiple devices in collaboration on data collection and resource sharing. Such kind of paradigm can be used in smart cities and in environmental monitoring. At the same time, video data received from video cameras can be analyzed by both a video server and mobile clients, depending on the context.

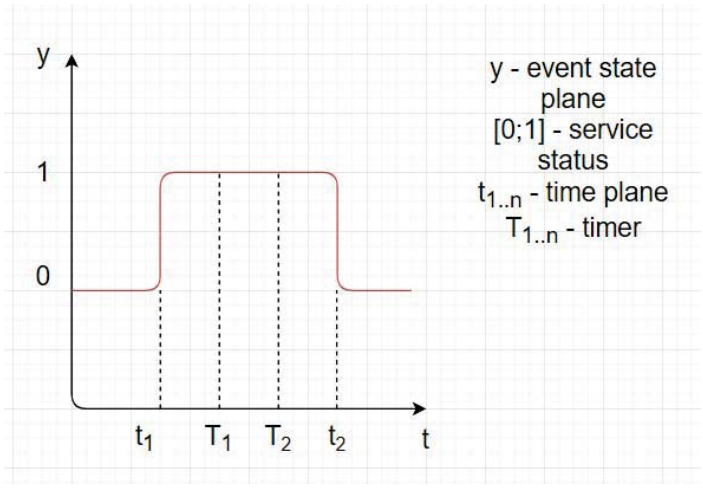

Fig. 2. Event plane graph

Event plane graph is shown in Fig. 2. The schedule of events is represented on the chart. The y axis represents the event state plane, the t axis represents the time plane. In some cases, the state of a simple service can be considered in the representation of two values $(0, 1)$. In the example with a worker located near the machine, the service will show 1, if the machine does not have anyone, the service will show 0. On the graph at time t1, a person appeared at the machine, at time t_2 he moved away from the machine. Thus, the module that implements this event sends the simplest queries to the MongoDB database in case of a change in the current state of the service.

In some cases, it becomes necessary to additionally check the working status of the service, since it can stop working at any time (for unknown reasons). It is supposed to use a timer $T(T_1, T_2)$ that would send requests to a specific database to request the current status of the service. If the service has not changed the value for a long time, this may indicate that the service is currently inoperative, which means that the status needs to be set to 0. For a more accurate timer setting, you need a large amount of statistics that we currently have are not available, therefore the timer can be set provisionally (for example, once every 5 minutes).

Some events may be represented by other data. For example, we are faced with the task of detecting the number of people in a certain territory. In this case, the main state of such an event will be the current number of recognized people. At the same time, this same problem can be solved in another way in the form of complex events. Suppose there are several cameras on the territory, each of which is directed to an area where there can be a person (and no more than 1). In this case, each individual module that processes a specific video stream from the camera will determine whether there is a person in the camera or not and create an event in the database if the person is in the camera, changing the status of the event to 1. Then, you can track the status of all cameras of all modules and see how many people are currently on the territory. In complex systems, in the presence of a large number of not only video cameras, but also various sensors, but also other devices of the IoT environment, it is also possible to present any incidents or phenomena in the form of simple events and then generate complex events. It is possible to solve the problem when it is necessary to analyze many simple events, identify the relationships between them and build complex events, which will then be presented as web services for end users.

Visual model of the video services based on events is shown in Fig. 3. The following components are used.

1) Camera 1;
2) Camera 2;
3) Human detection module;
4) MongoDB Database;
5) Simple events generator;
6) Complex events generator;
7) Operator presence detection service;
8) Web representation;
9) Users abilities.

The video service should implement a system of primary processing of video data for operational monitoring, including the conversion of analog signals to digital, calculation of statistical estimates, etc. The result is streaming data transmitted further for subject-oriented analysis and storage (both on local and remote computers). Requires the use of well-known industrial automation equipment. It is possible to use typical computers (desktop computer or laptop) for local processing of incoming measurements. The size of the service can be up to 100 different data streams coming from several cameras per 1 unit of robotic production equipment.

The document-oriented MongoDB database was chosen as the main information storage for storing and processing temporary data in large volumes. The main object of such a data model is an event, which is presented as a separate document in JSON format for each event that occurs in the system, has links to related events, and contains links to data.

In the visual model, several cameras (2 or more) are processed by camera modules (receiving configurations of video cameras and stream). Including the sensor module processes the data from the sensors. Then the data goes to the MongoDB database. The events module monitors the main events occurring from the video camera, accessing the MongoDB database. As an example, simple events A and B are created here. The observer of complex events creates composite (complex) events on the basis of data from the database and simple events. Such events are complex and may consist of several simple ones (for example, the readings of video cameras and sensors together create a complex event). The service module creates an application (web-interface) for end users in the form of camera images, data analyzed, and so on.

The presented event-oriented model can be used in the following applications:

- home security and recognition of violators in a protected area;
- determination of human health indicators using recognition of human movement (gait);
- recognition of objects (types, quantity, condition) on the conveyor line in production;
- recognition of the interaction of people and physical objects (e.g., the employees' proximity to the machine in the production hall).

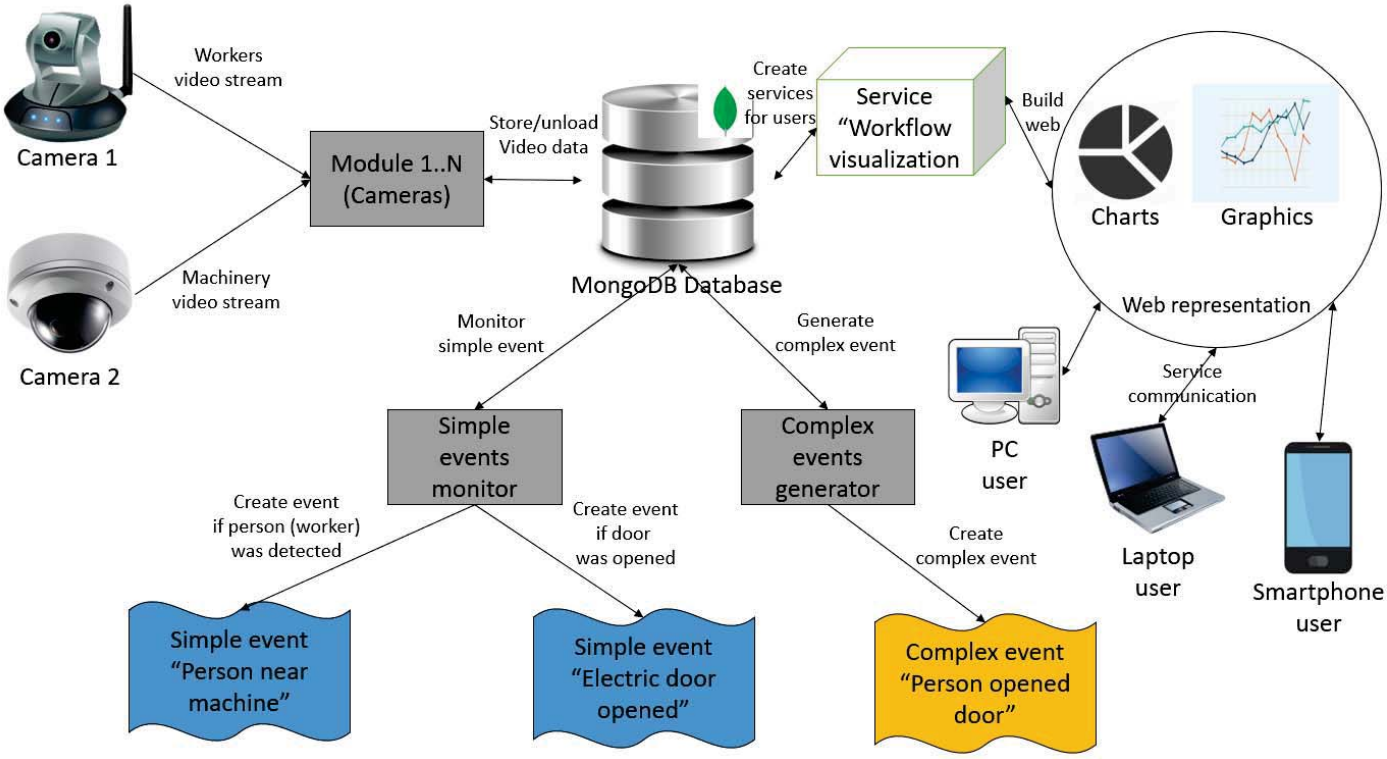

Fig. 3. Event-oriented model of video services

The presented model is quite flexible and can be supplemented with the following components: additional devices generating information about environment (e.g., sensors, accelerometers, sensors), interpreters (e.g., MySQL - using a relational database when a new service requires it; including the use of several servers in case of calculating a large number of video streams or when users need to join the server that is closest to its location), information nodes (when it is necessary to provide transfer of information at certain levels to other users and devices, including the form of alerts), representatives (e.g., web-charts, diagrams, bar graphs, tables and images).

In the case of a greater number of heterogeneous components, it becomes possible to monitor more simple events related to sensors. As a part of the IoT, a large number of sensors (including a smartphone as a sensor) will form more complex wireless sensor networks. Within such a network, it is possible to monitor various events related to sensors and video cameras. For example: an event of determining the distance from a static camera to a sensor (smartphone) using an image from a smartphone camera. In a similar way, search for the distance from the smartphone to other sensors can be organized. Advantages of the use of analytical sensor-camera events in edge-centric systems of IoT is the use of various information from different sources and their integration within one composite event for presentation as a service, involving mobile devices of system users, creating simple and composite events by users depending on the components and devices used, the variability of the technical devices used in the framework of various topologies (mainly from simple complex: point-to-point, star, tree, mesh).

V. Demo Implementation

Our demo implements the following functions.

- Connection to cameras module;
- Video service module;
- Person detection module;
- Event generation module.

The module for connecting to cameras is designed to connect to one or more cameras in order to receive a stream using standard OpenCV library tools. Camcorders located at the manufacturing facility to which you need to connect. To configure the module, the following installed libraries: Python (3+), multiprocessing, cv2 (OpenCV). Receiving video from cameras is carried out using the RTSP protocol. The module uses the File System (FS) to save data (connection to cameras, date of connection, IP addresses / camera names). The primary data processing scheme includes the following parameters: IP cameras, camera name, resolution (width and height), fps (frames per second, number of frames per second), a streaming codec. As a result of connecting to cameras, the user observes the camera image in real time on his device.

The video service module provides the ability to play video using the file system, a link to which is located in the MongoDB database. Video is played on the device (user or server). The module is configured using the VideoLan tool, implemented in Python using the python-vlc library. This module does not save data, but reproduces them. As the primary processing, you must specify the path to the file source

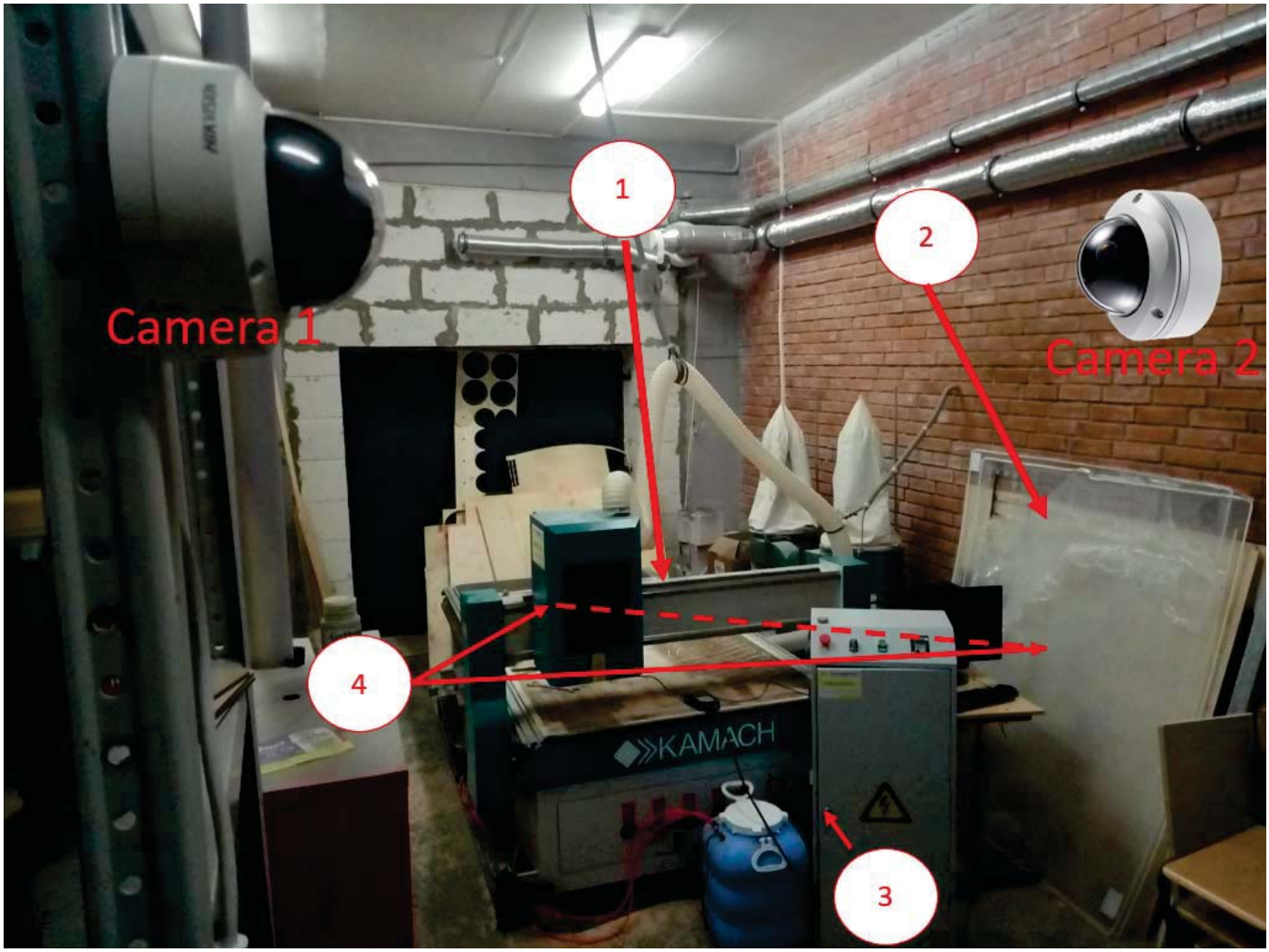

Fig. 4. Example event: a person appears close to production equipment

in the file system. As a result, you can view the recorded video, pause it or stop it. It is possible to play multiple videos in order. Functionality, in most cases, is similar to the implementation of a simple video player.

The purpose of the machine's person detection module is to verify the presence of an operator in the work area using recognition algorithms. Incoming image processing occurs using the trained Caffe neural network. A person (his silhouette) appearing in the frame is recognized by the module from the incoming camera image. To run the module, you must have installed libraries: Python 3.6+, OpenCV, Caffe. The video stream is saved using the OpenCV library, the log of human detection is saved using the logging module. As a result, a person recognition accuracy of 0.95 (95%) was achieved.

The event generation module creates an event from the camera using one of the modules and stores it in the MongoDB database. Interaction with other modules of the subsystem occurs (e.g., detection of a person at the machine). Depending on the initial parameters received from other modules, an event is generated that is written to the database. Events are generated using the language Python using the pymongo libraries (MongoClient), json. The following information is stored in the MongoDB database: time stamp (time of occurrence, that is, the end of the event generation), type (specific description of the event, for example, "A person appeared in the frame"), priority (importance level of the event), sender (unique identifier of the source information generation). At the initialization stage, the collection is generated "Configs" in the database, that is, special settings for connecting to cameras. After that, new events are created in the events collection. As a result of the module working in the database MongoDB data saved events.

For consideration, we took the task described, including in the previous paragraphs (the task of recognizing the actions of workers at the machine). Thus, we single out several actions that a worker can perform when he is near the machine. Event tracking is carried out using the appropriate modules in case the service is running. Any kind of recognition is created using neural networks. Examples are visualized in Fig. 4.

(1) No one near machine: event of the absence of a person in the frame, the module recognizes the image from (both Camera 1 and Camera 2 can be used). If it was

not possible to recognize the person, then an event is recorded;

(2) Person detected (person near machine or near machine area): event of the presence of a person in the frame (similar to the previous paragraph). In case of successful recognition of a person, an event is recorded;

(3) Electric cabinet opened: event of recognition of the opening of the door of the electric cabinet. Thus, for this service there are 2 states: the door is open, the door is closed. In case of successful recognition of an open door, an event is recorded;

(4) Counting distance between person and machine: event of calculating the current distance between a person and a machine. This event can only occur if the "Person detected" event occurs. The module calculates the distance between the recognized person and the machine.

To summarize the demo implementation, let us list the following programming technologies that were used in the demo development.

1) PyCharm: an integrated development environment for software programming for the most convenient way to write code;
2) Python3: a multifunctional programming language for applying the basic functions of working with data objects, filesystems, databases;
3) MongoDB: cross-platform document-oriented non-relational (NoSQL) database for storing objects in the form of events;
4) pymongo, MongoClient: Python libraries for working with the MongoDB database;
5) Neo4j: graphical database management system for semantics representation between events;
6) neo4j (GraphDatabase): Python language library (driver) for connecting and working with the Neo4j system;
7) py2neo (Graph), neomodel: Python library for working with the Neo4j system (analysis and generation of composite events).

VI. Conclusion

This paper considered an IoT-enabled class of advanced video services where many video capture devices can be used for video-analytics. For service development problem we discussed microservices system, event-driven model, and context-based mechanisms. The problem of data processing with multiple data streams is solved based on an edge-centric microservice-oriented system and its event-driven context-based model. The proof-of-the-concept demo implementation considers personnel recognition around a production equipment unit. We showed that the proposed concept model is feasible for applying in digital monitoring the smart manufacturing process.

Acknowledgment

The research was financially supported by the Ministry of Education and Science of Russia within project # 2.5124.2017/8.9 of the basic part of state research assignment for 2017–2019. The reported study was funded from Russian Fund for Basic Research according to research project # 19-07-01027. The results were implemented by the Government Program of Flagship University Development for Petrozavodsk State University in 2017–2021.

References

[1] D. Korzun, S. Balandin, and A. Gurtov, "Deployment of Smart Spaces in Internet of Things: Overview of the design challenges," in *Internet of Things, Smart Spaces, and Next Generation Networking*, ser. Lecture Notes in Computer Science, S. Balandin, S. Andreev, and Y. Koucheryavy, Eds., vol. 8121. Springer, Aug. 2013, pp. 48–59.

[2] G. Cristescu, R. Dobrescu, O. Chenaru, and G. Florea, "DEW: A new edge computing component for distributed dynamic networks," in *Proc. 2019 22nd Intl Conf. on Control Systems and Computer Science (CSCS)*, May 2019, pp. 547–551.

[3] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, *"Fog Computing: A Platform for Internet of Things and Analytics"*. Springer International Publishing, 2014, pp. 169–186.

[4] G. Ananthanarayanan, P. Bahl, P. Bod?k, K. Chintalapudi, M. Philipose, L. Ravindranath, and S. Sinha, "Real-time video analytics: The killer app for edge computing," *Computer*, vol. 50, no. 10, pp. 58–67, 2017.

[5] L. Cuimei, Q. Zhiliang, J. Nan, and W. Jianhua, "Human face detection algorithm via Haar cascade classifier combined with three additional classifiers," in *Proc. 2017 13th IEEE Intl Conf. on Electronic Measurement Instruments (ICEMI)*, Oct. 2017, pp. 483–487.

[6] N. A. Bazhenov and D. G. Korzun, "Use of everyday mobile video cameras in IoT applications," in *Proc. 22nd Conf. Open Innovations Association FRUCT*, May 2018, pp. 344–350.

[7] N. Bazhenov, D. Korzun, and S. Balandin, "Smartphone-oriented development of video data based services," in *Proc. 23rd Conf. Open Innovations Association FRUCT*, Nov. 2019, pp. 62–68.

[8] D. Korzun, A. Varfolomeyev, A. Shabaev, and V. Kuznetsov, "On dependability of smart applications within edge-centric and fog computing paradigms," in *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, May 2018, pp. 502–507.

[9] H. Chen, T. Finin, A. Joshi, L. Kagal, F. Perich, and D. Chakraborty, "Intelligent agents meet the semantic web in smart spaces," *IEEE Internet Computing*, vol. 8, pp. 69–79, November 2004.

[10] N. Bazhenov, A. Harkovchuk, S. Marchenkov, and D. Korzun, "A low-cost indoor service for human recognition," in *Proc. 24th Conf. Open Innovations Association FRUCT*, Apr. 2019, pp. 816–817.

[11] D. G. Korzun, S. I. Balandin, A. M. Kashevnik, A. V. Smirnov, and A. V. Gurtov, "Smart spaces-based application development: M3 architecture, design principles, use cases, and evaluation," *International Journal of Embedded and Real-Time Communication Systems*, vol. 8, no. 2, pp. 66–100, 2017.

[12] H. Zhang, N. Yang, Z. Xu, B. Tang, and H. Ma, "Microservice based video cloud platform with performance-aware service path selection," in *Proc. 2018 IEEE Intl Conf. on Web Services (ICWS)*, Jul. 2018, pp. 306–309, 8456365.

[13] W. Naiyapo and W. Jumpamule, "An event driven approach to create UML models," in *Proc. 22nd Intl Computer Science and Engineering Conference (ICSEC 2018)*, Nov. 2018, pp. 1–5, 8712621.

[14] J. Shao, X. Zhang, and Z. Cao, "Research on context-based instances selection of microservice," in *Proc. the 2nd Int'l Conf. on Computer Science and Application Engineering (CSAE '18)*. New York, NY, USA: ACM, 2018, pp. 1:1–1:5.

[15] A. Meigal, K. Prokhorov, N. Bazhenov, L. Gerasimova-Meigal, and D. Korzun, "Towards a personal at-home lab for motion video tracking in patients with Parkinson's disease," in *Proc. 21st Conf. Open Innovations Association FRUCT*, Nov. 2017, pp. 231–237.

[16] S. liu and X. Bu, "Performance modeling and assessment of unified video surveillance system based on ubiquitous sg-eiot," in *2019 IEEE International Conference on Energy Internet (ICEI)*, May 2019, pp. 238–243.

[17] S. Durga, S. Surya, and E. Daniel, "Smartmobicam: Towards a new paradigm for leveraging smartphone cameras and iaas cloud for smart city video surveillance," in *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, May 2018, pp. 1035–1038.

[18] D. Korzun, "Service formalism and architectural abstractions for smart space applications," in *Proc. 10th Central & Eastern European Software Engineering Conference in Russia (CEE-SECR 2014)*. ACM, Oct. 2014, pp. 19:1–19:7.

[19] E. Ovaska, T. S. Cinotti, and A. Toninelli, "The design principles and practices of interoperable smart spaces," in *Advanced Design Approaches to Emerging Software Systems: Principles, Methodology and Tools*, X. Liu and Y. Li, Eds. IGI Global, 2012, pp. 18–47.

[20] T. Sultana and K. A. Wahid, "Choice of application layer protocols for next generation video surveillance using internet of video things," *IEEE Access*, vol. 7, pp. 41 607–41 624, 2019.

[21] F. Montori, L. Bedogni, and L. Bononi, "A collaborative internet of things architecture for smart cities and environmental monitoring," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 592–605, April 2018.