CIT. Journal of Computing and Information Technology, Vol. 24, No. 4, December 2016, 349–368
doi: 10.20532/cit.2016.1002869

Developing Agent-Oriented Video Surveillance System through Agent-Oriented Methodology (AOM)

Cheah Wai Shiang[1], Bong Tien Onn[1], Fu Swee Tee[2], Muhammad Asyraf bin Khairuddin[1] and Msury Mahunnah[3]

[1]Faculty of Computer Science & IT, Universiti Malaysia Sarawak, Sarawak, Malaysia
[2]Faculty of Engineering, Computing and Science, Swinburne University Sarawak, Sarawak, Malaysia
[3]Faculty of Information Technology, Tallinn Technology University, Tallinn, Estonia

Agent-oriented methodology (AOM) is a comprehensive and unified agent methodology for agent-oriented software development. Although AOM is claimed to be able to cope with a complex system development, it is still not yet determined up to what extent this may be true. Therefore, it is vital to conduct an investigation to validate this methodology. This paper presents the adoption of AOM in developing an agent-oriented video surveillance system (VSS). An intruder handling scenario is designed and implemented through AOM. AOM provides an alternative method to engineer a distributed security system in a systematic manner. It presents the security system at a holistic view; provides a better conceptualization of agent-oriented security system and supports rapid prototyping as well as simulation of video surveillance system.

ACM CCS (2012) Classification: Computing methodologies → Artificial intelligence → Distributed artificial intelligence → Intelligent agents

Software and its engineering → Software notations and tools → System description languages → System modeling languages

Keywords: agent-oriented software engineering, video surveillance

1. Introduction

Agent technology has been used in building various domain specific applications. The agent paradigm introduces a software entity (e.g. agent) that is autonomous, proactive and able to interact with other agents for task accomplishment [1]. This kind of software supports complex applications like ambient intelligence, e-business, peer-to-peer, bio-informatics, negotiation [2] which demand the software to be robust, effective [3], co-operative to wide environments, customizable to support user needs, secure, and evolve over time to cope with changing requirements.

Until recently, the agent technology has been adopted in a range of areas including collaborative learning games [4], rural ICT [5], ubiquitous computing, e-commerce (business to business-B2B and business to client-B2C) [6], robotic [7], library [8], e-learning, manufacturing, logistic [9], environment, banking [10], construction [11], bioinformatics, accident management [12], power management [3], crisis management [13], sustainable software [14], mathematical model [15] and grid computing. For example, the agent technology is used to support the collaborative design environment among the project's participants in the construction application. It is used to facilitate decision support at various stages of construction project like engineering design, negotiation and so on. However, agent technology has not been widely adopted by the software community [16]. The reasons for the setbacks are the diversity of agent-oriented software engineering methodologies and the lack of maturity in some of the methodologies [17]. The agent methodologies are proposed to aid the agent developer with the introduction of technique, terminology, notation and guideline during the development of the agent system [18].

To date, about 30 agent-oriented methodologies have been introduced [19]. It is reported that some of the agent methodologies lack generality. They are focused on specific systems and agent architectures [20]. In addition, some

of the methodologies do not consist of sufficient detail to be of real use [21]. The variety of agent methodologies that have direct or indirect influences on the object-oriented methodologies can cause difficulty for the industrial developers in selecting the methodology [17]. This paper presents a detailed case study to validate agent-oriented methodology (AOM). Agent-oriented methodology (AOM) [22] is a comprehensive and unified agent methodology for agent-oriented software development. Although AOM is claimed to be able to cope with a complex system development, it is still not yet determined up to what extent this may be true. Therefore, it is vital to conduct an investigation to validate this methodology in order to promote the agent technology to a wider community.

Video surveillance is a complex system. The adoption of agent technology in video surveillance leads to several benefits. The agent paradigm supports decentralization, autonomy, fault tolerance and flexible [1], robustness, low coupling [23]. Hence, it is worth to research into the adoption of AOM in this domain.

The paper first covers the requirements of the video surveillance system (VSS) through HOMER, an elicitation method for AOM. Then, the conceptualized domain modelling of VSS is presented. This is followed by the elaboration of VSS design through platform independent design modelling. Finally, it demonstrates the transformation of the design models into JADE implementation. The main contribution of this paper is to validate the AOM through a case study of VSS development. This paper shows the feasibility and applicability of AOM to model a complex distributed system. In addition, the detailed modelling can be served as a guideline for developers in engineering a distributed security system, VSS.

Section 2 presents the survey on agent-based video surveillance systems. It covers the current practice in designing and developing agent-based surveillance systems and the background of agent-oriented methodology. An elaborated case study is presented in Section 3. An intruder handling scenario is highlighted and used for the rest of the discussion in this paper. Section 4 and Section 5 present the modelling process of intruder handling through AOM. Section 4 covers the requirement elicitation step in AOM. Meanwhile, Section 5 presents the details modelling of intruder handling through conceptual domain modelling, platform independent design and modelling and platform specific design and modelling. Section 6 presents the implementation of intruder handling in JADE, a multi-agent platform for developing agents in Java. A runtime aspect of agent based intruder handling is presented in this section. The paper is concluded in Section 7.

2. Related Works

Agent technology is adopted in video surveillance [1], [24], [25]. A multi-agent framework is introduced for the detection of suspicious activities in crowded scenes in a distributed multi-camera closed circuit TV (CCTV) network environment [1]. In [24], [26], agent coordination protocol and scheme are introduced to support agents-controlled camera. In this case, the agent behaves like human camera operator to reason and communicate on surveillance tasks. A multi-agent architecture is introduced to overcome the limitations of the current surveillance systems [25]. It has been reported that the current surveillance solutions suffer from the lack of flexibility and scalability. An empirical study is conducted and it validates the flexibility and scalability of the agent technology in distributed video surveillance [25]. From the survey, a software engineering approach (e.g. agent architecture, requirement engineering study etc) is adopted when designing and developing multi-agent video surveillance system. Although agent methodologies are introduced, there is not much addressing in designing and developing an agent-based video surveillance system. In line with the work to adopt agent methodology in agent-based video surveillance [24], [27], [28], [29], we adopt AOM in designing and developing a video surveillance system. The AOM is able to provide an alternative technique in modelling an agent-based surveillance system.

[27] presents the modelling of a surveillance system using the agent methodology named Ignenias. [23] provides a detailed description of VigilAgent methodology, which has been applied to modelling and implements multisensory surveillance systems. Finally, [28] presents the modelling and development of a person-following mobile robot application using Prometheus.

AOM is a methodology to model a complex socio-technical system. Sterling and Taveter combined the ROADMAP and AOR methodology [22] to produce a set of systematic methods, vocabularies and notations for conceptualizing socio-technical systems. It is compliant to model-driven development approach. Generally, the modelling process consists of conceptual independent modelling (CIM), platform independent design (PIM), and platform specific design (PSM). Specifically, the modelling process involves the design of goals, roles, interactions and domain knowledge models. This is followed by deciding the agent types, identifying the agents' knowledge, formulating interactions between agents and determining agents' behaviour. In brief, each agent models are described as follows.

Goal model. Goal model describes the purpose of the system (e.g. system functionality) at a higher level of abstraction. The notion of goal provides an overview of the functionalities that should be achieved by an agent system. Goals can be divided into sub-goals. Achieving a goal consumes resources and a goal is related to a particular role which indicates the actor or agent that is involved in achieving the goal [30].

Role model. A role model describes the roles involved within an organization. A role model is represented as a role schema, which consists of the following elements: role name, role description, responsibilities, and constraints. In an organization, people are assigned roles and positions to perform specific tasks. A position is required to subsume common tasks and subtasks under it.

Domain Model. The domain model represents the information that is handled by the system as a set of domain entities and the relationships between them.

Behaviour model. A behaviour model outlines the actions, reactions and responses of different types of agents [22]. It enables both proactive and reactive behaviour to be modelled. An agent achieves a goal by performing activities. The sequence of activities is modelled by means of control flows. A rule is the basic behaviour modelling construct. These rules are triggered either at the start of activities or by conditions that have been fulfilled, or an action event caused by external agents. The execution of a particular activity is modelled by triggering a rule to update the agent's mental state and/or to send messages or to perform an action. An agent is proactive if its mental state is capable of triggering an activity. An agent reacts due to the perception received through a communication action or physical action by a human agent. A communication action involves exchanging messages between agents. A physical action involves a direct command by a human, which normally occurs through a graphical user interface.

Interaction Model. An interaction model models the social influence between agents. In this model, interactions between agents are represented through message passing. The interaction model models the content and the order of the messages to be exchanged.

Scenario model. A scenario model consists of activities to describe how agents carry out certain roles to achieve a particular goal.

Knowledge model. A knowledge model represents the details of the information types that are required by agents to solve a particular problem. A knowledge model specifies informational object types, predicate types, relationships between objects, and private and shared objects.

3. Elaborated Case Study – Distributed Video Surveillance System – Intruder Handling Scenario

Residential home burglary is a serious social problem in Malaysia. Thieves like to break into houses when the house is unattended, especially during daytime. Thieves are intruders who break into the house through climbing over the wall, breaking the lock or door etc. In order to prevent thieves and home break-ins, CCTV is installed to monitor any intruders who are entering the house. The CCTV detects any unfamiliar faces or intruders, then sends an intrusion alert to the house owner as well as to the police station. The police officer will wait for the confirmation of the detected intruder from the house owner and further notify the on duty police patrol officer if needed. In addition, the police officer will contact the house owner for further action. In the following section, we present modelling of the elaborated case study through AOM. This involves elicitation requirement process, and agent modelling (CIM, PIM and PSM).

4. Requirement Elicitation of Agent-Oriented VSS

A Human-Oriented Method for Eliciting Requirement, HOMER [31] is used to elicit requirements for an agent system. HOMER is based on the organization metaphor in "hiring a staff" to collect and identify the requirements of a given problem. The elicitation questions are shown in Table 1. In this case, software engineers elicit and reason on various considerations in recruiting staffs to solve a problem. From the answers gathered, the discovered requirements can be easily translated into the goal model, role model, organization model and domain model based on the guidelines proposed by [32]. In other words, the questions described in HOMER have a direct realization in the ROADMAP goal model and role schema.

Table 1 shows the elicitation answers for intruder handling scenario. Several stakeholders are involved in working on intruder handling scenario. They are the security manager, security personnel, family members (e.g. house owner and family members), visitors and neighbors. Each stakeholder has their own role and responsibilities. For example, the security manager has the responsibilities to observe the changes of environment, alert authorized personnel or house owner upon an intruder detected, by sending a message, as well as by continuous monitoring and tracking the intruder. The information tabulated in Table 1 is used to furnish the agent modelling process as elaborated in the following section.

5. Agent Modelling

Within the AOM, three phases are involved in modelling a multi-agent system. They are conceptual independent modelling (CIM), platform independent design and modelling (PIM) and platform specific design and modelling (PSM). Section 5.1 presents the conceptual domain modelling of intruder handling. Section 5.2 presents the platform independent design and modelling of intruder handling, and finally Section 5.3 presents the JADE-based design and modelling of intruder handling. All the models are presented in the appendix A.

5.1 Conceptual Independent Modelling

The CIM level reflects the early requirements and analysis for an agent-oriented system. Requirements analysis is a common stage among various agent-oriented methodologies, which is used to model agent system at a higher level of abstraction as well as to understand and analyse the requirements for developing an agent system. It is intended to present an overview of the system and determine its functionalities. Ignoring it can lead to misunderstanding the system in design [33]. Furthermore, the analysis normally involves activities that provide the context in which the system is to be designed [33]. The conceptual domain modelling of intruder handling is presented at Stage I: modelling goal and role, Stage II: modelling role, Stage III: modelling interaction frame and Stage IV: modelling domain knowledge.

Stage I: modelling goal and role. Figure 1 shows the goal model to handle an intruder. The notion of goal provides an overview of the functionalities that should be achieved by an agent system. Goals can be divided into sub-goals. In addition to functional goals, there are quality goals that represent non-functional requirements for the system. Achieving a goal consumes resources and a goal is related to a particular role which indicates the actor or agent that is involved in achieving the goal [22]. A role is the position played by an individual in an organization. The agent is a software entity that is situated in an environment.

In Figure 1, quality goals are incorporated in the main goal, indicating that the response needs to be appropriate and timely. Three roles are required to achieve the overall goal: Security Manager, Intruder, and Evaluator. The overall goal has been decomposed into five sub-goals: "Detection", "Identify", "Respond", "Scheduling" and "Evaluate". Two quality goals, "Timely detection" and "Accurate identification" are added to the "Detection" and "Identify" sub-goals respectively. The "Respond" sub-goal in turn has been expanded into two sub-goals: "Greeting" and "Communication". The "Communication" sub-goal is further divided into eight sub-goals. To accomplish these sub-goals, additional roles such as police, visitor, security guard, volunteer person (RELA), insurance agent, family members, immediate neighbour and owner are involved.

Table 1. Elicitation questions and answers for intruder handling.

From HOMER's question	Answer(s)
1. If you were to hire more staff to handle your current problem, which positions would you need to fill?	1. Security manager 2. Security personnel 3. Family members 4. Visitors 5. Neighbours
2. For each position, we need to collect a "job description": (a) What is the purpose of this position? What aspects of the problem will this position solve or partially solve?	1. Security manager • Observe the environment change. • Alert authorized personnel/person registered in the system of environment change. • Security manager will detect unauthorized person and send an alert message to the house owner, and other registered personnel in the system. • Send location of the threat to the security personnel for further action. 2. Security personnel • Respond immediately to home security threat. • Go to the location immediately after receiving alert message. 3. Family members, visitors, neighbours • Take immediate precaution and stay away from security threat location.
2. For each position, we need to collect a "job description": (b) What tasks will commonly be required?	1. Security manager • Send an alert message and location to security personnel, home owner, family members, visitors and neighbors. 2. Security personnel • Go to alert location and investigate the threat. 3. Family members, Visitors, Neighbors • Stay away from threat location.
2(c). For each task above: *i*. What sub-tasks make up this task?	1. Communication
2(c). For each task above: *ii*. What constraints are there for this task?	Messages not delivered.
2(d). Which system/people in the company does this person rely upon?	1. Security manager – relies on intruder detection system(services) 2. Security personnel – rely on security manager 3. Family members, visitors, neighbors – rely on security manager
2(e). Who else in the company relies upon this person?	None
2(f). What knowledge does this person require to perform his tasks correctly?	1. Security manager – list of personnel and personal information in the company
2(g). What resources, existing and new, are required by this person in fulfilling his position?	1. Image and contact number.
3. What code of behaviour must be observed by all of your employees? (a) Are there other codes of behaviour for certain positions, and what are they?	None
4. What other rules and regulations must your company adhere to?	None

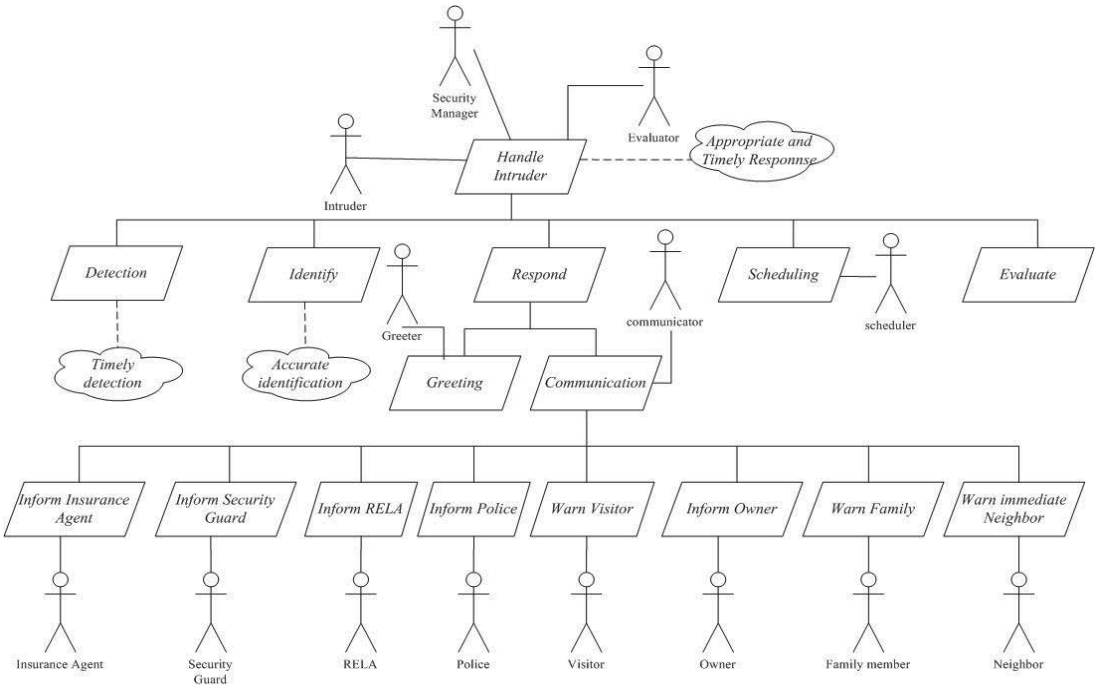

Figure 1. The goal model for intruder handling.

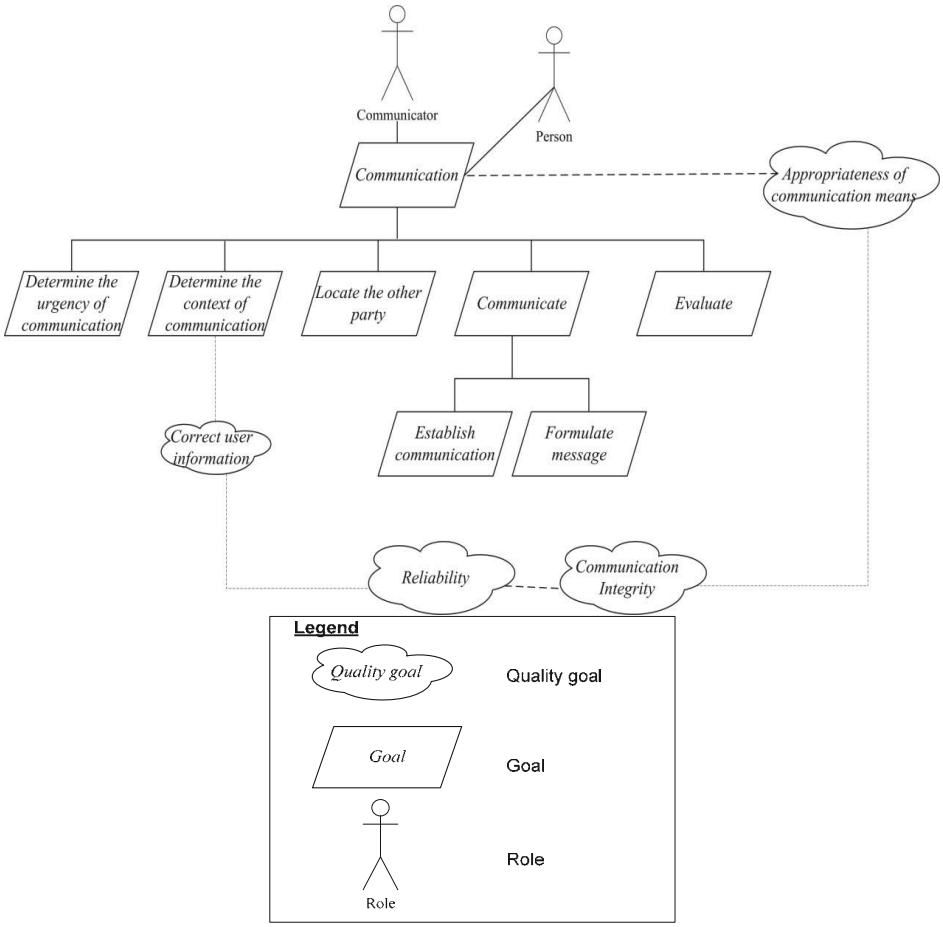

Figure 2. Goal model for communication.

Table 2. Role model for security manager.

Role Name	Security Manager
Description	To identify and respond to an intruder detected in the house.
Responsibilities	Detect the occurrence of a person in the environment. Take an image of the person. Compare the image against the database of known people. Verify with the owner the identity of the person detected in the environment. Contact the police and send the capture image to them. Notify each visitor expected that day to stay away. Inform the owner that the police are on their way and the visitors have been warned not to enter the house.
Constraints	System needs to be provided with photo of the owner, family members, and visitors in advance. The camera needs to be installed in a place where a subject is easily detected and image can be captured clearly. The owner, family members, and visitors must have a communication device in order to receive messages.

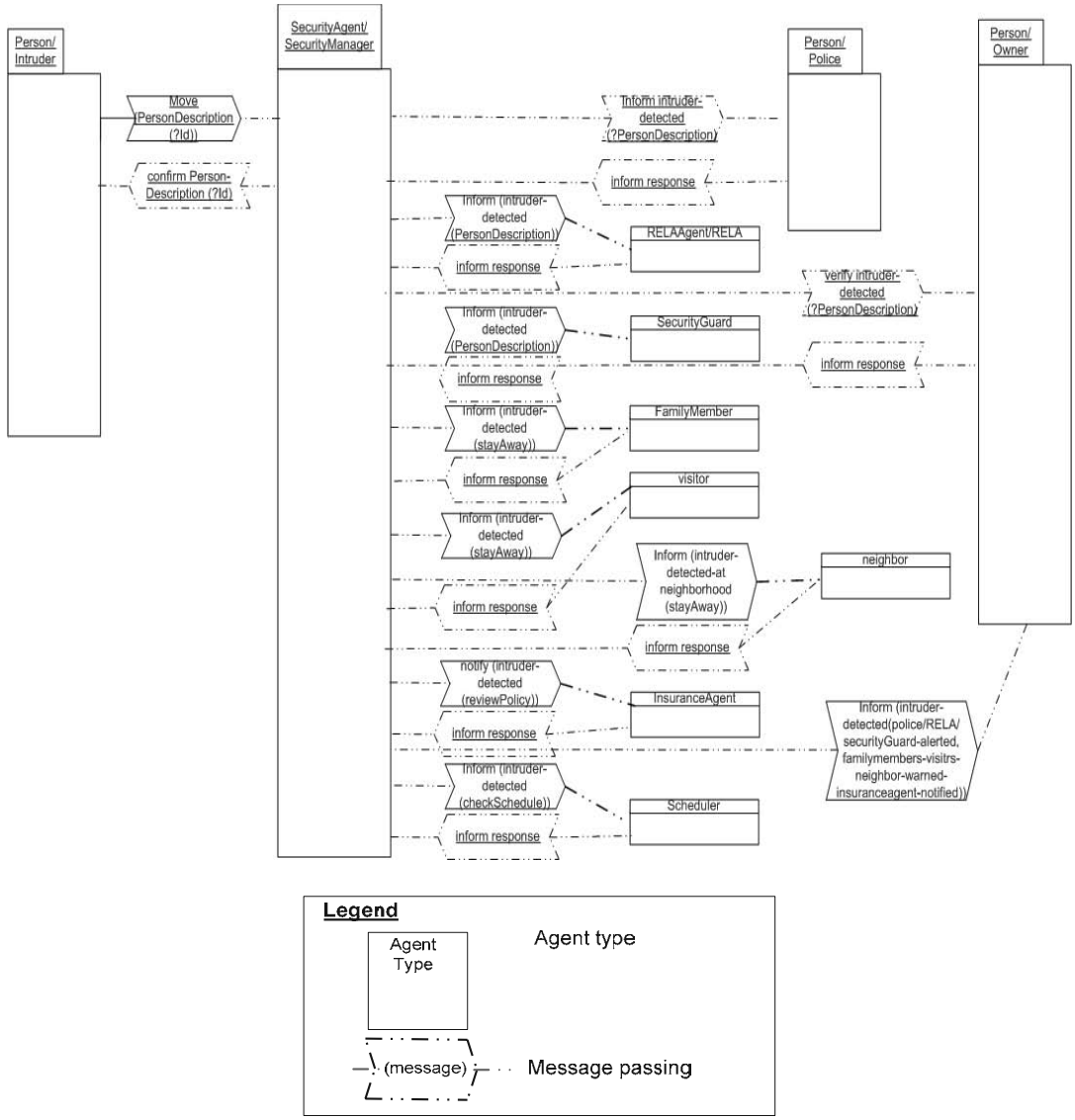

Figure 3. Interaction-frame diagram for intruder handling.

In general, the security manager will first start the face recognition service to monitor the changes of the environment, which includes extra object detected in the environment. Once the changes are identified, the security manager will act as a communicator to establish communication with other roles/persons as shown in Figure 2 with the intention to alert them. The "Person" modelled in this system refers to the visitor, owner, family members, police, security guard, RELA, neighbor, insurance agent and any other relevant personnel.

Stage II: modelling roles. Various roles are required in achieving the goals within the intruder handling. To exemplify it, we present a role model of security manager. Table 2 describes the role model for the security manager. The model presents its responsibilities and constraints that may restrain the operations while serving these responsibilities.

Stage III: modelling interaction. Interaction frame diagram is used to generalize the types of action events that occur between agents and other types of agents. The interaction-frame diagram models the possible interactions between agents for both physical interactions and communication. The order in which the agents interact or the conditions under which one or another alternative interaction occur are not mentioned. Figure 3 presents the interaction-frame diagram for intruder handling.

After determining the agent types, we can now capture the interactions between agents of those types with the interaction model through an in-

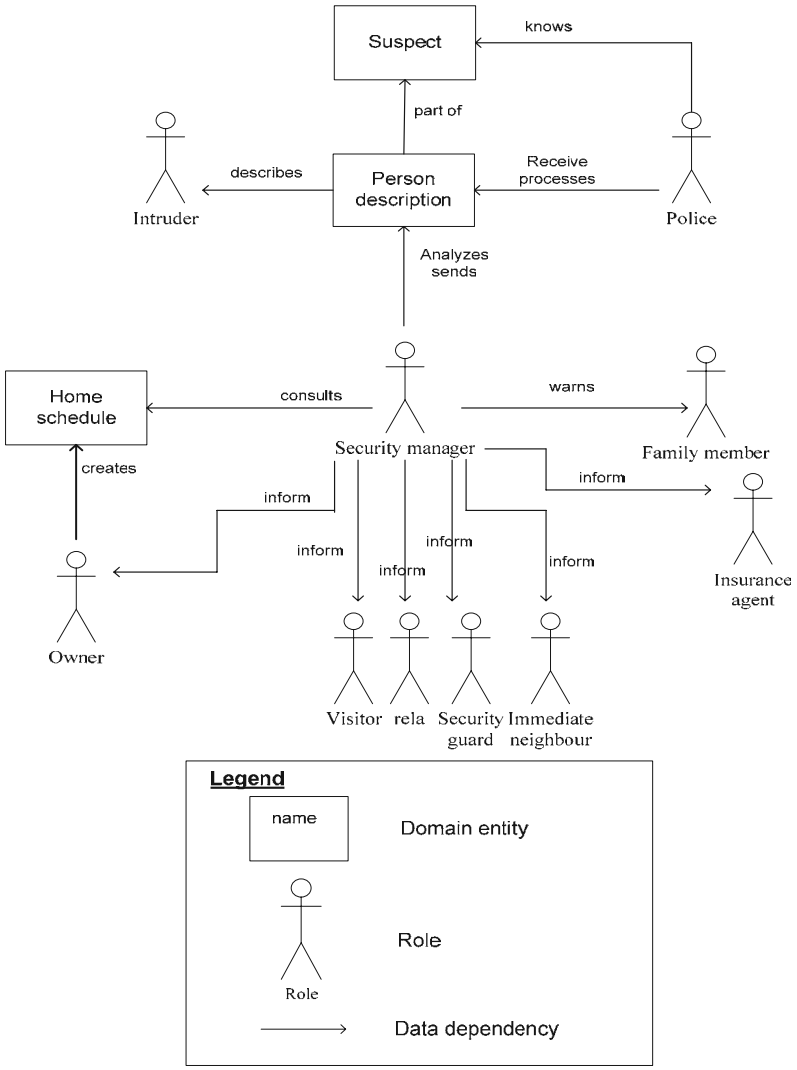

Figure 4. The domain model for intruder handling.

teraction-frame diagram. Interactions can be extracted from the responsibilities documented in the role schemas. Figure 3 consists of two interaction frames: between the agent of a person (intruder) and the security manager, and between the agent of the security manager and the persons (house owner, police, security guard, RELA, insurance agent, family members, immediate neighbor and visitor(s)). Messages in interaction frames have two modalities: "inform" and "confirm". A message of the "inform" modality serves to alert another agent about an incident. For example, when an unknown person is detected, the security manager will send a message to inform the police agent about that. The police agent will then confirm the message and respond to the security manager accordingly.

Stage IV: modelling domain knowledge. Figure 4 shows the domain model that describes the relationship between different entities or roles when an event has transpired. The PersonDescription domain entity caches the visual information captured about the person detected by the security manager. The security manager compares and matches the captured image against his database. The person is regarded as an intruder if the system fails to identify him. In this case, the person's description will be forwarded to the police, who may be able to identify the concrete suspect. To identify those who are authorized to be in the house such as plumbers or electricians, the security manager will consult the house schedule in the HouseSchedule domain entity. HouseSchedule domain entity stores the start and end times of various activities that are taking place in the house, such as visits by friends and colleagues, family celebrations, and calls by service people. The schedule is created by the owner.

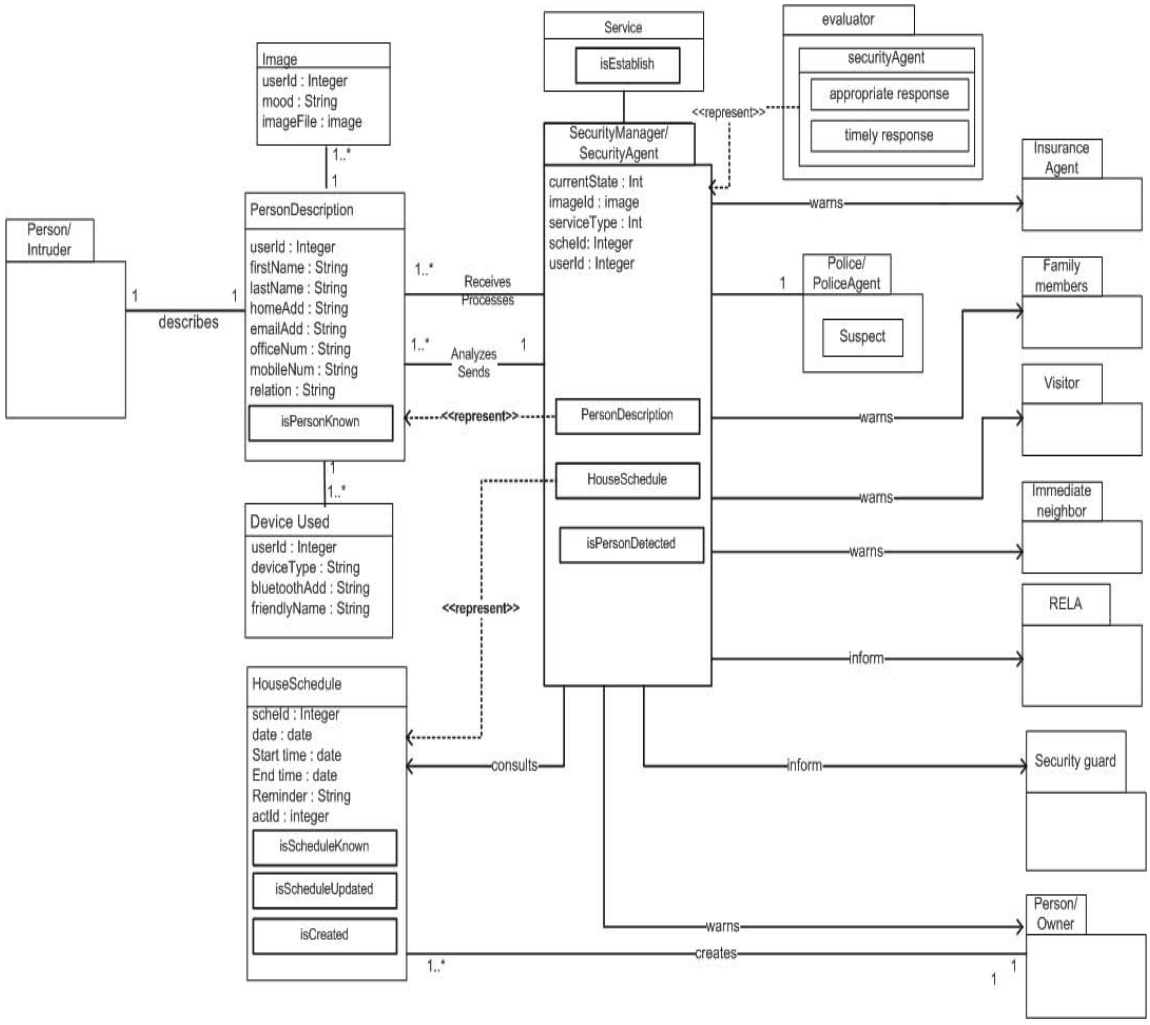

Figure 5. Knowledge model for intruder handling.

5.2 Platform Independent Design and Modelling (PIM) for Intruder Detection

The PIM level reflects the design of an agent system. Designing an agent-oriented system involves various design activities. An agent consumes resources. To reflect that, domain conceptualization is a design activity for building the ontology that is required for agents. This involves activities to identify the concepts, their attributes and predicates, and relations between the concepts. Design of an agent's internal structure involves activities to determine the arrangement of information flow, decision control, and the arrangement of inference steps for agent reasoning and action activation. In addition, interactions between agents are designed by detailing communication states of agents and presenting the syntax (e.g. messages and parameters) for agent communication.

The PIM begins with stage V and stage VI. After deciding agents' type in stage V, the next step is to convert the domain model into knowledge model. To simplify the naming convention, we assume that there is a one-to-one direct mapping from role to agent type. In other words, the role of security manager is mapped to the agent type of "securityManagerAgent". With this practice, we can focus on the knowledge model next.

Stage VI: modelling agent knowledge. The knowledge model for the intruder handling system is presented in the agent diagram as shown in Figure 5. The knowledge model is constructed from the domain model in Figure 4. The knowledge model further elaborates and expands the knowledge entities that are used by the agents. The *PersonDescription* object type is shared between agents of the *SecurityAgent* and *PoliceAgent* types. The *HouseSchedule* contains the attributes to store useful data such as time, activities and reminder. It is owned privately by the SecurityManagerAgent. SecurityManagerAgent will execute the face recognition functions, engage the communication services and house schedule services. To begin

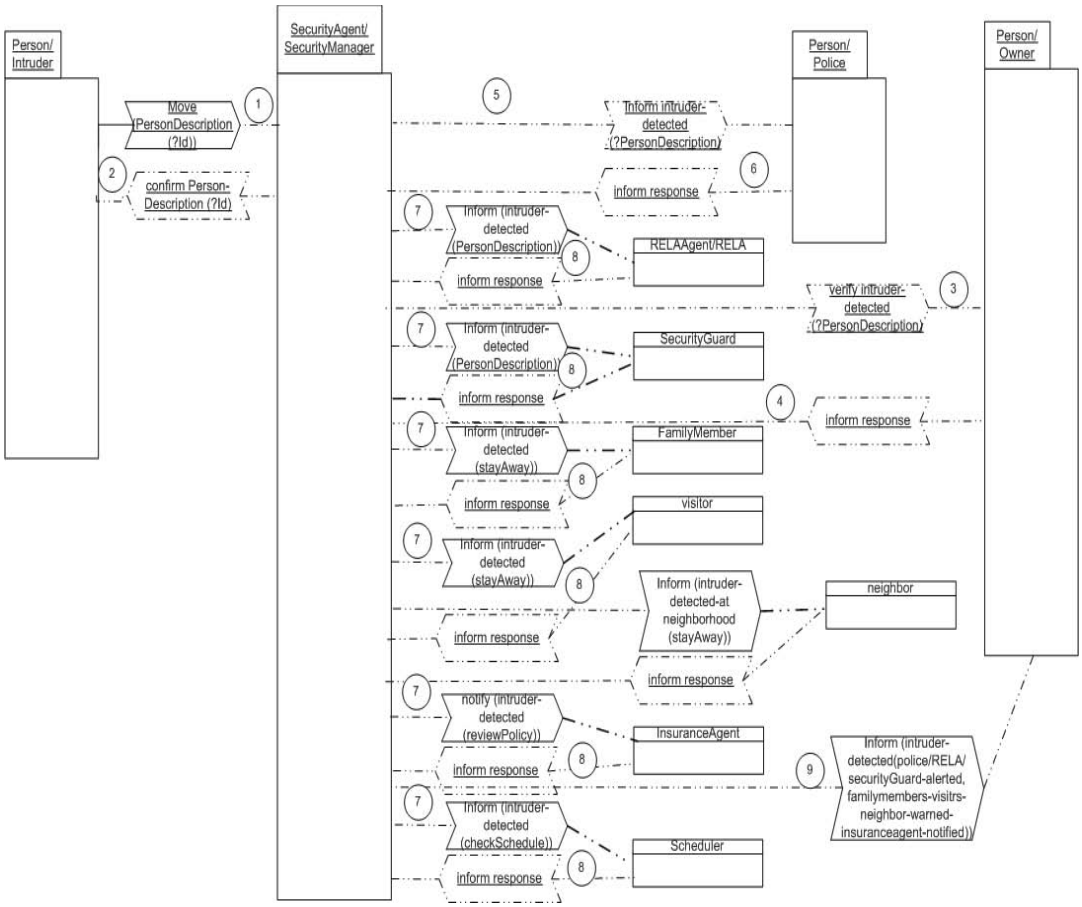

Figure 6. Interaction sequence diagram from intruder handling.

the intruder detection process, face recognition function is deployed to detect the changes of the environment, for instance, in the living room of a house. When a change in the environment is detected with the presence of an unidentified object or person, the securityAgent is notified and the communication service is established.

Stage VII: modelling interaction between agents. The sequence of interactions between various types of agents in handling intruder is modelled in Figure 6. The intruder-handling function is activated upon the detection of object changes or movement by the sensors. This will trigger the face recognition function to capture the image to determine the threat encountered. If the object is unidentifiable, the system will send an 'intruder' event notification to the SecurityManagerAgent. Then, the SecurityManagerAgent will in turn send an "intruder detected" message to the policeAgent, RELA, SecurityGuard, the house owner and alert the visitors, neighbor, family members, in order to stay away from the threat. To ensure a quick response, the policeAgent will automatically assign any police within the vicinity to the scene to conduct further investigation. The RELA and SecurityGuard will also be sent to the location to assist in the investigation. During this incident, the policeAgent will commence communication with the house owner to update him.

Stage VIII: modelling scenario model. Stage VIII presents a more detailed list of actions to handle the scenario. This artifact will be used in modelling the agent's behaviour using agent-oriented relationship (AOR) behavioural diagram. A scenario model consists of a sequence of steps labelled with its type, name, role, description, and the data it accesses. Each step represents a goal, action, percept, or sub-scenario. A sample scenario for handling intruder is introduced in Table 3. The start of the step is shown in the main scenario model and the end of the step is the last step of sub-scenario. When an object/person enters the space within the camera's view, the detector will capture the image of the object/person (scenario 2) and then analyze or identify it using the face recognition program (scenarios 3, 4, and 5). It then compares the captured image against the database to identify the person. If the person is unrecognizable, the security manager will notify the house owner of the unknown person in the house. At the same time, the security manager will send a message to the enforcement officer or agent to alert them (scenario 7), and warn the family members, visitors and immediate neighbor to stay away (scenario 8). The security manager will then update the house owner on the actions taken (scenario 9).

Stage VIII: modelling agent behaviour. Figure 7 shows a simplified version of the behavioural model for intruder handling. The outermost activity is started by rule R1, which is triggered by an action event labelled as *move (?PersonDescription)*. This modelling construct represents a physical movement of an object or person detected by the security agent in the form of event. Precision of the sensors is not of a concern at this stage of the system engineering process. Rule R1 also creates an instance of the *PersonDescription* object type within the security agent to store important data of the person. "Detect person" activity starts an "Identify intruder" sub-activity that triggers rule R2. This rule verifies the identity of the person through *isKnown(PersonDescription)* function. If the person is unidentifiable, the activity "Respond" is executed. The "Respond" activity consists of a series of sub-activities which include alerting the relevant parties. The activity types modelled in the behavioural model should correspond to the goals illustrated in the goal model in Figure 1. This implies that an activity should achieve the goal set at the preliminary design stage. For example, the "Respond" activity should effectuate the actions to achieve the "Respond" goal. R3 and R4 are reaction rules in Figure 7 that respond to the sub-activities by initiating communication through messages sent to the relevant recipients.

Progressing forward from conceptual models in PIM to more concrete models of PSM, next section will demonstrate transformation and derivation of the programming constructs based on the models presented in this section.

5.3. Platform Specific Modelling and Design (PSM) of Intruder Detection Scenario

The PSM layer is the lowest level of the system design, moving towards implementation. This level provides design details that "specify how the system is to be implemented in a specific platform, architecture, and tool or programming

Table 3. The scenario model for intruder handling.

SCENARIO 1					
Goal	Handling Intruder				
Initiator	Security Manager				
Trigger	Intruder				
Failure	Home break-in				
DESCRIPTION					
Condition	Step	Activity	Agent types and roles	Resources	Quality goals
When an object/ person enters a space that is within the camera's view	1	Person walks into a room. The system detects that there are changes in the environment. (Scenario 2)	Detector/Person Detector/Person	Camera/ PC/internet	Timely detection
	2	The system identifies the person's presence in the environment. (Scenario 3, 4, or 5)	Identifier/ Detector/Person		
	3	The system checks the house schedule. (Scenario 6)	scheduler		
	4	Take an image of the person.	Security Manager		
	5	Compare the image against the database of known persons.	Security Manager		
	6	Inform the owner there is an unknown person in the environment.			
	7	Contact the law enforcement officer and send the image to them. (Scenario 7)	Communicator/ Person		
	8	Send a stay away message to family members, visitors and immediate neighbour. (Scenario 8)	Communicator/ Person		
	9	Inform the owner that the police and other security officers are on their way and that their family members, visitor and neighbour had been warned. (Scenario 9)			

language" [22]. It facilitates conversion from the PIM models to derive the skeleton programs or program templates that reflect the structure of the system. The design model is transformed from the PIM level based on the guidelines proposed by the author [32].

Stage IX: JADE specific information model. Figure 8 shows the snippets of JADE specific model for PersonDescription information type that was modelled in Figure 5 as the knowledge model of intruder handling. As mentioned before, the agent types IntruderHandler, Police, Owner, Visitor, Family Members, Security guard, RELA, Neighbour, and Insurance agent are represented as the respective agent types of JADE. The shared object type IntruderDescription and the private object type Person are implemented as the corresponding Java object types. The predicate isKnown is converted to a method attached to the object type Person. The information type of PersonDescription is transformed into the JADE object class while the attributes of the information type are declared

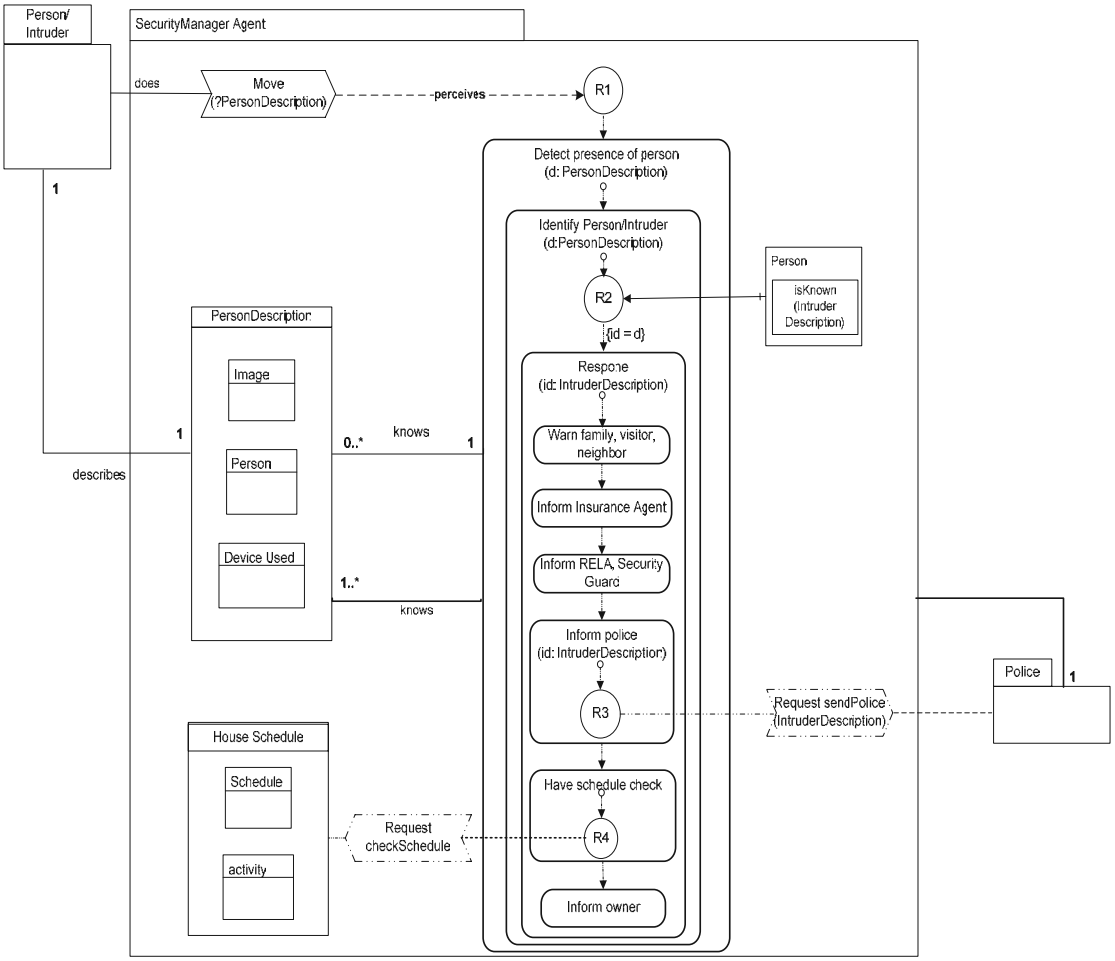

Figure 7. The behaviour model for the intruder-handling.

```
import jade.content.*;
public class PersonDescription implements Concept {
private int Userid;
private String firstName;
private String lastName;
private String homeAdd;
public int getUserId() {
return Userid;
  }
public String getfirstName() {
return firstName;
  }
public String getlastName() {
return lastName;
  }
//...others information model
}
```

Figure 8. JADE specific information model for part of the information type of PersonDescription.

as String data type. The relationship of the PersonDescription linked to the image and device used will be implemented with JADE object class such as Image and DeviceUsed. Figure 9 shows part of the face recognition coding that captures image, analyzes it and informs the SecurityManagerAgent when an unknown person is detected. It forms part of the service model in JADE.

Stage XI: JADE specific interaction model. Figure 10 shows a JADE specific interaction model, which is implemented as intruder handling ontology. It models the ontology that is shared among the agents during agent interaction by registering the type and properties of the entities. The ontology scheme consists of an action schema and an information type schema to specify the action type and concepts involved while handling intruder detection. The action schema models the CreatePerson action to create a new person object with detail parameters of information types as specified in the PersonDescription and the domain type. The object members that are set to mandatory indicate that all of the attributes must be set as compulsory during the agent communication. The JADE specific interaction model is a transformation of both the interaction model and knowledge model that are presented in Figures 8 and 9 respectively.

Stage XII: JADE specific behavioural model. Figure 11 partially shows PSM behavioural model in respond to intruder detection. Once an intruder is detected by the face recognition service, the SecurityManagerAgent will send an informed message labeled as "Intruder Alert" to all agents registered on JADE Yellow Pages.

6. Implementation of Intruder Handling Scenario

We presented the modelling of intruder handling in the Section 5. This section presents the implementation of intruder handling in JADE, a multi-agent programming platform. Figure 12 shows part of the physical distribution and agents of the VSS. All the nodes are connected wirelessly with a static IP for face recognition service (192.168.1.50), intruder detection system (IDS) Main Server (192.168.1.5), Police Main Server (192.168.1.4) and dynamic IP for agents involved in the scenario.

A webcam is connected to the face recognition system. An IDS Main_Server will act as context tier that performs all the work of detecting changes in the environment and interpreting the data captured to determine the context information by securityAgent. When a person enters the vicinity under scrutiny, the image recognizer

```
//some opencv human detection code
void FrameGrabber_Standard(object sender, EventArgs e){
//Get the current frame form capture device
currentFrame = grabber.QueryFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
//Convert it to Grayscale
if (currentFrame != null){
gray_frame = currentFrame.Convert<Gray, Byte>();
//Face Detector
MCvAvgComp[][] facesDetected = gray_frame.DetectHaarCascade(Face, 1.2, 10,
Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING, new Size(50, 50));
//Action for each element detected...
//Draw the label for each face detected and recognized
//add unknown face
ActivateAgent();
//..other processing code
```

Figure 9. Service model for object detection in OpenCV and the sending of intruder detection event to SecurityAgent.

```
package ontologies;
import jade.content.onto.*;
import jade.content.schema.*;

public class IntruderOntology extends Ontology implements IntruderVocabulary {
        private static final long serialVersionUID = 1L;
// ---------->The name identifying this ontology
public static final String ONTOLOGY_NAME = "Intruder-Ontology";
 // ---------->The singleton instance of this ontology
private static Ontology instance = new IntruderOntology();
  // ----------> Method to access the singleton ontology object
public static Ontology getInstance() { return instance; }
  // Private constructor
private IntruderOntology() {
super(ONTOLOGY_NAME, BasicOntology.getInstance());
try {
// ------- Add Concepts
// Person
ConceptSchema cs = new ConceptSchema(PERSON);
add(cs, UserProfile.class);
cs.add(PERSON_ID, (PrimitiveSchema) getSchema(BasicOntology.INTEGER),
ObjectSchema.MANDATORY);
cs.add(PERSON_FIRST_NAME, (PrimitiveSchema) getSchema(BasicOntology.STRING),
ObjectSchema.MANDATORY);
// House Schedule
add(cs = new ConceptSchema(HOUSE_SCHEDULE), HouseSchedule.class);
cs.add(HOUSE_SCHEDULE_ID, (PrimitiveSchema) getSchema(BasicOntology.INTEGER),
// ------- Add AgentActions
catch (OntologyException oe) {
    }
  }
}
```

Figure 10. JADE specific information model for the intruder handling ontology.

(Web camera) detects the presence of a person and captures his face. The face recognition service extracts the data from the image and sends the contextual information pertaining to the person detected to the securityAgent. Once the person is identified as an intruder, the securityAgent sends a message to the ownerAgent to further validate the identity of the person. When the ownerAgent is unable to recognize the person, the securityAgent will notify the policeManagerAgent and securityGuardAgent or RELA of the intrusion. Thereafter, the securityAgent contacts the scheduled visitorAgent, family members, and neighbourAgent to warn them to stay away. Meanwhile, the policeManagerAgent assigns the nearest police officer to the scene by comparing the distance in between the respective police officer with the scene. Figure 13 presents the communication between securityManagerAgent, visitorAgent, familyAgent and RelaOfficerAgent during the intructor handling. Meanwhile, Figure 14 presents the sample of communication between policeManagerAgent and respective policeAgent.

7. Conclusion

The main contribution of this paper is to introduce AOM for designing and developing an agent-oriented video surveillance system. It attempts to validate the feasibility of this metho-

dology in developing a complex socio-technical system in a qualitative manner. With AOM, we can engineer a distributed video surveillance system in a systematic manner. The comprehensive elaboration and decomposition of the models from different perspectives can serve as a guideline for novice developer during the development of multi-agent system. With AOM, there is a possibility to reuse the models in maintaining the VSS system. In addition, those models can be reused in other similar kind of projects in which more investigation is needed in the near future. In addition, the integration of AOM models with other models is worth to explore. This is because some of the AOM models are lacking explicit expression. For example, the goal model has failed to model the goal dependency in details. On the other hand,

```
import jade.core.Agent;
import jade.core.behaviours.CyclicBehaviour;
import jade.domain.AMSService;
import jade.domain.FIPAAgentManagement.AMSAgentDescription;
import jade.domain.FIPAAgentManagement.SearchConstraints;
import jade.lang.acl.ACLMessage;
public class SecurityManagerAgent extends Agent{
private static final long serialVersionUID = 1L;
protected void setup() {
AMSAgentDescription [] agents = null;
try {
  SearchConstraints c = new SearchConstraints();
  c.setMaxResults (new Long(-1));
  agents = AMSService.search( this, new AMSAgentDescription (), c );
}
catch (Exception e) {
System.out.println( "Problem searching AMS: " + e );
}
//once the intruder detection event is trigger, the securityAgent will send a messages to others
ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
msg.setContent( "Intruder Alert" );

for (int i=0; i<agents.length;i++)
msg.addReceiver( agents[i].getName() );
send(msg);
addBehaviour(new CyclicBehaviour(this){
private static final long serialVersionUID = 1L;
        public void action() {
        ACLMessage msg= receive();
        if (msg!=null)
        System.out.println( "== " + " -> " +  msg.getContent() + " from "+  msg.getSender().getName() );
        msg=null;
 }
});
}
}
```

Figure 11. JADE behaviour model for securityAgent upon the detected intruder.

the behaviour model has failed to present the overall capability of an individual agent. Based on our experience, presenting the overall goal and goal dependency and capabilities of the agent are needed to ease debugging and discussion with the stakeholders, in which more to explore in future. Also, more example can be explored like abnormal human behaviour as stated in [36].

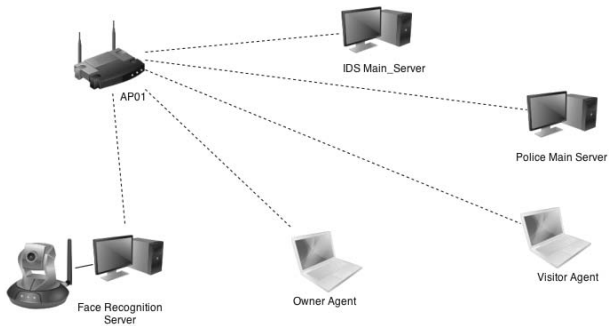

Figure 12. Simulation of intruder handling.

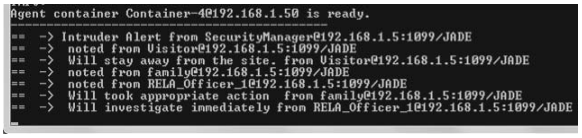

Figure 13. Communication between securityManagerAgent, visitorAgent, familyAgent and RelaOfficerAgent.

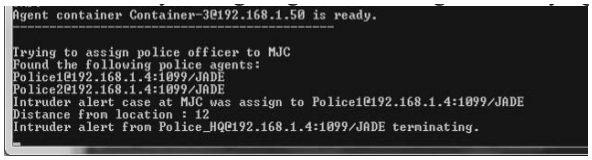

Figure 14. Communication between policeManagerAgent with the respective policeAgent.

References

[1] N. Ejaz *et al.*, "A Collaborative Multi-Agent Framework for Abnormal Activity Detection in Crowded Areas", in *Int. Journal of Innovative Computing, Information and Control*, vol. 8, no.6, 2012.

[2] C. Wai Shiang *et al.*, "Software Aagent Negotiation Development: An Experience Report", *Proc. of the Int. Conf. on Intelligent Systems Design and Applications*, pp. 881–886, 2006. http://dx.doi.org/10.1109/ISDA.2006.253728

[3] X. Liu and K. Wang, "Active Power Control Simulation Platform Research of Wind Farm Based on Multi-Agent" in *MATEC Web of Conferences*, vol. 22, 2015, EDP Sciences. http://dx.doi.org/10.1051/matecconf/20152202004

[4] C. Wai Shiang *et al.*, "Designing a Shared Single Display Education Application through Interactive Patterns", in *Journal of Software Engineering and Applications*, vol. 7, no. 13, pp. 1074, 2014. http://dx.doi.org/10.4236/jsea.2014.713095

[5] C. Wai Shiang *et al.*, "Long Lamai Community ICT4D E-Commerce System Modelling: An Agent-Oriented Role-Based Approach", in *The Electronic Journal of Information Systems in Developing Countries*, vol. 75, 2016.

[6] D. Chen *et al.*, "An Agent-Based Model for Consumer-to-Business Electronic Commerce", in *Expert Systems With Applications*, vol. 34, no. 1, pp. 469–481, 2008. http://dx.doi.org/10.1016/j.eswa.2006.09.020

[7] A. J. C. Trappey *et al.*, "Development of an Intelligent Agent System for Collaborative Mold Production with RFID Technology", in *Robotics and Computer Integrated Manufacturing*, vol. 25, no. 1, pp. 42–56, 2009. http://dx.doi.org/10.1016/j.rcim.2007.06.002

[8] J. A. Sánchez *et al.*, "An Agent-Based Approach to the Construction of Floristic Digital Libraries", in *Proc. of the 3rd ACM Conf. on Digital Libraries*, New York, NY, USA, 1998. http://dx.doi.org/10.1145/276675.276699

[9] P. E. Dossou *et al.*, "The Use of Multi-Agent Systems for Improving a Logistic Platform in a GRAI Environment", *Highlights of Practical Applications of Agents, Multi-Agent Systems, and Sustainability-The PAAMS Collection*, Springer International Publishing, pp.126–135, 2015. http://dx.doi.org/10.1007/978-3-319-19033-4_11

[10] S. Gao and D. Xu, "Conceptual Modeling and Development of an Intelligent Agent-Assisted Decision Support System for Anti-Money Laundering", in *Expert Systems With Applications*, vol. 36, no.2, pp. 1493–1504, 2009. http://dx.doi.org/10.1016/j.eswa.2007.11.059

[11] Z. Ren and C. J. Anumba, "Multi-Agent Systems in Construction-State of the Art and Prospects", in *Automation in Construction*, vol. 13, no. 3, pp. 421–434, 2004. http://dx.doi.org/10.1016/j.autcon.2003.12.002

[12] R. Gowri *et al.*, "Development of Multi-Agent System for Fire Accident Detection Using Gaia Methodology", in *Int. Journal of Computer Science and Information Security*, vol. 8, no. 1, pp. 190–194, 2010.

[13] R. A. Gonzalez, "Developing a Multi-Agent System of a Crisis Response Organization", in *Business Process Management Journal*, vol. 16, no. 5, pp. 847–870, 2009. http://dx.doi.org/10.1108/14637151011076502

[14] L. Chee Wyai *et al.*, "Engineering Sustainable Software: A Case Study from Offline Computer Support Collaborative Annotation System", in *IEEE 9th Malaysian Software Engineering Conference (MySEC)*, 2015, pp. 272-277. http://dx.doi.org/10.1109/MySEC.2015.7475232

[15] C. Wai Shiang *et al.*, "Agent Oriented Requirement Engineering for Lake Mathematical Modelling: Preliminary Study", *Journal of Telecommunication, Electronic and Computer Engineering (JTEC),* vol 8, no 2, pp. 5–10, 2016.

[16] A. Oluyomi *et al.*, "Description Templates for Agent-Oriented Patterns", in *The Journal of Systems & Software*, vol. 81, no. 1, pp. 20–36, 2008. http://dx.doi.org/10.1016/j.jss.2007.06.020

[17] A. Sturm and O. Shehory, "Agent-Oriented Software Engineering: Revisiting the State of the Art", in *Agent-Oriented Software Engineering: Reflections on Architectures, Methodologies, Languages and Frameworks*, pp. 13–26, 2014. http://dx.doi.org/10.1007/978-3-642-54432-3_2

[18] Q. N. N. Tran and G. Low, "MOBMAS: A Methodology for Ontology-Based Multi-Agent Systems Development", in *Information and Software Technology*, vol. 50, no. 7–8, pp. 697–722, 2008. http://dx.doi.org/10.1016/j.infsof.2007.07.005

[19] P. Koutsabasis and J. Darzentas, "Methodologies for Agent Systems Development: Underlying Assumptions and Implications for Design", in *AI & Society*, vol. 23, no. 3, pp. 379–407, 2009. http://dx.doi.org/10.1007/s00146-007-0110-9

[20] F. Zambonelli *et al.*, "Organisational Rules as an Abstraction for the Analysis and Design of Multi-Agent Systems", in *Int. Journal of Software Engineering and Knowledge Engineering*, vol. 11, no. 3, pp. 303–328, 2001. http://dx.doi.org/10.1142/S0218194001000505

[21] K. H. Dam and M. Winikoff, "Comparing Agent-Oriented Methodologies", in P. Giorgini, B. Henderson-Sellers and M. Winikoff (Ed.), Agent-Oriented Information Systems, Springer LNAI 3030, pp. 78–93, 2004.

[22] L. Sterling and K. Taveter, "The Art of Agent Oriented Modelling", Cambridge, MA: MIT Press, 2009.

[23] J. M. Gascueña *et al.*, "Knowledge Modelling through Computational Agents: Application to Surveillance Systems", in *Expert Systems, The Journal of Knowledge Engineering*, vol. 28, no. 4, 2011. http://dx.doi.org/10.1111/j.1468-0394.2011.00609.x

[24] J. Garcia *et al.*, "Agent-Based Coordination of Cameras", in *Int. Journal of Computer Science and applications*, vol. 2, no. 1, pp. 33–37, 2005.

[25] D. Vallejo *et al.*, "A Multi-Agent Architecture for Supporting Distributed Normality-Based Intelligent Surveillance", in *Engineering Applications of Artificial Intelligence*, vol. 24, no. 2, pp. 325–340, 2011. http://dx.doi.org/10.1016/j.engappai.2010.11.005

[26] F. Castanedo *et al.*, "Data Fusion to Improve Trajectory Tracking in a Cooperative Surveillance Multi-Agent Architecture", in *Information Fusion*, vol. 11, pp. 243–255, 2010. http://dx.doi.org/10.1016/j.inffus.2009.09.002

[27] J. Pavón *et al.*, "Development of Intelligent Multisensor Surveillance Systems with Agents", in *Robotics and Autonomous Systems*, vol. 55, no. 12, pp. 892–903, 2007. http://dx.doi.org/10.1016/j.robot.2007.07.009

[28] J. M. Gascueña *et al.*, "Model-to-Model and Model-to-Text: Looking for the Automation of VigilAgent", in *Expert Systems: The Journal of Knowledge Engineering*, vol. 31, no. 3, pp. 199–212, 2014. http://dx.doi.org/10.1111/exsy.12023

[29] J. M. Gascueña and A. Fernández-Caballero, "On the Use of Agent Technology in Intelligent, Multisensory and Distributed Surveillance", in T*he Knowledge Engineering Review*, vol. 26, no. 2, pp. 191–208, 2007. http://dx.doi.org/10.1017/S0269888911000026

[30] T. Juan *et al.*, "ROADMAP: Extending the Gaia Methodology for Complex Open Systems", *Proc. of the 1st Int. Joint Conf. on Autonomous Agents and Multiagent Systems: part 1*, pp. 3–10, 2002. http://dx.doi.org/10.1145/544741.544744

[31] D. Wilmann and L. Sterling, "Guiding Agent-Oriented Requirements Elicitation: HOMER", *Proc. of the 5th Int. Conf. on Quality Software*, 419–424, 2005. http://dx.doi.org/10.1109/QSIC.2005.34

[32] C. Wai Shiang *et al.*, "Task Knowledge Patterns reuse in Multi-Agent System Development", *Proc. of the 13th Int. Conf. on Principles and Practice of Multi-Agent Systems*, Kolkata, India, 2010. http://dx.doi.org/10.1007/978-3-642-25920-3_33

[33] P. Bresciani *et al.*, "Tropos: An Agent-Oriented Software Development Methodology", in *Autonomous Agents and Multi-Agent Systems*, vol. 8, no. 3, pp. 203–236, 2004.

[34] J. M. Gascueña and A. Fernández-Caballero, "Agent-Oriented Modeling and Development of a Person-Following Mobile Robot", in *Expert Systems with Applications*, vol. 38, no. 4, pp. 4280–4290, 2011. http://dx.doi.org/10.1016/j.eswa.2010.09.096

[35] V. Marik and D. McFarlane, "Industrial Adoption of Agent-Based Technologies", in *IEEE Intelligent Systems*, pp. 27–35, 2005. http://dx.doi.org/10.1109/MIS.2005.11

[36] A. Morozov *et al.*, "Intelligent Visual Surveillance Logic Programming: Implementation Issues", in *Workshop on Implementation of Constraint and Logic Programming Systems and Logic-based Methods in Programming Environments*, pp. 31, 2014.

Received: December 2015
Revised: November 2016
Accepted: November 2016

Contact addresses:
Cheah Wai Shiang
Faculty of Computer Science & IT
Universiti Malaysia Sarawak
94300 Kota Samarahan
Sarawak, Malaysia
e-mail: c.waishiang@gmail.com

Bong Tien Onn
Faculty of Computer Science & IT
Universiti Malaysia Sarawak
94300 Kota Samarahan
Sarawak, Malaysia
e-mail: onnes2209@yahoo.com

Fu Swee Tee
Faculty of Engineering, Computing and Science
Swinburne University Sarawak
Room No:E609, Building E, FECS
Sarawak, Malaysia
e-mail: sfu@swinburne.edu.my

Muhammad Asyraf bin Khairuddin
Faculty of Computer Science & IT
Universiti Malaysia Sarawak
94300 Kota Samarahan
Sarawak, Malaysia
e-mail: kmasyraf@unimas.my

Msury Mahunnah
Department of Informatics
Faculty of Information Technology
Tallinn Technology University
Tallinn, Estonia
e-mail: msurym@gmail.com

Cheas Wai Shiang is a senior lecturer at the Faculty of Computer Science & IT, Universiti Malaysia Sarawak. His research interests include agent-oriented modelling and simulation. His current research focus is to investigate agent-oriented modelling for cognitive agent in virtual world and robotic.

Bong Tien Onn was a Master student by coursework at the Faculty of Computer Science & IT, Universiti Malaysia Sarawak. His research work is investigating agent-oriented methodology for intelligent video surveillance system.

Fu Swee Tee is a Master student by coursework at the Faculty of Engineering, Computing and Science. Her research work is investigating ontology patterns for multi agent system development. She is currently a lecturer at Swinburne University Sarawak, Malaysia.

Muhammad Asyraf bin Khairuddin is a lecturer at the Faculty of Computer Science & IT, Unimas. He is currently pursuing a PhD study in the area of requirement engineering.

Msury Mahunnah is a PhD student at the Faculty of Information Technology, Tallinn Technology University, Estonia