Cooperation between Human and rObotic teams in catastroPhic INcidents

Deliverable D2.2 / Milestone M3

v1.0

April 2014

http://chopin.isr.uc.pt

Document Identification:

Project Title:	CHOPIN - Cooperation between Human and rObotic teams in catastroPhic INcidents
Project Reference:	PTDC/EEA-CRO/119000/2010
Duration of Project:	01/04/2012 – 31/03/2015
Leader Organization:	Institute of Systems and Robotics – University of Coimbra
Principal Investigator:	Prof. Rui P. Rocha
Work Package:	Work Package: WP2. Collaborative context awareness and context sharing between team of humans and team of robots
Deliverable:	D2.2. Context-aware robotic devices for urban search and rescue
Due Date:	April 22, 2014
Version / Date:	v1.0 / April 2014
Authors:	Eduardo Martins, João Quintas, Jorge Lobo, Paulo Menezes

Document Change Record:

Version	**Date**	**Edited by**	**Brief Description / Changes Summary**
v0.1	7/04/14	Jorge Lobo	update from LaTeX of D2.1 to D2.2
v0.2	20/04/14	Paulo Menezes	Corrections to the LaTeX formatting
v0.3	05/05/14	Paulo Menezes	Reviewing of chapters 3 and 5
v0.4	24/05/14	Jorge Lobo	Editing and Reviewing of summary intro and chapter 2 and TOC numbering
v1.0	25/05/14	Rui Rocha	final check and release of v1.0

Contents

Summary **13**

1 Context-awareness in urban search and rescue robotics **15**
1.1 Context . . . 15
1.2 Context-awareness . . . 17
1.2.1 Application to Scene Understanding . . . 17
1.2.2 Application to Smart Spaces . . . 17
1.2.3 Application to Wearable Sensor Networks . . . 18
1.2.4 Application to Robotics . . . 18
1.3 Context representation . . . 18
1.3.1 Knowledge representation . . . 19
1.3.2 Knowledge representation in SAR systems . . . 20
1.4 Context-aware systems architectures . . . 21
1.4.1 Self-learning systems . . . 21
1.4.2 Middleware and Context Management Systems . . . 23
1.5 CHOPIN Context . . . 24
1.5.1 Context recognition methodology . . . 26

2 iBombeiro hand-held device for first responders **29**
2.1 Introduction . . . 29
2.1.1 Related Work on Action Recognition on Mobile Devices . . . 29
2.1.2 Motivation . . . 30
2.1.3 Our approach . . . 32
2.1.4 Chapter summary . . . 34
2.2 The problem of online classification for firefighter action . . . 35
2.2.1 State of the art . . . 35
2.2.2 Feature Pre-Processing . . . 39
2.2.3 Action model . . . 42
2.2.3.1 iBombeiro Action Recognition Model . . . 44
2.2.3.2 Extended iBombeiro Recognition Model . . . 49
Learning Personalised iBombeiro Action Data . . . 49
2.2.4 Experimental setup in MATLAB . . . 51
2.2.4.1 Data Visualization in MATLAB . . . 51
2.2.4.2 Classifier . . . 52
2.2.4.3 Results and discussions . . . 53
2.3 Iphone Implementation . . . 54
2.3.1 iPhone as a device for collecting Data . . . 55
2.3.1.1 Sensors . . . 55

2.3.1.2 Parse and app for collecting data 57
2.3.2 iBombeiro app 58
2.3.2.1 Classifier implementation in IOS 60
2.3.2.2 Comunication protocol 60
Sockets 62
Webservices 64
2.3.2.3 iPhone database: CoreData 66
2.3.2.4 Experiments with an iPhone integrated in a manet . . 68
2.4 Conclusions and future work for iBombeiro 69

3 Programming and integration in ROS of context-aware SaR mobile robots for urban firefighting 73
3.1 Introduction 74
3.1.1 Context and motivation 74
3.1.2 Objectives 75
3.2 Multi-Sensor Information Fusion 76
3.2.1 Sensors 76
3.2.2 Multi-Sensor Information Fusion 76
3.3 Classification Methods 81
3.3.1 Neural Network 81
3.3.1.1 Types of Artificial Neural Networks 81
3.3.1.2 Networks based on Feedback and Feedforward connections 81
3.3.1.3 Methodology: Training, Testing and Validation Datasets and Classification 82
3.3.2 Fuzzy 86
3.3.2.1 Fuzzy Logic 86
3.3.3 Support Vector Machines 89
3.3.3.1 SVM 89
3.3.3.2 SVM kernel 90
3.3.4 Bayesian 93
3.3.4.1 Bayesian Probability 93
3.3.4.2 Sensor Models and Multisensor Bayesian Inference . . 93
3.3.4.3 Sensor Models and Multisensor Bayesian Inference . . 94
3.3.4.4 Naive Bayes classifier 94
3.4 A Comparative Study of Classification Methods 96
3.4.1 Discussion 96
3.4.2 Conclusion 96

4 Requirements analysis and design of a command center for USAR robotics 99
4.1 Introduction 99

4.2 Requirements analysis . . . 101
4.2.1 Application Context . . . 101
4.2.2 Operation Mode of Portuguese Firefighters . . . 101
4.2.3 Decision making . . . 103
4.2.4 Situation Awareness . . . 104
4.2.5 Risk Assessment . . . 105
4.2.6 User Classes . . . 105
4.3 Interface requirements . . . 106
4.3.1 Hardware Interfaces . . . 106
4.3.2 User interface requirements . . . 107
4.3.3 Context-awareness requirements . . . 109
4.3.4 Human-Robot Interaction requirements . . . 110
4.3.5 Functional Requirements . . . 112
4.3.5.1 Search and Rescue requirements . . . 112
4.3.5.2 Command and Control requirements . . . 114
4.3.5.3 Situation Awareness requirements . . . 114
4.3.5.4 Self-Contained Breathing Apparatus requirements . . . 115
4.3.5.5 Resources Management requirements . . . 115
4.3.5.6 User Warnings requirements . . . 115
4.3.6 Non functional requirements . . . 116
4.3.6.1 End User Operation . . . 116
4.3.6.2 Information Display . . . 116
4.3.7 Taxonomy of information flows . . . 118
4.4 Design of the CHOPIN Command Center . . . 120
4.4.1 Design of the CC human-machine interface . . . 120
4.5 Interface with human and robotics first responders . . . 121
4.6 Database requirements analysis . . . 123
4.6.1 Conceptual cluster models . . . 123
4.6.2 Data modeling . . . 124
4.7 Future work . . . 125

5 Conclusion 127

A Use Cases and System Models 129
A.1 Use-case diagrams . . . 129
A.2 Class diagrams . . . 130

B Messages Definition 139
B.1 Command and Control Messages . . . 139
B.1.1 *start2Scan()* . . . 139
B.1.2 *area2Scan()* . . . 140

B.1.3 *task2Perform()* . . . 141
B.1.4 *tacticalMode()* . . . 143
B.1.5 *searchCompleted()* . . . 144
B.2 Location messages . . . 145
B.2.1 *robotPose()* . . . 145
B.2.2 *firefighterPose()* . . . 147
B.3 Status messages . . . 148
B.3.1 *batteryStatus()* . . . 148
B.3.2 *scbaStatus()* . . . 149
B.3.3 *teamComp()* . . . 150
B.3.4 *sharedLevel()* . . . 151
B.3.5 *netStats()* . . . 152
B.4 Conditions messages . . . 153
B.4.1 *securityCondition()* . . . 153
B.4.2 *altExit()* . . . 155
B.4.3 *siteView()* . . . 156
B.5 Mapping messages . . . 158
B.5.1 *compressedMap()* . . . 158
B.5.2 *tempMap()* . . . 159
B.5.3 *gasMap()* . . . 161
B.5.4 *victimFound()* . . . 162
B.5.5 *smokeMap()* . . . 164
B.5.6 *hazardMat()* . . . 165
B.5.7 *fireOutbreak()* . . . 167
B.6 Messages and Human-Robot Interaction (HRI) taxonomy . . . 169

C Entity-Relationship Diagram **171**
C.1 Conceptual models . . . 171
C.1.1 *Incident cluster* . . . 172
C.1.2 *Incident divisions cluster* . . . 173
C.1.3 *Incident occurrences cluster* . . . 174
C.1.4 *Incident resources cluster* . . . 175
C.1.5 *Incident personnel cluster* . . . 176
C.1.6 *Incident locations cluster* . . . 177

Bibliography **179**

List of Figures

0.1 Basement garage of a shopping center (left) and underground fire (right). . 14
0.2 Manufacturing floors with risk of toxic gases leakage. 14

1.1 Abstract layer architecture for Context-Awareness Systems 22
1.2 Relation between context classification and description levels in D4.1 . . . 24

2.1 Firefighter equipped to wildfires and urban fires [CA05] 31
2.2 Emergency movement, used in diving 33
2.3 Personal Security Alarm [CA05] 33
2.4 User Acceleration and FFT for specified actions 42
2.5 User Acceleration for (a) A graphical model illustrating condicional independence between variables X and Y given the variable Z, (b) Simple Bayesian Network 43
2.6 iBombeiro Bayesian Model. For each a[n] an independent FFT is computed 45
2.7 Bayesian Program: iBombeiro Action Recognition Model 46
2.8 Gaussian distribution for specified actions 48
2.9 Extended iBombeiro Bayesian Model. 49
2.10 Bayesian Program: Extended iBombeiro Action Recognition Model 50
2.11 Data Visualizer in @Matlab 51
2.12 The three axes on the iPhone. 56
2.13 iPhone app for collecting data 58
2.14 Parse cloud, where our files are stored 59
2.15 The Seven-Layer OSI Model 61
2.16 Communication structure of the different actors. 63
2.17 Webservice 65
2.18 IBombeiro entity inside ibombeiro database 67
2.19 Tableview 68
2.20 Interaction with the MANET - iPhone sending messages 69
2.21 Interaction with the MANET - iPhone receiving messages 69
2.22 A crowd of climbers slog up the Lhotse Face, heading toward Camp IV, last stop before the summit. Loose regulations and a boom in commercial guiding over the past two decades have made Everest far more accessible to experts and novices alike @Andy Bardon 72

3.1 Flow chart representation of the four steps 79
3.2 Feedback connection via sensor manager in a data fusion process 79
3.3 A team of mobile robots with multi-sensors cooperatively observing an area of fire in different points 80
3.4 A Two-layer, Feed-Forward Network with Four Inputs and Two Outputs[2] 84
3.5 A Recurrent Neural Network with Four Inputs and Two Outputs[2] 84

3.6 Fuzzy ARTMAP architecture [Carpenter et.al.,1992] 85
3.7 Fuzzy logic temperature 88
3.8 Mapping of an input space non-linearly separable for a feature space . . . 89
3.9 Linear SVM example 90
3.10 The model of multi-sensor information fusion based on SVM[ZL09] 92
3.11 Measure of eficiency of the three classifiers [AKP+11] 98
3.12 Avarage value of the AUC [LCP+] 98

4.1 Layout of the CC vehicle. 102
4.2 Tactical Situation ("SITuação TACtica") (SITAC) main board and forms to register incidents on the ground in the Operational Command Post ("Posto de Comando de Operações") (PCO). 103
4.3 The dynamic OODA loop (DOODA loop). functions are given in black, products in red, and inputs in green 104
4.4 System architecture 107
4.5 Taxonomy for multi-agent HRI. 111
4.6 The possible combinations of Command Center of Operations ("Centro de Comando de Operações") (CCO), firefighters and robots, acting as individuals or in teams. (adapted from [YD04]) 113
4.7 Communication structure of the different actors. 119
4.8 Tree structure of the messages between the CC, firefighters and robots (* indicates priority messages) 120
4.9 Human Machine Interface of the CC. 121
4.10 Human Robot Interaction of the CC 122
4.11 Database conceptual schema - *root entity cluster*. 123

A.1 Theatre of operations organization use-cases 129
A.2 Tactical Mode and Risk Assessment use-cases. 130
A.3 Personnel Management use-cases 131
A.4 Resources Management use-cases 131
A.5 Manage Water supply use-cases 132
A.6 Fire attack use-cases 133
A.7 Search and Rescue use-cases 134
A.8 Domain class Diagram 135
A.9 Fire Attack and Search and Rescue class diagram 136
A.10 Sequence Diagram of the Fire Attack and Search and Rescue operations . 137

B.1 Messages affecting the HRI taxonomy 169

C.1 Database conceptual schema - *root entity cluster*. 171
C.2 Cluster 1.1 - Incident conceptual model 172
C.3 Cluster 1.2 - Incident divisions conceptual model 173

C.4 Cluster 1.3 - Incident occurrences conceptual model 174
C.5 Cluster 1.4 - Incident resources conceptual model 175
C.6 Cluster 1.5 - Incident personnel conceptual model 176
C.7 Cluster 1.6 - Incident locations conceptual model 177

List of Tables

1.1 Information summary scenario 1 25
1.2 Information summary scenario 2 26

2.1 Matlab results 54
2.2 Matlab results for one more action 55

4.1 Search and Rescue Functional Requirements. 113
4.2 Command and Control Functional Requirements. 114
4.3 Situation Awareness Functional Requirements. 114
4.4 Self-Contained Breathing Apparatus Functional Requirements. 115
4.5 Resources Management Functional Requirements. 115
4.6 User Warnings Functional Requirements. 115
4.7 End User Operation non Functional Requirements. 116
4.8 Information Display non Functional Requirements. 117

Acronyms

BS Base Station . . . 106

CHOPIN Cooperation between Human and rObotic teams in catastroPhic Incidents 100

ROS Robot Operating System . . . 119

CCO Command Center of Operations ("Centro de Comando de Operações") . . 8

C2 Command and Control . . . 103

CSCW Computer-Supported Cooperative Work . . . 110

ERD Entity-Relationship Diagram . . . 124

USAR Urban Search and Rescue . . . 99

FPS First-Person Shooter . . . 122

HMI Human machine interface . . . 106

HRI Human-Robot Interaction . . . 6

CC Command Center . . . 99

CDOS District Command ("Comando Distrital de Operações de Socorro")

CODIS Comandante Distrital de Operações de Socorro . . . 100

SC Sector Commander . . . 106

LODD Line Of Duty Death . . . 99

MANET Mobile ad hoc Network . . . 100

ANPC National Authority for Civil Protection ("Autoridade Nacional de Protecção Civil") . . . 100

CEPLAN Planning Section ("CElula de PLANeamento") . . . 101

CELOG Logistics Section ("CElula de LOGística") . . . 101

CECOP Operations Section ("CElula de Combate/OPerações") . . . 101

PCO Operational Command Post ("Posto de Comando de Operações") . . . 8

COS Incident Commander ("Comandante das Operações de Socorro") . . . 101

SIRESP Portuguese Integrated Security and Emergency Network System ("Sistema Integrado das Redes de Emergência e Segurança de Portugal") . . . 102

TETRA European Terrestrial Trunked Radio . . . 102

TO Theatre of Operations . . . 99

SADO Operational Decision Support System ("Sistema de Apoio à Decisão Operacional")

SQL Structured Query Language 124
SITAC Tactical Situation ("SITuação TACtica") 8
JSON JavaScript Object Notation 118
GPS Global Positioning System 106
GIS Geographic Information System 106
SGO Operations Management System ("Sistema de Gestão de Operações ") . 101
SA Situation Awareness 104
SCBA Self-Contained Breathing Apparatus 105
SGML Standard Generalized Markup Language 118
XML eXtensible Markup Language 118
ENB Firefighters National School ("Escola Nacional de Bombeiros") 100

Summary

This deliverable aims at reporting the research work carried out in the #2 work package (WP2) titled "Collaborative context awareness and context sharing between team of humans and team of robots" in the period from February 2013 to April 2014. This is one of the deliverables that belongs to the #3 milestone of the CHOPIN project, which was planned for the end of April 2014. This deliverable fulfils and corresponds to an aggregation of the following deliverables identified in the original project proposal:

- D2.1.2. Scientific publication presenting experiments with context-aware mobile robots in search and rescue scenarios.
- D2.2.1. Scientific publication about interchangeability of context information between human and robotic teams.
- D2.3.1. First design prototype of a context-aware HMI for FRs in the scope of SAR operations.
- D5.3. Hand-held device prototype for first responders with a first version of context-aware human-machine interfaces.

This deliverable describes the work developed to attain the main goals of WP2 and WP5, which are:

- Implementation of context recognition methods in mobile robots based on Bayesian inference techniques.
- Developing techniques of collaborative context awareness between a team of human agents and a team of mobile robots;
- Developing techniques for context sharing between these teams.
- Designing context-aware human-machine interfaces.
- Programming of hand-held devices to be used in human teams;
- Programming of mobile robots for experiments with robotic teams

The research work herein reported was preceded by the release in February 2013 of the deliverable, D2.1, of which D2.2 is an update.

Previously the requirements were analysed and the following two test scenarios for the project were defined:

- Fire in a large basement garage (Fig. 0.1);
- Leakage of toxic gases and radioactive substances in a facility (Fig. 0.2).

Figure 0.1: Basement garage of a shopping center (left) and underground fire (right).

Figure 0.2: Manufacturing floors with risk of toxic gases leakage.

This technical report results from the contributions of several people involved on this project and is organised as follows: This Introduction and overall structure was prepared by Jorge Lobo and Paulo Menezes; chapter 1 presents the definition of context-awareness for this project, and was prepared by João Quintas and Paulo Menezes; Chapter 2 describes the use of a mobile phone for acquiring signals about the firemen activities, prepared by Andrea Pinto and Jorge Lobo; Chapter 3 describes the integration SaR mobile robots in ROS, prepared by Nuno Ferreira and Rui Rocha; Chapter 4 presents the requirements analysis and design of the command center for this project, prepared by Eduardo Martins, Paulo Menezes and Rui Rocha; the report is closed by the Conclusions chapter 5, prepared by Paulo Menezes.

1 Context-awareness in urban search and rescue robotics

The study of the cooperative interaction between teams of autonomous robots and teams of human agents is essentially a new scientific problem that imposes new challenges to be addressed.

One important benefit of the cooperation between human teams and robotic teams is better situation awareness (SA) by humans. Preliminary empiric results about this interaction in the scope of search and rescue (SAR) [MB05], [SVS+10] demonstrate the need for deeper studies on how humans should interact with teams of autonomous mobile robots and on how to establish a mutual beneficial interaction as regards situation awareness (SA) and consequently sharing reliable context information.

Context affects almost all aspects of behavior, and it does so for the most part automatically, without a conscious reasoning effort. Human beings think and act according individualized experiences occurring in environments often constrained to socio-cultural and spatio-temporal constrains. Thus we can say that humans and other animals behavior is context-dependent.

The open challenges regarding situation awareness and context recognition and sharing include [MB05]: transforming raw data into cognitive information by fusing data either distributed in time/space; more flexible dissemination of information within the decision-making hierarchy, instead of the classical one-way information flow (upwards in the hierarchy); and selectively propagating information to distributed users with diverse needs based on context sharing and information utility assessment.

The CHOPIN project aim to address some of these issues and will focus on devising new methods for collaborative context awareness and context sharing between teams of humans and teams of robots.

1.1 Context

Bill Shilit et.al., in [SAW94a] is recognized as the first to introduce the concept of context-aware systems in ubiquitous computing. The paper defines context-aware computing and describes four categories of context-aware applications. However, the concept of context is not given a conclusive definition in the paper. It is referred as the information about where you are, who you are with and what resources are nearby. Nevertheless, a clear definition is presented for a context-aware system. The context-aware system adapts according to the location of use, the collection of nearby people, hosts, and accessible devices, as well as to changes in such things over time. Context-aware applications are presented in four categories: *proximate selection*, *automatic contextual reconfiguration*, *contextual information and commands* and *context-triggered actions*. The context is not identified automatically, thus for the different applications

a set of informations is considered to be contextual-information. For proximate selection applications the context is the location and locate-objects that are nearby. The located-objects can be physical or non-physical. For automatic contextual reconfiguration applications the context is location, the task that is being performed and the people involved in the task. For contextual information and commands applications the context is location, the located-objects nearby, the tasks that are typically associated to the location and the located-objects (contextual information). For context-triggered actions applications the context is similar to that of the contextual information and commands applications. In terms of prototyping the applications the contextual information was used to control the user graphical interface.

Jason Pascoe in [Pas98a] addressed the deployment of context-awareness in wearable computers. Four generic contextual capabilities that can be utilized sensing, adaptation, resource discovery, and augmentation. A brief overview has been given of the work in designing and implementing a Contextual Information Service to remedy the complexity of obtaining and working with contextual data that impedes the utilization of context-awareness in many applications. The CIS encourages developers to add context-aware features to their software by providing them with an extensible shared model of context that is transparent of any underlying complexities. Pascoe definition for context extend that introduced by Schilit [SAW94a] and presented it as a "*...subjective concept that is defined by the entity that perceives it*".

Turner in [Tur98] , defined context and situation as: "*The term context means any identifiable configuration of environmental, mission-related and agent-related features that has predictive power for an agent's behaviour. The term situation is used to refer to the entire set of circumstances surrounding an agent, including the agent's own internal state. Context is thus the elements of the situation that should impact behavior*". A suitable knowledge representation strategy is required in such a way that can be easily created by a human and understandable by the robot. Turner proposed the representation of contextual information using contextual schemas (c-schema), where each schema is organized in a conceptual, content-addressable memory. In his approach a c-schema is an auto-contained description of a specific context, created according a specified format and stored in text file.

Anind Dey et.al. in [Dey01] is a reference in terms of defining Context and Context-Awareness Systems. They extended previous definitions given by Schilit [SAW94a] and Pascoe [Pas98a] to a more general and objective conceptualization. Dey et. al. aimed to define context beyond the regular use of synonyms and comparison expressions, which often result in subjective interpretations and difficult the identification of which features should be considered context. In the last decade Dey et. al. definition for context and context-awareness systems has been adopted for several researchers and can be considered in the scope of our work. According to Dey et. al., context is "any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and

an application, including the user and applications themselves.". This means that "... if a piece of information can be used to characterize the situation of a participant in an interaction, then that information is context...". The definition of a context-awareness system is given then as a system that "... uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task...".

1.2 Context-awareness

Humans perception is heavily influenced by top-down predictions, making it more difficult to detect out-of-context objects than familiar ones, and there are numerous studies showing the priming effect of one concept on another (see e.g. Glass and Holyoak, 1986 [GH86]). Pevtzow and Goldstone (1994) [PG94] suggest that the categories a person has learned affects what features of an object he or she perceives. Context also impacts decision-making and action. Preference measurements are also context-sensitive (Mellers and Cooke, 1996 [MC96]). Learning is affected by context, as studies as far back as those of Pavlov have demonstrated. Context has been studied extensively in language use, usually with "context" meaning the history of prior utterances (e.g. Ferstl, 1994 [Fer94]), but also including other kinds of context. Holtgraves (1994) [Hol94] has found that the status of the speaker relative to the hearer affects whether the literal meaning of an indirect request is activated.

These and other examples have been motivating different scientists to design and implement context-aware systems in a variety of fields of application.

1.2.1 Application to Scene Understanding

In the literature we can find recent works that used contextual information models to understand contents in a scene. The authors of [GZS+09] [PBHD06] and [DYL09] used the context information in order to accurately limit the region of interest in images, increasing the performance of detection algorithms, since their approaches helped limiting the areas where the algorithms should look for objects in the scene by avoiding their programs to "waste" time processing image areas with no-meaning in the scene.

1.2.2 Application to Smart Spaces

The application of contextual information knowledge to Smart Spaces systems helped the authors from [vdVNHF] and [Ph.10] to design better interaction mechanisms between users and their environment. These advances envisage that in a short to mid-term time basis we will assist to a boom of pervasive and ubiquitous computing, with a degree on intelligence never seen before. Thus, the application of similar systems will be more

and more near the consumer market and will have a numerous number of applications from home health care, automatic surveillance domains, etc.

1.2.3 Application to Wearable Sensor Networks

In [PEK+06] and [CFA+09] the authors present their approach to integrate contextual models into wearable sensor networks systems. They used this type of sensor networks to monitor the user's biomechanical and physiological behaviours, envisaging future applications in the health care domain. The introduction of context models into their algorithms raised their applications performance in some cases from 22 to 96%, as proposed in [PEK+06].

1.2.4 Application to Robotics

In [CMS+09] the authors presented a roadmap for developmental robotics focusing on the integration of action and language knowledge in robotic systems. The work preconizes the still open challenges in Robotics. The relation between knowledge discovery and representation and action can be traduced into semantic concepts modeled as context. Thus, the field of Robotics will also have an open space for the application of context modeling and context-awareness applications. These thoughts are strengthened in the case of social robotics, where the robotic systems must deliver intelligible responses in the most cognitive-like way.

1.3 Context representation

Artificial intelligent agents must be context-sensitive and adapt its own behavior to changes from the environment and in their internal states. The context-mediated behavior (CMB), presented in [Tur98] presented an approach where an agent should have explicit knowledge about contexts in which it may find itself, then use that knowledge when in those contexts. The group of explicit context representation represented the knowledge base for the agent. In this knowledge base the contexts were represented by a set of sentences expressing the involved actors, the context features to be identified and the corresponding behaviour.

Complex domains of knowledge require more general and flexible representation besides first-order logic. For example abstract and general concepts – such as Event, Time, Physical Objects, and Beliefs – that occur in many different domains are not well suited for first-order logic representation. The process of representing such concepts is sometimes called ontological engineering [NR03].

The organization of objects into categories is a vital part of knowledge representation. Although interaction with the world takes place at the level of the individual objects, much of the reasoning process takes place at the level of categories [NR03].

1.3.1 Knowledge representation

In reasoning systems, categories play the important role of being the building blocks of large-scale knowledge representation schemes. In these systems knowledge is often represented by means of ontologies.

Categories help to organize the knowledge base through inheritance. This inheritance allows making predictions about the objects once they are categorized. Additionally, inheritance makes possible the reasoning process of inferring the presence of certain objects from perceptual input, inferring category membership from the perceived properties of the objects, and then uses category information to make predictions about the objects.

Subclass relations consolidate categories into a taxonomy or a taxonomic hierarchy. Taxonomies have been used for centuries in technical fields; therefore they represent an important aspect of general common-sense knowledge.

In reasoning systems, categories play the important role of being the building blocks of large-scale knowledge representation schemes.

Knowledge representation can be described using different format types. Popular formats comprise Resource Description Framework (RDF), the Web Ontology Language (OWL) and Semantic Web Rule Language (SWRL).

The Resource Description Framework (RDF) [w3cb] [wikb] is a framework for representing information in the Web. RDF is a family of World Wide Web Consortium (W3C) specifications originally designed as a metadata data model. It has come to be used as a general method for conceptual description or modeling of information that is implemented in web resources, using a variety of syntax formats.

The OWL Web Ontology Language is designed for use by applications that need to process the content of information instead of just presenting information to humans. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics. OWL has three increasingly-expressive sublanguages: OWL Lite, OWL DL, and OWL Full [w3ca] [wika].

The Semantic Web Rule Language (SWRL) is a proposal for a Semantic Web rules-language, combining sublanguages of the OWL Web ontology Language (OWL DL and Lite) with those of the Rule Markup Language (Unary/Binary Datalog) [w3cc] [wikc].

The representation of knowledge using these description languages and associated frameworks and tools can be integrated into a tool chain methodology that ease the development of knowledge based systems.

Different authors have studied the application and integration of this type of representation in different domains of application. Fernández-Breis and Martínez-Béjar [FBMB02] described a cooperative framework for integrating ontologies. In particular, they presented a system that could serve as a framework for cooperatively built, integration-derived (i.e., global) ontologies. Their system aimed towards ontology integration and was intended for use by normal and expert users. Maedche and Staab [MS00] created MAFRA. MAFRA is part of a multi-ontology system, and aimed to automatically detect similarities of entities contained in two different department ontologies. An interesting contribution of the MAFRA framework is the definition of a semantic bridge. This is a module that establishes correspondences between entities from the source and target ontology based on similarities found between them. Calvanese et al. [CDGL01] proposed a formal framework for Ontology Integration Systems – OISs. The framework provided the basis for ontology integration, which was the main focus of their work. Madhavan et al. [MBDH02] developed a framework and proposed a language for ontology mapping. Their framework enabled mapping between models in different representation languages without first translating the models into a common language, the authors claim. Kiryakov et al. [KSD01] developed a framework, OntoMapO, for accessing and integrating upper-level ontologies. They provided a service that allowed a user to import linguistic ontologies onto a Web server, which could then be mapped onto other ontologies. Kent (2000) proposed a framework for ontological structures to support ontology sharing. It was based on the Barwise-Seligman theory of information flow. Kent argued that IFF represents the dynamism and stability of knowledge. Moritz Tenorth and Michael Beetz in [Ten11] and [TKJ+10] presented KNOWROB. According to [Ten11], KNOWROB combines static encyclopedic knowledge, common-sense knowledge, task descriptions, environment models, object information and information about observed actions that has been acquired from various sources (manually axiomatized, derived from observations, or imported from the web). It supports different deterministic and probabilistic reasoning mechanisms, clustering, classification and segmentation methods, and includes query interfaces as well as visualization tools.

1.3.2 Knowledge representation in SAR systems

In the particular case of the SAR systems some authors also presented some works dedicated to represented relevant information models using ontologies.

In [SM05] the authors proposed a neutral knowledge representation for capturing relevant information about robots and their capabilities within search and rescue robot systems. They chose an ontological approach for its flexibility of adaptation as the robot requirements evolve. The Robot Ontology was presented containing 230 classes, 245 attributes (properties), and 180 instances. The authors identified as future work: further specialization of the ontology to provide greater level of detail in the areas that have already been addressed; explore other standards efforts and existing ontologies

that can be leveraged, such as ontologies for Sensors, Power Source, Materials, Environment; continue to incorporate the requirements from the requirements workshops into the robot ontology structure; Explore the use of reasoning engines to suggest robots as well as configurations (e.g., sensors to be mounted) for different situations.

In [Wei09] applied an ontological approach into solving the problems of semantic heterogeneity of information in decision-making systems for maritime search and rescue applications. In this paper, maritime search and rescue decision-making ontology reference model was proposed. They designed an ontology for classification on maritime perils as an example to illustrate the process of ontology description based on Protégé. In contrast to traditional knowledge-based approaches, e.g. formal specification languages, ontology seems to be well suited for an evolutionary approach to the specification of requirements and domain knowledge.

In [KK13] they demonstrated a method for the interaction of a robot with 3D landmarks in a search and rescue environment, based upon ontological knowledge, both pre-existing and additionally computed, as an aid to collaborative efforts by human-robot rescue teams. They performed experiments on some car models and robot configurations and found that poses thus generated by the functional mapping workflow perform far better than those by an algorithm naive of the ontological knowledge. In the future, they plan to perform experiments with a navigating robot, with a camera on a movable arm and plan trajectories around several crashed cars that optimize the amount of visualization inside these cars. Further, they plan to extend the notion of openings and containers to other use cases e.g., entering a hole into a room of known dimensions, climbing a known stairway and so on.

1.4 Context-aware systems architectures

An abstract architecture for context-aware systems can be drawn based on the literature that explores context-aware applications. The abstract architecture is consisted of four layers: (1) network layer, which involves a network supporting context-aware systems and sensor collecting low-level of context information; (2) middleware layer that manages processes and stores context information; (3) application layer that provides users with appropriate service; and (4) user infrastructure layer to offer suitable interface to users.

1.4.1 Self-learning systems

According to Oliver Brdiczka et. al in "... Computerized spaces and their devices require situational information, to respond correctly to human activity. In order to become context aware, computer systems must thus maintain a model describing the environment, its occupants, and their activities. ..." [BCR09] [BLMC09]. The typical

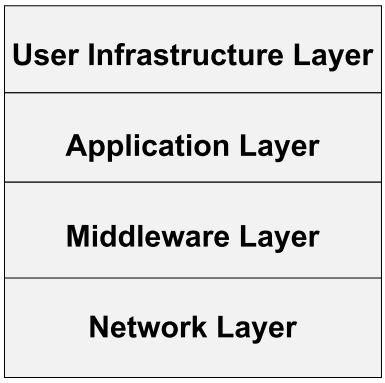


Figure 1.1: Abstract layer architecture for Context-Awareness Systems

approach to provide contextual information to the application is by manually defining the context models according to their end goal and taking into account particular user needs. This approach doesn't traduce the "real world" in the sense that it fails if the user needs evolution over time is taken into account. Brdiczka considered that "... New activities and scenarios emerge in a smart environment, and others disappear. New services must be integrated into the environment, whereas obsolete services should be deleted. Thus, a fixed context model is not sufficient. ..." [BCR09] [BLMC09]. Thus, the research for more intelligent, self-learning and self-adaptable systems is justified facing the inefficiency of the common approaches terms of long-term maintenance and economical prospect, i.e. having an expert periodically adjusting the system according user needs would be expensive. In [BCR09] and [BLMC09] they proposed a framework for learning situation models, with the objective of building an evolutionary context model to provide context-aware services in a smart environment. The proposed framework combined different machine learning methods that acquire and adapt the situation model. In their approach the roles are first learned and detected using support vector machines (SVMs) based data prepared by an expert. Situations are then extracted in an unsupervised manner from observation data using the Jeffrey divergence between sliding histograms. The resulting situation segments can then be used to learn situation labels given by the user, allowing the situation model to evolve according user feedback. For Situation Learning and Recognition they used Hidden Markov Models (HMMs) to modelling the different situations. Relation between Context-dependent systems and Knowledge Structure Context modeling can be seen as a higher level information fusion process, where the models take into account not only the discrete entities characteristics (e.g. color, shape, etc.) but also the relationships between them and their surroundings or other nearby entities. An important component of higher level fusion is knowledge discovery. A recent publication from Bradley J. Rhodes [Rho07] addressed an automated process for discover ontologies for representing knowledge, such

as taxonomies/thesaurus. In his approach he used a neural network with associative, incremental learning (NAIL) algorithm to process the co-occurrences in imagery-based data to find the relationships between elements of the lexicon, thus "... uncovering the knowledge structure 'hidden' in the dataset...". He illustrated the usefulness of knowledge structure discovered using a Bayesian network, which produced an inference engine capable of exploiting the learned knowledge model. According Bradley in [Rho07] "...Effective automation of knowledge discovery in an information fusion context has considerable potential for aiding the development of machine-based situation awareness capabilities...". The NAIL algorithm was used in recent works like [ZRB08] and [GZS+09]. In [ZRB08] they combined the NAIL algorithm with the Object Probabilistic Associative Learning (OPAL) and the Hyper-Elliptical Learning and Matching (HELM) algorithms, in order to propose the Complex Object Associative Learning Enables Scene Exploitation neural network (COALESCE) algorithm. In [GZS+09] a hybrid neuro-Bayesian spatial contextual reasoning algorithm was proposed for scene content understanding, based on the OPAL and a modified version of NAIL designated by Probabilistic Neuro Associative Incremental Learning (pNAIL) algorithm. In both works the authors proposed biological-inspired approaches, which emphasize the reasoning without a priori knowledge. The proposed algorithms learned the association between scene and complex objects and their primitive components with and/or without a priori knowledge. Additionally, the spatial relationships between the simple constituents and their probabilities were learned incrementally.

1.4.2 Middleware and Context Management Systems

In [MLKK06] and [dREdS08] middlewares are presented for context-aware systems. In [MLKK06] they presented the system architecture of Context-aware Middleware for URC Systems (CAMUS). CAMUS is to provide a framework for the development and execution of context-aware applications for Ubiquitous Robotic Companion Systems (URC) such as network-based robots and software robots. CAMUS gathers context information from different sensors and delivers filtered context information to different applications. The authors of [dREdS08] discussed the challenges and trade-offs of the implementation of middlewares to support ubiquitous context-awareness, i.e., where context-aware applications may move through different contexts without suffer disruption in their context-based interactions, ensuring the provision of applications services. In [PBL+09] and [GRB09] the authors propose context management systems that were designed to manage context information sharing between different context-aware devices and these devices have their context changed. These works address the problem of context information exchange, which need to be tackled when using portable devices that move or are carried between different places. In [PBL+09], a bridging approach is proposed that allowed different Context Management Systems (CMSs) exchange information trough the mobile device. They focused on a bridge named Context

Discovery Adapter (CDA) which bridged two CMSs with greatly different properties, and allowed a mobile device to be the exchanging "channel" between the two CMSs. In [GRB09] they presented their studies to identify the most significant CMS features, to be taken into account in a mobility context and made a comparison between the available CMSs at the time.

1.5 CHOPIN Context

Context will be defined as the set of information which influence the performance of an agent while attempting to execute a desired behaviour. This means that any set of information will be only considered to be context if it anticipate how the agent should behave when that information is present.

According to the literature they can be identified different types of context, varying according the information to be represented. The sets of information identified to be context can represent either external (physical) elements that can be measured by sensors (e.g. location, light, sound, movement, temperature, etc.), or internal elements related with the entity's goals, tasks, beliefs, etc.

Examples of different contexts in search and rescue (SAR) scenarios are "robot progressing in an unknown environment", "robot searching for victims", "robot escaping from danger", "robot near a victim", etc.

Three main classes of context were identified, which are related with environment, mission and self-characteristics of the agent. These classes group information at distinct levels of abstraction. This means that we can relate each of this classes to a different description level on the previous work done in the report D4.1.

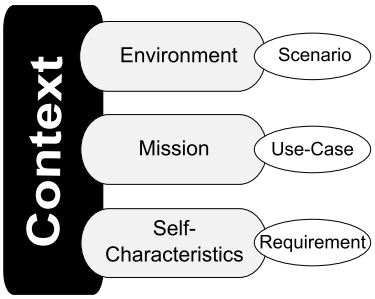


Figure 1.2: Relation between context classification and description levels in D4.1

In figure 1.2 we illustrate the relation between each class of contextual information and the levels described as scenario, use-case and requirement in the requirement analysis done in the report D4.1.

Each class can group different sets of attributes or properties configurations to represent different contexts of that class. The attributes or properties will be designated as *features of context*.

Taking into account these considerations, we must explore some possibilities for the expected contexts and behaviours to be considered by the human and robotic agents. In this report, this analysis will be presented from an individual perspective skipping team approach to the next deliverable.

Deliverable D4.1 was analysed in order to identify major classes of information that could be used as a taxonomy for the knowledge explicitly described in scenarios 1 and 2 including corresponding use-cases and requirements. This information was summarized in tables 1.1 and 1.2.

Table 1.1: Information summary scenario 1

Scenario	Use Case	Task	Requirement	Equipment
sc1 Fire in large basement garage	uc1.1 Co-operative mapping and exploration through smoke	t1.1.1 deploing a MANET	range sensors not disturbed by smoke peer-to-peer and multicast ad hoc wireless communication through MANET cooperative SLAM, navigation and exploration algorithms cooperative mapping of relevant environmental variables (sensors for gases concentration, temperature) detection of human bodies based on thermal and vocal information audio and light beacons to attract attention input/output audio devices and light for telepresence near the victim pull bar where victims can grab while being guided in evacuation procedure	ground mobile robots (differential robot, car-like robot) range sensing through smoke (sonars, ultrawide band radar wireless communication (wifi, zigbee, bluetooth) SLAM, navigation, exploration algorithms (EKF, particle filter, visual SLAM) measurement of dangerous gases concentration (miscellaneous transducers) detection of human bodies (infrared camera, thermopile) detection of sound direction (microphone array) signalizing the presence of the robot (yellow light, speaker, beeper) remote image of poor illuminated sites (cameras, light) audio communication (microphone, speaker) guiding victims in evacuation procedure (pull bar)
		t1.1.2 exploring the environment		
		t1.1.3 building a map for navigation		
		t1.1.4 detecting and localizing fire outbreaks		
		t1.1.5 detecting and localizing victims		
	uc1.2 Assessment of victim health condition and telepresence	t1.2.1 signalizing location		
		t1.2.2 monitor fire evolution in the surroundings and transmit information to the command center		

Table 1.2: Information summary scenario 2

Scenario	Use Case	Task	Requirement	Equipment
sc2 Leakage of toxic gases and radioactive substances in a facility	uc2.1 Definition of security perimeter around incident zone	t2.1.1 sweeping periodically the incident zone contour	common range sensors for robot's map building and navigation multi-hop peer-to-peer and multicast ad hoc wireless communication through MANET cooperative SLAM, navigation and exploration algorithms cooperative mapping of relevant environmental variables (sensors for gases concentration, temperature) detection of human bodies based on computer vision, thermal and vocal information video stream of the robot's operations (e.g. teleoperation) audio and light beacons to attract attention input/output audio devices and light for telepresence near the victim pull bar where victims can grab while being guided in evacuation procedure	ground mobile robots (differential robot, car-like robot) small unmanned aerial vehicles (helicopter, quadrotor drone) optical range sensors (LRF, kinect) wireless communication (wifi, zigbee, bluetooth) SLAM, navigation, exploration algorithms (EKF, particle filter, visual SLAM) measurement of dangerous gases concentration (miscellaneous transducers) detection of human bodies (computer vision) detection of sound direction (microphone array) signalizing the presence of the robot (yellow light, speaker, beeper) remote image of poor illuminated sites (cameras, light) audio communication (microphone, speaker) video interaction (digital display, camera) guiding victims in evacuation procedure (pull bar)
		t2.1.2 cooperative assessment of dangerous substances		
		t2.1.3 cooperative mapping		
	uc2.2 Cooperative reconnaissance by mobile robots inside incident zone	t2.2.1 cooperative exploration of the environment and building a map for both robot and human navigation		
		t2.2.2 cooperative detection and precise localization of leakage source		
		t2.2.3 cooperative assessment of dangerous substances and temperature within the facility		
		t2.2.4 cooperative detection and localization of victims		
	uc2.3 Assessment of victim health condition and telepresence	t2.3.1 signalizing location		
		t2.3.2 monitor concentration of dangerous substances and temperature in the neighbourhood of the victim		
		t2.3.3 establish a contact bridge between the victim and remote human rescuers		

1.5.1 Context recognition methodology

The sharing of context between teams of humans and teams of robots requires that robots are endowed with context recognition capabilities. In this project, Bayesian inference methods for context recognition will be implemented on mobile robots, based on the extensive know-how of the research team in Bayesian techniques [LFD09], [FLD10], [MLD11]. Despite contributing to SA, the development of context awareness in the scope of the interaction between human and robotic teams can be used to share in-

formation efficiently and selectively among agents, either human or robotic, based on measures of information utility assessment. For instance, in a search and rescue (SAR) mission, when the robotic team finds a new victim, it may use its awareness about the context of human agents to share this information only with those agents that are also searching for victims and are available to go to the newly found victim's place and rescue him/her.

Before studying context sharing between teams of humans and team of mobile robots, which is the main goal of task T2.2, this task aims at endowing mobile robots with artificial context recognition capabilities, thus mimicking context recognition abilities usually exhibited by humans. This means programming context-aware mobile robots, i.e. robots with the ability to characterize the elements of the environment and its task [BDR07].

These methods involve the real-time processing of sensing data (raw-data) from different sensors on-board the robot, to obtain higher-level semantic information that can be used by the robot to perceive its current context and reason about it. They will be implemented in mobile robots which are equipped with different subsets of the following set of sensors: vision (either monocular or stereo), laser range finder, sonars, microphone, accelerometers and gyroscope, GPS, and sensors for various environmental quantities (temperature, humidity, concentration of toxic gases, etc.).

Bayesian probabilistic techniques have been used to represent and deal with uncertain, imprecise and incomplete information in a variety of different problems of robotics, namely in artificial perception.

Lobo et al. employed a Bayesian approach to propose a visual-inertial image stabilization method [LFD09] which mimics the human vestibular system in robotic systems, through sensor fusion of artificial vision and inertial sensors. In the sequel, a real-time implementation of a Bayesian framework for robotic multisensory perception on a graphics processing unit [FLD10] was presented. This framework provides multimodal perception based on stereo vision, stereo auditory and inertial sensors and takes inspiration from biological evidence of these different sensing modalities and their high interdependence. In [MLD11], Menezes et al. used a particle filter to perform human motion capture. The technique was demonstrated in an assistant mobile robot which was able to track the 3D motion of the upper human body parts from sequences of monocular images. Similar Bayesian approaches are particularly well suited to recognize a given context from a set of different contexts that a robot should be able to distinguish. Examples of Bayesian probabilistic models that will be considered in this task include dynamic Bayesian networks, hidden Markov models (HMMs), Kalman filters, particle filters, mixture models and maximum entropy models.

2 iBombeiro hand-held device for first responders

2.1 Introduction

Mobile devices are becoming increasingly sophisticated and the latest generation of smart cell phones now incorporates many diverse and powerful sensors. These sensors include acceleration sensors (accelerometers), GPS sensors, vision sensors (cameras), audio sensors (microphones), light sensors, temperature sensors, direction sensors (magnetic compasses).

The objective of this work is to have a system capable of automatically classifying the physical actions performed by a human subject. By the term physical actions, we mean either static postures, such as standing, or dynamic motions, such as walking, running, stair climbing, descend stairs.

2.1.1 Related Work on Action Recognition on Mobile Devices

Real-world processes generally produce observable outputs which can be characterised as signals. The signals can be discrete or continuous in nature. The original source can be stationary or non-stationary (i.e, the signal properties vary over time). The signals can be pure, or can be corrupted from other signal sources (e.g., noise) or by transmission distortions, reverberation, etc.
A problem of fundamental interest is characterising such real world signals in terms of signal models. [Rab89]
Action recognition refers to the process of understanding and classifying meaningful movements, by studying those signals. There are many diferent works in the field of action recognition.
Researchers have already prototyped wearable computer systems that use acceleration, audio, video, and other sensors to recognize user activity.
Research has motivated activity recognition in numerous domains:

- Elderly Care. Monitoring the Activities of Daily Living estimate quality of self-care and enable assisted living. [HM11]
- Proactive Healthcare. Fighting obesity, coaching toward more active life. [SZL+10, LRT+, KWM10]
- Psychiatry. Correlation of activities with moods, mood swings, manic depressions.
- Security / Workflow Monitoring. Tracking maintenance staff, crowds, managing accountability, emergency/rescue operations, security concerning questions.

- Groupware. Sharing activity information in groups / over social networks.
- Memory Support. Managing diaries, auto-filling journals for later accounting.

Besides the areas cited above, where the usage of the smartphones is increasing, especially in the elderly care area, there's also a growing number of applications for smartphones that are already using GPS positioning to track user's activity. The user can choose an activity from a predefined list, and with the positioning and the predefined values for each activity, concerning calories, distances, altitudes, accelerations (calculated in the distance/time basis) the smartphone can track and measure all the variables for that user/activity.

We can cite as example Sports Tracker (`http://www.sports-tracker.com/`), an old product from Nokia that now as been developed to all major plataforms. The free app is available for Android, iPhone, Windows Phone, Nokia N9 and Nokia Symbian devices. Sports Tracker started in 2004, while his developers worked at Nokia. The application was the first of its kind for mobile phones, although at that time there were no phones with GPS, smartphones or Google Maps. With Sports Tracker it is possible to track and analyze performances, monitor progress, store all training data in a personal workout diary and sharing the workouts and photos with friends and other similar-minded people via the Sports Tracker community, Facebook and Twitter.

Another great and free app is RunKeeper (`http://runkeeper.com/`). RunKeeper is available for Android and iOS and is targeted as a "personal trainer in the pocket". RunKeeper track the user's activities as running, walking, cycling, hiking, biking and more, using the GPS in phone. Deal with stats, history, progress and nofifications. Mesure heart rate with many available sensors, among a huge list of services.

Other GPS based tracking apps can be mencioned like Edumondo (`http://www.endomondo.com/`), or My Tracks, that belongs to Google Inc. (`https://code.google.com/p/mytracks/`,

A great advantage of the smartphone's globalization is the integration with other services such as social networks and cloud sharing. For example, there exists a synchonization app for RunKeeper to Evernote (`https://evernote.com/`).
Today, everything can be easily shared (despite the security issues concerning the "online status").

A major improvement in this kind of online applications would be the integration of the accelerometer data in the system in order to process activity recognition. The users were able to perform diferent activities without redefine the activity to be monitored.

2.1.2 Motivation

With the increase and constant evolution of the new technologies there are a huge number of sectors that are highly benefited. One of this sectores is clearly the civil

protection (Figure 2.1).
The civil protection aims to prevent collective risks and the occurrence of serious accidents or resulting disasters; to attenuate collective risks and to limit its effect; to rescue and to assist people and other living beings in danger; to protect cultural and environmental assets and other assets of high public interest and to support the reestablishment of normality in the life of people living in the areas affected by serious accidents or disasters.

Figure 2.1: Firefighter equipped to wildfires and urban fires [CA05]

The study of the cooperative interaction between teams of autonomous robots and teams of human agents is essentially a new scientific problem that imposes new challenges to be addressed.

In the context of first responder teams, the firefigter is a key "link", highly exposed to risks when in the field. Having mixed teams with robots can have operational advantages, but the firefighter need to be instrumented so that his state can be monitored.

It's necessary then to provide the firefighter the technology, integrated in human and robotic teams, in order to take full advantage of this human-machine paradigm, whether for its own survival, whether for the success of the team, by developing a

system that can provide data about his context awareness and about explicit signs, previously defined.

2.1.3 Our approach

We will use an iPhone to perform action recognition, a task which involves identifying the physical action a user is performing. To implement this system we collected labeled accelerometer data from users as they performed the activities and gestures previous defined. Six activities were considered: walking, standing, ascending stairs, descending stairs, running, and an explicit movement, to exemplify emergency.
With the collected data we provide a training dataset. We then used that resulting training dataset to induce a predictive model for action recognition. This predicting model was finally implemented in the iPhone, in order to perform the classification online, and in real time, sending the firefighter status to the Command Center Operations (CCO).

This work, iBombeiro, addresses the following problems:

- Having an iPhone, how can we use the acceleration data from iPhone sensors in order to have a powerful and lightweight system capable of action recognition?
- After the identification of the action recognition, how can we do the integration of the device as a node in the MANET, created in the purpose of the CHOPIN Project, in order to send messages with the firefighter status to the CCO?

This develoved device prototype for first responders will be named as ***iBombeiro***.

In the scope of the CHOPIN project, and in a meeting with the *Bombeiros Sapadores de Coimbra*, in order to define some procedures, has been observed that there is no specific movements or gestures to signal different situations. The possibility of using one or two simple gestures used in diving for emergencies was covered, so the movement to express emergency will be the one shown at Figure 2.2.

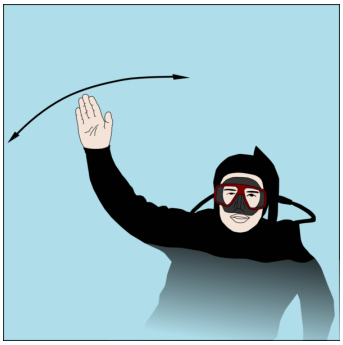


Figure 2.2: Emergency movement, used in diving[1]

The firefighters also use an emergency device, the Personal Alert Safety System - PASS (Figure 2.3).
PASS is a small device, heat and water projection resistant that the firefighter holds in the harness or in the Self-contained breathing apparatus (SCBA) and serves to alert others that he is in trouble. The device has a small beacon that indicates when connected, its proper functioning.
The emergency audible alarm can be triggered in two ways:

- Automatic - If the firefighter stay immobilized for more than 30 seconds;
- Manual - If the firefighter needs to launch an emergency alert.

Figure 2.3: Personal Security Alarm [CA05]

[1] http://en.wikipedia.org/wiki/Diver_communications

In general, this equipment is suitable for work in potentially hazardous environments, especially when firefighters have to work alone.
In this work, we simulate the PASS behavior. In the automatic mode, when a firefighter is standing more than 30 seconds, a flag is activated, and the status of the firefighter changes to "danger", and a sound is also triggered. The manual mode is simulated with the emergency movement. This work does not pretend to be a substitute of this kind of apparatus.

2.1.4 Chapter summary

This chapter is organized as follows.

In section 2.1 we present an overview of the iBombeiro work, the motivation and the context where this work is fitted. We present also related work in action recognition, for different areas.

Section 2.2 presents the relevant background material, for the problem of online classification. After a brief introduction to the state of the art of classifiers, and after an introduction to the theory of the Dynamic Bayesian Networks, we present the features that we used in this work. Then we propose a Bayesian Model for iBombeiro, followed by the respective Bayesian Programming.
In this section we also present the experimental setup that was developed in MATLAB, by explaining the algorithm used in the classifier. We finish the section by presenting the results obtained with the developed classifier, and we discuss those results.

In section 2.3 we present the iPhone as a device for collecting data. We present the implementation of the classifier formulated in section 2, in the iPhone (iBombeiro app). We also deal with the problem of the network communication, by presenting three diferent ways of showing the status of the firefighter: sending a message to the CCO via MANET, sending a message via webservice, saving the message in the iPhone database, for posterior analysis.

Section 2.4 presents a new Bayesian Model introduced in order to deal with more variables to represent a personalized implementation of the iBombeiro. Due to time constrains we present only the mathematical validation.

Finally, in section 2.5 we have some concluding remarks on the iBombeiro device to be used in CHOPIN, and provide directions for future work.

2.2 The problem of online classification for firefighter action

What is classification? Classification is the process of using a model to predict unknown values (output variables), using a number of known values (input variables).
In order to perform classification, first it's necessary to model the relationship between the input variables and the output variables we are predicting. This process involves learning a model using data in which both the input variables and the output variables are present. This model can subsequently be used on unseen data in which only the input data is present, in order to predict the output variables.

2.2.1 State of the art

There exists a wide range of features and algorithms for classification of action/gesture recognition with accelerometer derived features. Commonly used methods in the context of activity recognition include Naive Bayes classifiers, decision trees, nearest neighbor methods, support vector machines (SVM), Neural Networks (NN) and Hidden Markov Models (HMM).

One of the first dynamic gesture recognition systems was developed by [HHH97]. They present a method for the recognition of dynamic gestures using Hidden Markov Models (HMM). It addresses the problems of segmentation (the extraction of the beginning and ending of a gesture) and recognition of the preprocessed and segmented gesture data.The recognition took hours to complete.

In [PEa+06], methods used for classification of everyday activities like walking, running, and cycling are described. The aim of the study was to find out how to recognize activities, which sensors are useful and what kind of signal processing and classification is required. A large and realistic data library of sensor data was collected. Sixteen test persons took part in the data collection, resulting in approximately 31 hours of annotated, 35-channel data recorded in an everyday environment. The test persons carried a set of wearable sensors while performing several activities during the 2 hours measurement session. Classification results of three classifiers are shown: custom decision tree, automatically generated decision tree, and artificial neural network. The classification accuracies using leaveone-subject-out cross validation range from 58 to 97% for custom decision tree classifier, from 56 to 97% for automatically generated decision tree, and from 22 to 96% for artificial neural network. Total classification accuracy is 82% for custom decision tree classifier, 86% for automatically generated decision tree, and 82% for artificial neural network.

[SLC11] presents the designing of a neural network for the classification of Human activity. The classification is done using only frequency domain feature. A Triaxial accelerometer sensor, housed in a chest worn sensor unit, has been used for capturing the acceleration of the movements associated. All the three axis acceleration data were collected at a base station PC via a CC2420 2.4GHz ISM band radio (zigbee wireless compliant), processed and classified using MATLAB. A fast training algorithm i.e. Levenberg-marquardt algorithm was used for training. The designed network (with 2 hidden layers) is giving a mean classification rate of 91.82 + 0.047% and 83.96 + 0.118 % for training and test data sets respectively. The mean classification rate with neural network classifier is also an improved one as compared to the previous results without the neural network classifier. The classification accuracy and use of wireless sensor node makes the system compatible to use in ubiquitous home and health environment.

[Pre08] presented a real-time hand gesture recognition system, with a mobile phone, which identifies relevant parts in the continuous sensor data stream, and classifies them to the most probable gesture. He decided to create an automatic segmentation method without any buttons, which results more natural interaction. The mobile phone is used to capture accelerometer data, transmit it via Bluetooth to a nearby computer, where the data is analysed and gesture classification is performed in quasi real-time. Two distinct pattern recognition methods are compared, SVM and HMM. The results showed that both methods performed almost equally well. SVM achieved an average recognition rate of 96% and HMM reached 97.6%.

In [JM09], they address the problem of the recognition of isolated, complex, dynamic hand gestures. The goal was to provide an empirical comparison of two state-of-the-art techniques for temporal event modelling combined with specific features on two different databases. The models proposed are the HMM and Input/Output Hidden Markov Model (IOHMM), implemented within the framework of an open source machine learning library (www.torch.ch).

Accelerometer-based gesture recognition with a Nintendo Wii controller is explored by [SPHB08]. In their work, the sensor data is transmitted via Bluetooth from the controller device to a nearby PC, where the signal is processed. They also released the open-source Java library wiigee, which facilitates training and recognition of hand gestures, performed with a Wii controller.

[SZL+10] uses accelerometer-embedded mobile phones to monitor one's daily physical activities for sake of changing people's sedentary lifestyle. In contrast to the previous work of recognizing user's physical activities by using a single accelerometer-embedded device and placing it in a known position or fixed orientation, this work intends to recognize the physical activities in the natural setting where the mobile

phone's position and orientation are varying, depending on the position, material and size of the hosting pocket. By specifying 6 pocket positions, this work develops a SVM based classifier to recognize 7 common physical activities. By introducing an orientation insensitive sensor reading dimension, they boost the overall F-score from 91.5% to 93.1%. With known pocket position, the overall F-score increases to 94.8%.

In [NH09] they evaluate the various gesture recognition algorithms currently in use, after which the most suitable algorithm was optimized in order to implement it on a mobile device. Gesture recognition techniques studied include hiddenMarkov models, artificial neural networks and dynamic time warping. A dataset for evaluating the gesture recognition algorithms was gathered using a mobile device's embedded accelerometer. The algorithms were evaluated based on computational efficiency, recognition accuracy and storage efficiency. The optimized algorithm was implemented on the mobile device to test the empirical validity of the study.
Initially a 4-state HMM was used, but the recognition accuracy of the HMM proved to be too low to be able to accurately compare themodel with the other algorithms. Using 8 states for the HMM increases the recognition accuracy to above 90%, which is approximately equal to the accuracy obtained with the other algorithms.
For the Neural networks, a total of 72 of the 80 samples were correctly classified, for an overall accuracy of 90%.
The training and recognition procedures in Dynamic time warping DTW are potentially much faster than other techniques used for gesture recognition, such as HMMs and ANNs. Since there are no training steps or preprocessing required for the DTW algorithm, the raw gesture data could be evaluated directly. With the DTW algorithm, a total of 77 of the 80 samples were correctly classified, for an overall accuracy of 96.25%. This compares very well with the HMM algorithm with 8 states, and the DTW algorithm is therefore considered sufficiently accurate as to implement it. The gesture recognition algorithm was ported to the mobile device by making use of Nokia's Open C platform.

[Kli09] implemented a gesture recognition application for the iPhone and its core features have been consolidated into a reusable Objective-C library. The iPhone' s built-in accelerometer is used to capture sensor data generated by hand movements. Discrete hidden Markov models, which work in combination with a vector quantiser, are the core of the recognition system. Various tests have been conducted in order to optimise certain application specific values, such as the codebook size and the initial prototype vectors. The final system reaches recognition results between 80 to 95 %.

LiveMove by AiLive (http://www.ailive.net/) is a commercial software product, which provides comparable functionality. It is aimed for game developers and provides gesture recognition methods for several accelerometer-equipped game controllers. Based

on the Wiimote they developed a gesture recognition that employs state of the art recognition methodology such as HMM, filters and classifiers, and aim to optimise hand gesture recognition for the Wiimote.

[YA99] presented an algorithm for extracting two-dimensional motion fields of objects across a video sequence and classifying each as one of a set of a priori known classes. An application of the algorithm is in sign language recognition where an utterance is interpreted based on, for example, hand location, shape, an motion. The performance of the algorithm is evaluated on the task of recognising 40 complex hand gestures of American Sign Language (ASL). They used Time Delay Neural Networks (TDNN) to classify the hand gestures, based on motion trajectories.

[KKM+05] study accelerometer-based gesture recognition for controlling a television, a video recorder and lighting in a design environment. For a sensing device, they utilise a so-called SoapBox (Sensing, Operating and Activating Peripheral Box), and use HMM for the recognition.

In the research of [eS05], tests were done using computational methods to recognise the conducting gestures of a maestro, as timing, dynamic and expressive gestures from both hands. Taking advantage of image processing, tempo and dynamic were extracted using analysis of the vertical velocity from the right hand. The conducting gesture recognition system was implemented using Elman and T-CombNET neural network structure after an image processing stage.

[SMC09] presents a system aimed to serve as the enabling platform for a wearable assistant. The method observes manipulations from a wearable camera and classifies activities from roughly stabilized low resolution images (160x120 pixels) with the help of a 3-level Dynamic Bayesian Network and adapted temporal templates.

In [LRT+] a physical activity (PA) recognition algorithm for a wearable wireless sensor network using both ambulatory electrocardiogram (ECG) and accelerometer signals is proposed. First, in the time domain, the cardiac activity mean and the motion artifact noise of the ECG signal are modeled by a Hermite polynomial expansion and principal component analysis, respectively. A set of time domain accelerometer features is also extracted. A support vector machine (SVM) is employed for supervised classification using these time domain features. Second, motivated by their potential for handling convolutional noise, cepstral features extracted from ECG and accelerometer signals based on a frame level analysis are modeled using Gaussian mixture models (GMM). Third, to reduce the dimension of the tri-axial accelerometer cepstral features which are concatenated and fused at the feature level, heteroscedastic linear discriminant analysis is performed. Finally, to improve the overall recognition per-

formance, fusion of the multi-modal (ECG and accelerometer) and multi-domain (time domain SVM and cepstral domain GMM) subsystems at the score level is performed. The classification accuracy ranges from 79.3% to 97.3% for various testing scenarios and outperforms the stateof-the-art single accelerometer based PA recognition system by over 24% relative error reduction on their 9-category PA database.

Bayesian networks are widely used to perform classification tasks, with the following advantages.

- Based on probability theory.
- Not a black box approach.
- Allows rich structure.
- Can mix expert opinion and data to build models.
- Backwards reasoning - in addition to predicting outputs given inputs, we can use output values to infer inputs.
- Support for missing data during learning and classification.

Based in some of the works cited above, and due to its particular characteristics and properties we choose Dynamic Bayesian Network models to form the basis of the pattern recognition process, for iBombeiro.

2.2.2 Feature Pre-Processing

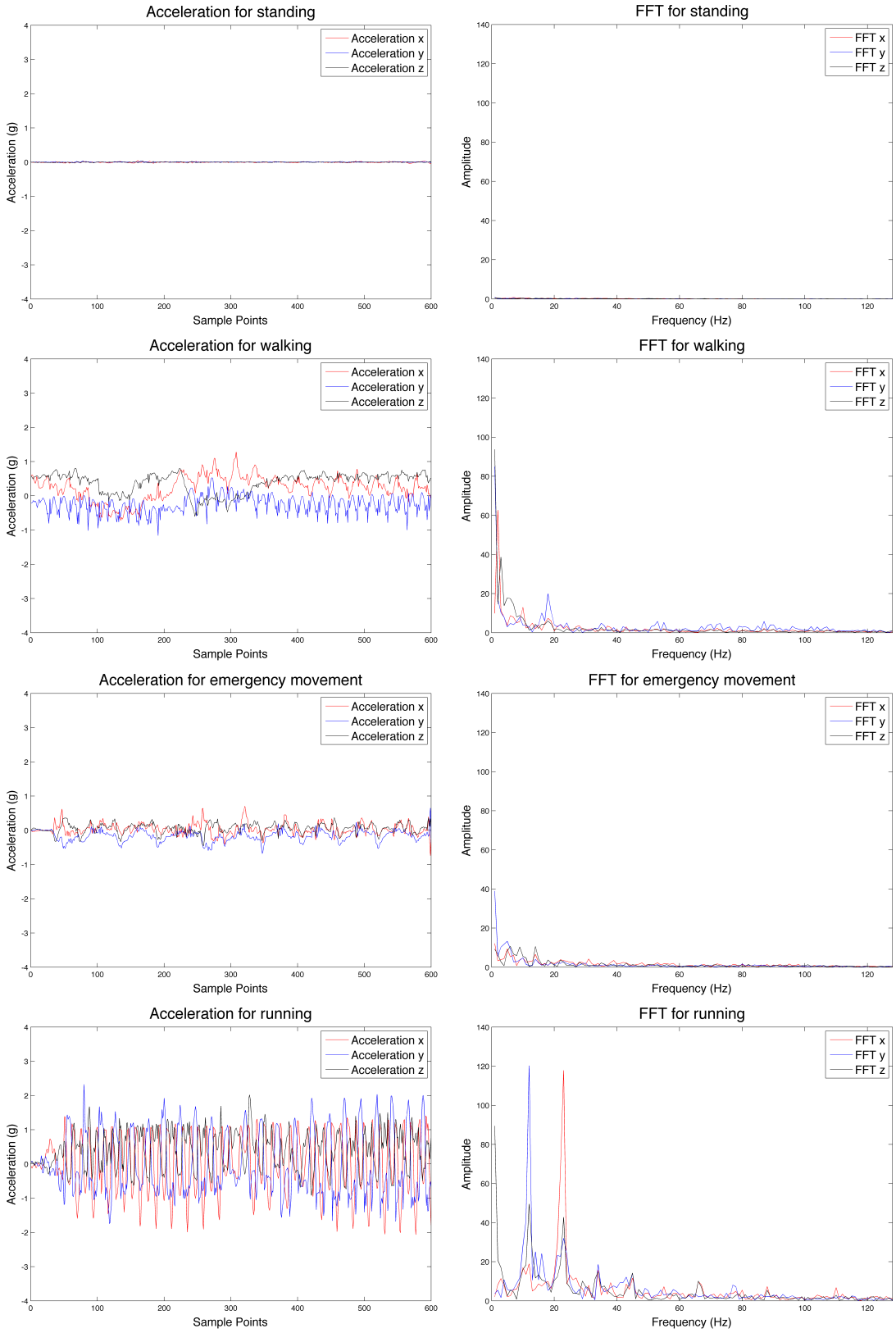
Any classification method uses a set of features or parameters to characterize each object, where these features should be relevant to the task at hand.
We consider here methods for supervised classification, meaning that a human expert both has determined into what classes an object may be categorized and also has provided a set of sample objects with known classes. This set of known objects is called the training set because it is used by the classification programs to learn how to classify objects.

In this work a number of summary statistics were derived from the accelerometer data. These statistics (henceforth referred to as "features") serve the purpose of presenting the accelerometer data in ways believed to hold greater discriminatory power between movements and context, compared to the raw acceleration values. Such features serve as the data space on which the proposed method performs the classification. Based in previous work [KAQ+10, KSAD12, KD12], we decided to apply the fourier transform to the acceleration data, read from iPhone sensors.

The Fourier transform is among the most widely used tools for transforming data sequences and functions (single or multi-dimensional), from what is referred to as the time domain to the frequency domain. Any continuous, periodic function can be represented as a linear combination of sines and cosines. A sine is a function of the form: $Asin(2\pi\omega t + \phi)$, where A is the amplitude, ω is the frequency measured in cycles (or periods) per second, and ϕ is the phase, which is used for getting values other than 0 at t = 0. A cosine function has exactly the same components as the sine function, and can be viewed as a shifted sine (or more accurately - a sine with phase $\frac{\pi}{2}$).
The result of applying the Fourier transform to a function is called the frequency spectrum or the power spectrum of the function.
The Discrete Fourier Transform (DFT) is the equivalent of the continuous Fourier Transform for signals known only at instants separated by sample times T (i.e. a finite sequence of data).
The DFT is defined as:

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{\frac{-i2\pi nk}{N}}, k = 0..N-1 \tag{2.1}$$

Where $X(K)$ represents an array of complex frequency-domain data, $x(n)$ an array of complex time-domain data, n an index of time steps, k an index of frequency spectral lines and N represents the size of the data arrays (i.e. the size of the sequence).
DFT decomposes a sequence of signals into a series of components with different frequency or time intervals. This operation is useful in many fields, but in most cases computing it directly from definition is too slow to be practical.
One of the most appealing aspects of the DFT is the existence of an eficient procedure for calculating it, using $Nlog_2N$ complex operations, rather than N^2 operations required for the naive algorithm. The algorithm for fast DFT, is known as the FFT (the Fast Fourier Transform). It takes advantage of symmetry properties of the complex roots of unity, and uses repeated clever partitioning of the input sequence into two equally long subsequences, each of which can be separately (and quickly) processed. In order to take full advantage of the repetitive partitioning into equal two parts, the original sequence needs to be of length or periodicity which is a power of 2.
In Figure 2.4 the signals for the acceleration values, for each action, and their FFT is shown.


Acceleration for standing
Acceleration x
Acceleration y
Acceleration z
Acceleration (g)
Sample Points
FFT for standing
FFT x
FFT y
FFT z
Amplitude
Frequency (Hz)
Acceleration for walking
Acceleration x
Acceleration y
Acceleration z
Acceleration (g)
Sample Points
FFT for walking
FFT x
FFT y
FFT z
Amplitude
Frequency (Hz)
Acceleration for emergency movement
Acceleration x
Acceleration y
Acceleration z
Acceleration (g)
Sample Points
FFT for emergency movement
FFT x
FFT y
FFT z
Amplitude
Frequency (Hz)
Acceleration for running
Acceleration x
Acceleration y
Acceleration z
Acceleration (g)
Sample Points
FFT for running
FFT x
FFT y
FFT z
Amplitude
Frequency (Hz)

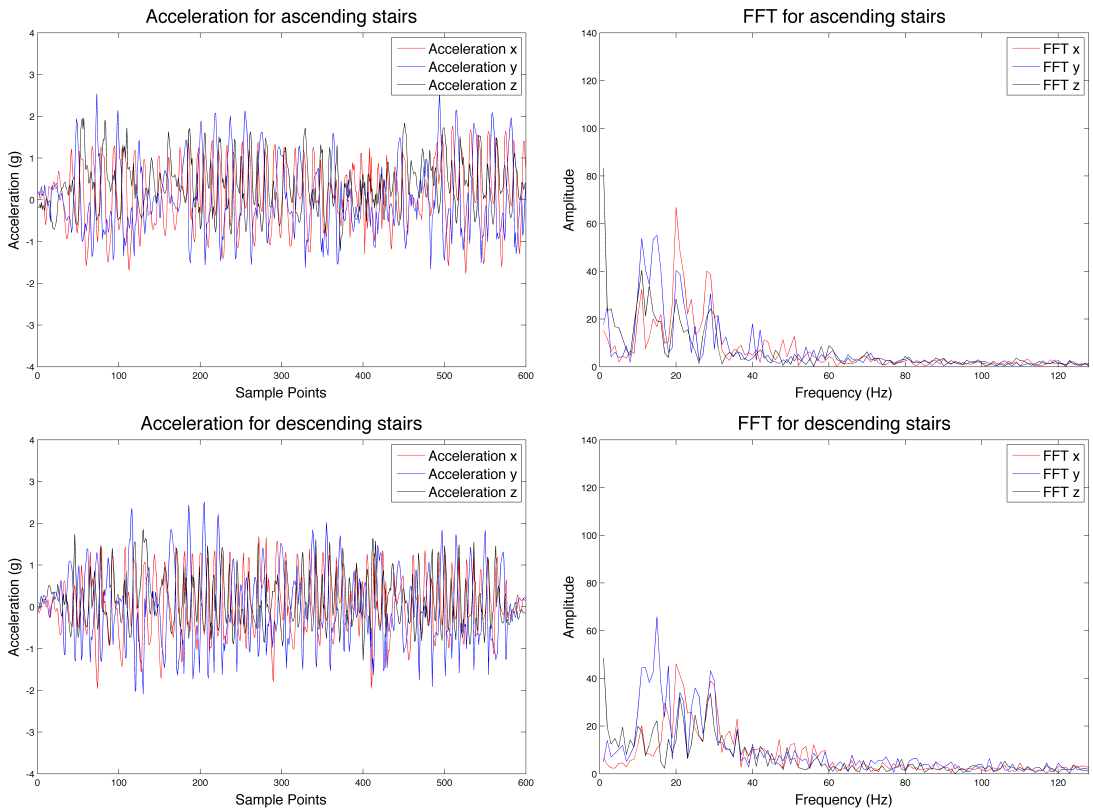


Figure 2.4: User Acceleration and FFT for specified actions

2.2.3 Action model

Probabilistic networks, also known as Bayesian networks (BN's), belief networks or causal networks are already well established as representations of domains involving uncertain relations among a group of random variables. A Bayesian Network is a specific type of graphical model that is represented as a Directed Acyclic Graph (DAG). Nodes in DAG are graphical representation of objects and events that exists in the real world, and are usually termed variables or states. Causal relations between nodes are represented drawing an arc (edge) between them. For any given edge between the variables (nodes) X and $X2$, if there is a causal relationship between the variables, the edge will be directional, leading from the cause variable to the effect variable. Figure 2.5(a) illustrates how X and Y are conditionally independent, given variable Z (note that this is not the same as saying that X and Y are tottaly independent. For each variable in the DAG there is probability distribution function (pdf), which dimensions and

definition depends on the edges leading into the variable.

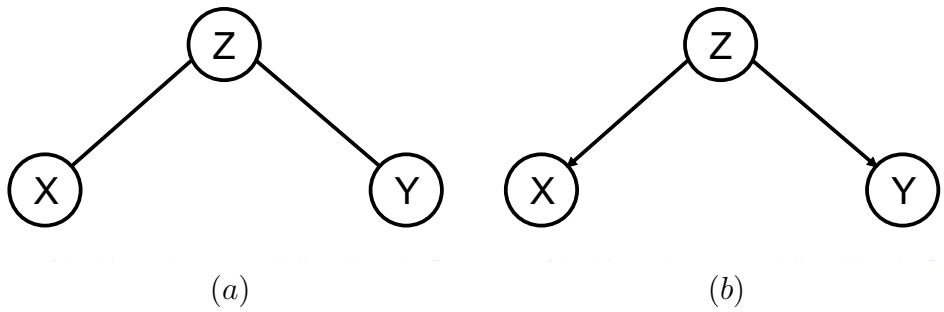


Figure 2.5: User Acceleration for (a) A graphical model illustrating condicional independence between variables X and Y given the variable Z, (b) Simple Bayesian Network

In short, a graphical mode consists of variables (nodes) $V = 1, 2...K$ with a set **D** of dependencies (edges) between the variables and set **P** of probability distribution functions for each variable.
BN's can be defined as aspecial case of more general class called graphical models in which nodes represent random variables, and the lack of arc represents conditional independence assumption between variables. Figure 2.5(b) illustrates a simple Bayesian Network.
BN's do not provide direct mechanism for representing temporal dependencies. In attempting to add temporal dimension into the BN model various approaches has been suggested. Frequent names used to describe this new dimension in BN models are "temporal" and "dynamic". Dynamic Bayesian Networks should be a name of a model that describes a system that is dynamically changing or evolving over time.

Probability theory is based on three basic axioms:

1. $0 \leqslant P(X) \leqslant 1$
2. $P(X) = 1$ if and only if X is certain
3. If X and Y are mutually exclusive, then $P(X \cup Y) = P(X) + P(Y)$

and a fundamental rule of probability calculus:

$$P(X, Y) = P(X|Y)P(Y) \tag{2.2}$$

where $P(X, Y)$ is the probability of the joint event $X \cap Y$ [MP01]

Bayes' rule is a rigorous method for interpreting evidence in the context of previous experience or knowledge.
The general form of the Bayes' Rule can be written as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.3}$$

Where:
$P(A|B)$ is called the posterior
$P(B|A)$ is the likelihood of observations given the model.
$P(A)$ is the Prior
$P(B)$ is the evidence (used as a normalization factor).

In essence, Bayes' rule is used to combine prior experience (in the form of a prior probability) with observed data (in the form of a likelihood) to interpret these data (in the form of a posterior probability). This process is known as Bayesian inference.
Bayesian inference is not guaranteed provide the correct answer. Instead, it provides the probability that each of a number of alternative answers is true, and these can then be used to find the answer that is most probably true. In other words, it provides an informed guess.

In iBombeiro, the inicial prior (for $t = 0$) is an uniform prior. The inicial prior is a matrix of order 1 x N where all the values are $1/N$ (N is the number of states).
For $t > 0$, $prior = posterior(t = t - 1)$

2.2.3.1 iBombeiro Action Recognition Model

Graphical models are a marriage between probability theory and graph theory. They provide a natural tool for dealing with two problems that occur throughout applied mathematics and engineering - uncertainty and complexity - and in particular they are playing an increasingly important role in the design and analysis of machine learning algorithms. Fundamental to the idea of a graphical model is the notion of modularity - a complex system is built by combining simpler parts. Probability theory provides the glue whereby the parts are combined, ensuring that the system as a whole is consistent, and providing ways to interface models to data. The graph theoretic side of graphical models provides both an intuitively appealing interface by which humans can model highly-interacting sets of variables as well as a data structure that lends itself naturally to the design of efficient general-purpose algorithms[Jor99]

Bayesian models provide a powerful language for describing and evaluating hypotheses about perceptual behaviors.
The Bayesian model for iBombeiro will be developed using a formalism known as

Bayesian Programming, consolidated by [BLS08].
In Bayesian Programmming the necessary information for completely specifying a probabilistic model consists of:

1. The relevant variables;
2. The conditional dependencies between these variables;
3. The parametric forms of the associated probability distributions;
4. Their associated parameter values.

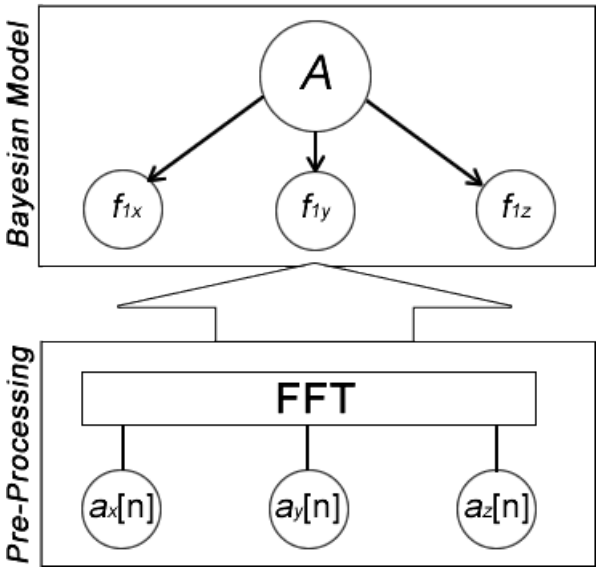


Figure 2.6: iBombeiro Bayesian Model. For each a[n] an independent FFT is computed

A Bayesian Program (BP) contains two parts:

- a description that is the probabilistic model of the studied phenomenon or the programmed behaviour;
- a question that specifies an inference problem to be solved using this model.

A description contains two parts:

- a specification part that formalizes the modeler's knowledge;
- an identification part where free parameters are learned from experimental data.

Finally, the specification is constructed from three parts:

- a selection of relevant variables to model the phenomenon;
- a decomposition, whereby the joint distribution on the relevant variables is expressed as a product of simpler distributions exploiting conditional independence between variables;
- the parametric forms in which either a given mathematical function or a question for another BP is associated with each of the distributions appearing in the decomposition.[CDB10]

In our model, and reporting to Figure 2.7 we have a stream of data, in form of acceleration $ax[n]$, $ay[n]$ and $az[n]$, where an independent FFT is performed in each acceleration. We can treat the acceleration variables independently because we use the acceleration from the Device Motion (see section 2.3.1.1). It means that the acceleration does not have the component of gravity, so the values of the accelerations are not dependent (i.e. if we move the acceleration in $Y - axis$ the value of the acceleration of the $X - axis$ will not change. Computacionally this gives a major advantage, otherwize cumputacional weight would be high.

The Figure 2.7 shows the Bayesian program used in iBombeiro.

Bayesian Program: iBombeiro Action Recognition Model

program
- *description*
  - *specification*
    - **Variables:**
    - $f_n \in \mathcal{F}$: First fourier coeficient variables.
    - A : Action variable.
    - **Decomposition:**
    - $P(A, f_1, \cdots, f_n) = P(f_x|A)P(f_y|A)P(f_z|A)P(A)$
    - **Formulation:**
    - $P(A)$: *Stochastic Matrix.*
    - $P(f_n|A)$: *Gaussian Distribution.*
  - **Identification**: Gaussian parameters μ and σ based on training dataset Ω.
- **Question:** $P(A|f_n)$ answered using *Maximum A Posteriori*, a method for Bayesian inference.

Figure 2.7: Bayesian Program: iBombeiro Action Recognition Model

Reporting to the Bayesian Program in Figure 2.7 we start by defining the relevant variables for the problem.

- $f_n \in \mathcal{F}$ is a random variable that denotes the first fourier coeficient calculated independently for the acceleration measurements, for all the axis and for all the classes, and whose domain is real.

- $\zeta \equiv \{Walking, Standing, Running, Upstairs, Downstairs, Mov1\}$ is a random variable denoting the chosen actions and it can have six possible diferent states.

In the decomposition we have the expression of the joint probability distribution as the product of simpler distributions, i.e. decompose join probabilities in a product of simpler conditional probabilities where the independence of the variables is explored. Decomposition is often assisted by graphics.
In this case, the decomposition of Figure 2.7, it establishes the dependencies according to Figure 2.6.

In the formulation step, the kind of parametric probability function that is adequate to represent the distribution of the decomposition is chosen.
As the domain of f_n is real, it is common to define the distribution as a gaussian density, moreover activities are expected to be repeatable therefore a normally distributed
So, for iBombeiro Bayesian Model we will have (as shown in Equation 2.4), for each action, the following gaussian destribution.

$$P(f1_x|\xi) = \begin{Bmatrix} N(0.2015, 0.1115), for & \xi = 1 \\ N(8.8507, 4.6458), for & \xi = 2 \\ N(2.9079, 1.3355), for & \xi = 3 \\ N(19.2620, 3.9258), for & \xi = 4 \\ N(14.2720, 4.7609), for & \xi = 5 \\ N(12.9820, 3.4223), for & \xi = 6 \end{Bmatrix}, P(f1_y|\xi) = \begin{Bmatrix} N(0.1441, 0.0451), for & \xi = 1 \\ N(6.3170, 3.1266), for & \xi = 2 \\ N(5.3208, 2.7935), for & \xi = 3 \\ N(20.3560, 5.1274), for & \xi = 4 \\ N(15.1590, 5.9118), for & \xi = 5 \\ N(15.6460, 4.4806), for & \xi = 6 \end{Bmatrix}$$

$$P(f1_z|\xi) = \begin{Bmatrix} N(0.1023, 0.0524), for & \xi = 1 \\ N(12.4900, 5.6156), for & \xi = 2 \\ N(3.0844, 1.0834), for & \xi = 3 \\ N(15.4830, 5.0610), for & \xi = 4 \\ N(12.5210, 4.2820), for & \xi = 5 \\ N(10.1690, 3.0088), for & \xi = 6 \end{Bmatrix} \tag{2.4}$$

where $1 = Standing$, $2 = Walking$, $3 = Emergency\ Movement$, $4 = Running$, $5 = Ascending\ stairs$ and $6 = Descending\ stairs$.

Figure 2.8 shows the gaussian distribution for each of the actions, where the gaussian parameters μ and σ are based on the training set.

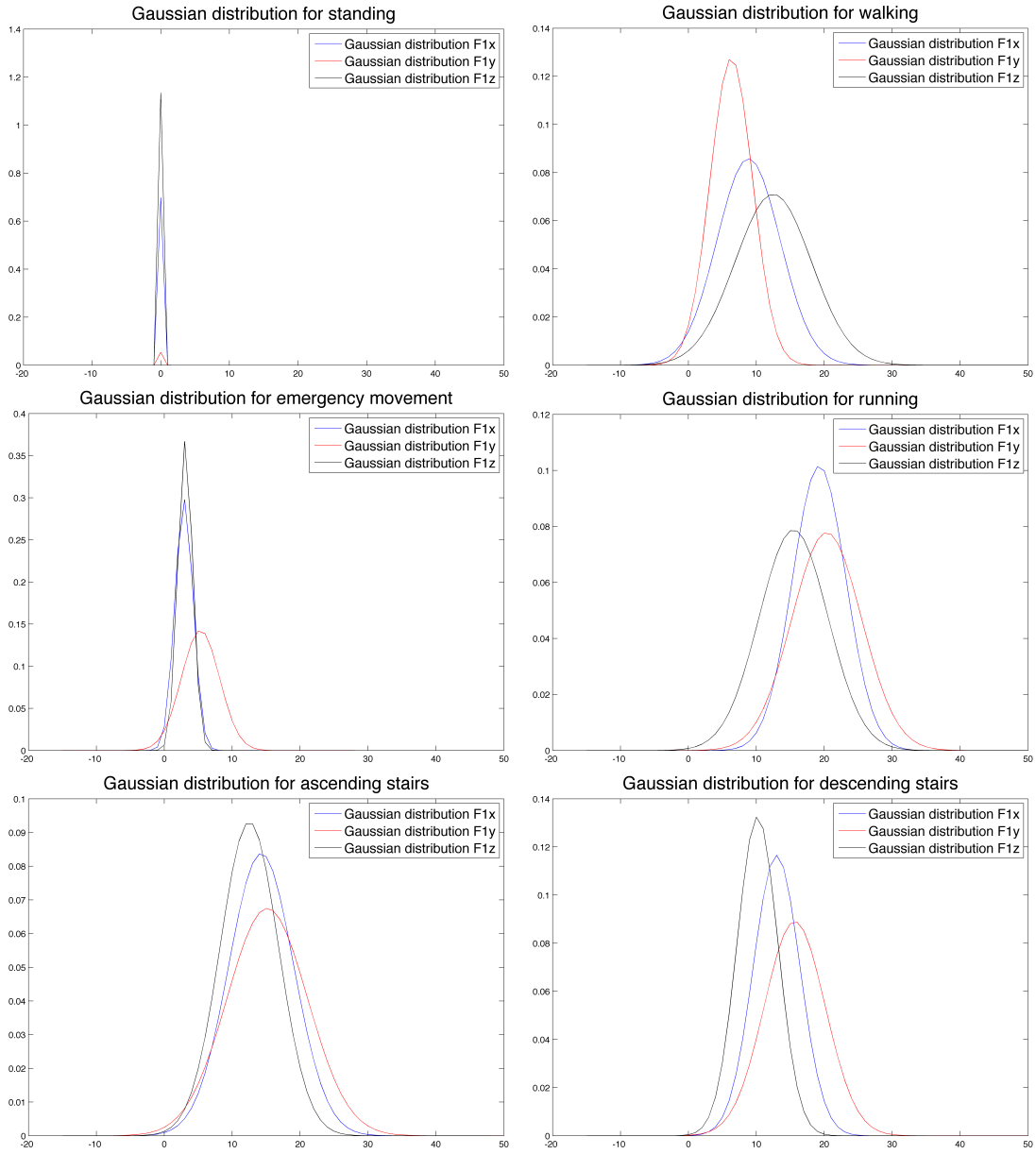


Figure 2.8: Gaussian distribution for specified actions

As for the question, "how can we maximize the posterior distribution?", it is answered wih MAP (Maximum A Posteriory).

$$\hat{A}_{MAP} = argmax_A(P(f_x|A)P(f_y|A)P(f_z|A)P(A)) \tag{2.5}$$

By replacing the prior at instant t, by the posterior calculated at instant $t = t - 1$, as the number of iterations advances, the MAP is expected to converge for a given state.

2.2.3.2 Extended iBombeiro Recognition Model

One major update to iBombeiro would be the development of a personalised model. With this personalisation each firefighter would have his own personal training dataset recorded on a server. The firefighter could choose an iD from the iPhone and his data would be downloaded, where the configuration file would have a record of all his personal training parameters. That personalisation of the iBombeiro app is expected to increase the acuraccy of the present results.

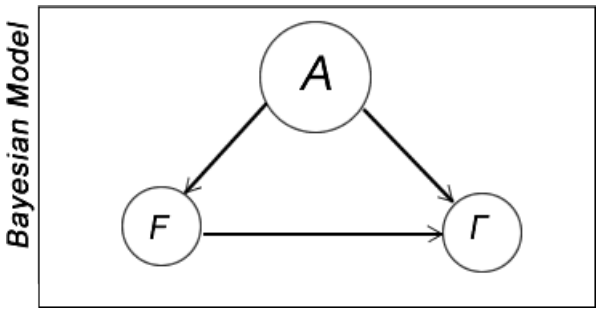


Figure 2.9: Extended iBombeiro Bayesian Model.

The Figure 2.9 presents the Bayesian model for personalised iBombeiro. The feature node it's a supernode, comprising different variables, independently distributed. The Figure 2.10 shows the Bayesian Programming for the Extended iBombeiro Recognition Model. This model was not implemented due to time constrains, therefore in this section we only formulate the model, placing the implementation as future work. The decomposition of the new Bayesian program presents us with a new variable, one whose states denote each person identity. For the extended iBombeiro model, the proposed decomposition of Figure 2.10 establishes the dependencies according to Figure 2.9, where the new learning will deal with multivariate distribution learning.

Learning Personalised iBombeiro Action Data As for the learning of the personalised model, the firefighter could train his own set of data. The process starts with the firefighters identifying himself and the desired action on the iPhone. If the action already exists on the server, the app will ask if he wants to overwrite existing training sets. The app will wait a few seconds until detecting the firefighter is standing and ready for performing the action. Then, a beep sound is issued to indicate the beginning of the training session. The training can be done either by assigning a predefined time to the action to be done, or by applying a method to verify the training distribution convergence.
By using the latter we expect have more assertive results.
System will be capable to computing a gaussian, in specific time intervals. If the difer-

Bayesian Program: Extended iBombeiro Action Recognition Model

program

description

specification

Variables:

$\gamma \in \Gamma$: person identity

$f_n \in \mathcal{F}$: First fourier coeficient variables.

A : Action variable.

Decomposition:

$P(A, \gamma, f_1, \cdots, f_n) = P(F|\gamma)P(F, \gamma|A)P(A)$

Formulation:

$P(A)$: *Stochastic Matrix.*

$P(f_n|\gamma)$: *Gaussian Distribution.*

$P(f_n, \gamma|A)$: *Kernel of Gaussian Distributions.*

Identification: Gaussian parameters μ and σ based on training dataset Ω.

Question: $P(A|f_n, \gamma)$ answered using *Maximum A Posteriori*, a method for Bayesian inference.

Figure 2.10: Bayesian Program: Extended iBombeiro Action Recognition Model

ence between the computed gaussian and the existing trained one is over a certain threshold, the learning continues, and after the specified time, a new training distribution is computed. If the diference between the gaussians is below a certain threshold the training phase can stop, an event signaled by another selective sound.
We can use Kullback-Leibler (KL) information, to infer about the convergence of the gaussians.
The Kullback-Leibler (KL) information is a fundamental quantity in probability and statistics that measures the similarity of two distributions. Suppose we have two models defined on the same sample space, one with with distribution F and a second one with distribution G.
Then the expectation, for a discrete variable

$$I^{KL}(F;G) = \sum f(x_i) log \frac{f(x_i)}{g(x_i)} \tag{2.6}$$

is the Kullback-Leibler information of G with regard to F. The new training set for that action/firefighter is uploaded and recorded to a server.
In the end of the training phase, a beep will warn the firefighter that the traning of the specified action is over.

The Bayesian program of the personalised ibombeiro, as formulated, enables not only a better accuracy by providing a personalised training dataset for each actor, but also another potentially innovative application. It will allow making inference about the identity of a given person, based on observed actions. The integration of this new system could bring enormous benefits to the network, an issue which is discussed latter.

2.2.4 Experimental setup in MATLAB

The experimental setup, with data collected from the iPhone (see Section 2.11), was developed in MATLAB, in order to create the classification model. First a GUI was developed in MATLAB, in order to study all the signals returned from the sensors. Then we developed a classifier in MATLAB.

2.2.4.1 Data Visualization in MATLAB

A Graphical User Interface (GUI) was developed in MATLAB, to perform the visualization of the data collected from the iPhone sensors (Figure 2.3.1).

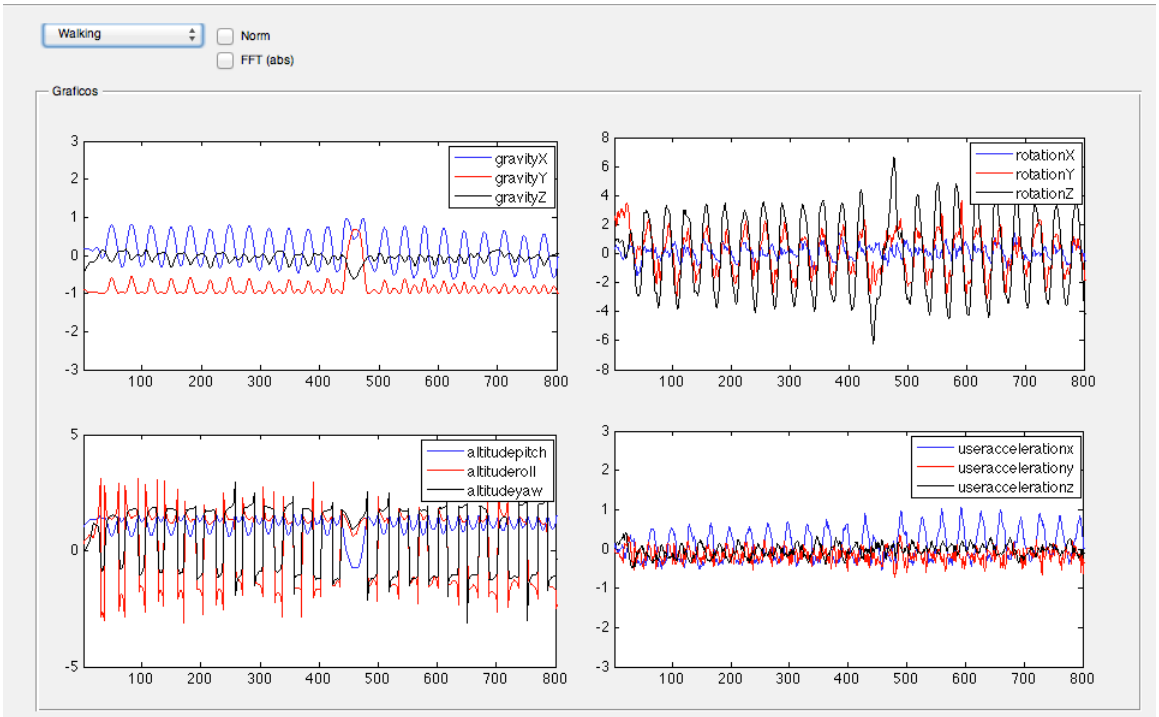


Figure 2.11: Data Visualizer in @Matlab

In the data visualizer it is possible to see:

- The gravity;
- The rotation rate;
- The altitude;
- The user acceleration;

- The FFT for each of the actions.

The GUI was prepared to show the different kinds of data.
In our case we only used the acceleration data, by interpreting this to be the most discriminatory variable. This minimization was also an objective: the reduction of the computacional load, due to the battery and CPU limitations of the device used.
However, for future work, we propose the development of techniques to assess discriminant data types, and the use of more variables in our model, in order to have a good value in the racio factor improvement versus computational load, in the classification.

2.2.4.2 Classifier

There are two phases to constructing a classifier, the training phase and the recognition phase. In the training phase, the training set is used to decide how the parameters ought to be weighted and combined in order to separate the various classes of objects. In the application phase, the weights determined in the training set are applied to a set of objects that do not have known classes in order to determine what their classes are likely to be.

In our work we use the first fourier coeficient, as feature.
Applying more features may bring benefits to the recognition accuracy in the case of computing on the powerful computers. However, when we are trying to implement these features inside the resource and power limited mobile phones, we should try to avoid the features that need complex computing workload, since it consumes much of computing resources and energy, which is critical to the user experience and acceptance of such application.

Our data will be validated using cross-validation.
Cross-validation is primarily a way of masuring the predictive performance of a statistical model, testing the same model on a set of data not used in estimation.

For the training phase our algorithm is the one presenting below.

```
Data: Data from iPhone's sensors
Result: training set
initialization;
N=Number of states;
for i = 0 to N do
    while not at end of data do
        for each acceleration x, y and z do
            read values from current window;
            compute fft;
            save first dynamic characterist (f1) in array;
        end
    end
    compute mean and standard deviation from array of f1;
    save the mean and the standard deviation in a file for current state (training set);
end
```

Algorithm 1: Algorithm for training phase

For the recognition phase the following algorithm is used.

```
Data: Data from iPhone's sensors
Result: Matrix of probabilities
initialization;
N=Number of states;
prior =[1/N 1/N...1/N];
for i = 0 to N do
    while not at end of data do
        for each acceleration x, y and z do
            read values from current window;
            compute fft, and calculate first fourier coeficient characterist (f1);
            compute probabilities from likelihood distributions;
        end
        posterior = distProb_x * distProb_y * distProb_z;
        normalization of the probabilities;
        prior=posterior;
    end
end
```

Algorithm 2: Algorithm for the recognition phase

2.2.4.3 Results and discussions

As we can see in the Table 2.1, the results are interesting, although the action with the worst results was the emergency movement. The observed confusion can be explained due to the fact the generated acceleration features from walking and this movement are very similar. More specifically, both generate back and fourth movements, which in our particular dataset are performed with similar accelerations.
When this movement is performed with a higher cadence, the confusion will translate to the gesture running.
Climbing stairs is other movement that does not have a very good recognition rate.

	Standing	Walking	Emergency Movement	Running	Ascending stairs	Descending stairs
Standing	1.00	0.00	0.00	0.00	0.00	0.00
Walking	0.00	0.9904	0.0096	0.00	0.00	0.00
Emergency movement	0.00	0.9474	0.0526	0.00	0.00	0.00
Running	0.00	0.0312	0.00	0.8594	0.1094	0.00
Ascending stairs	0.00	0.0625	0.00	0.00	0.9141	0.02335
Descending stairs	0.00	0.0779	0.00	0.00	0.2208	0.7013

Table 2.1: Matlab results

For dissemination purposes, we have tested yet another kind of movement, here named Test Movement. Imagining a person standing in the Y-axis, and with the Z-axis pointing to front, the test movement is performed parallel to the Z-axis, with the forearm making an angle of 90 degrees with the arm.

The following table (Table 2.2) shows the results with the introduction of this new action.

With the introduction of the new action (test movement), results have become worst. Detailed analysis found that the two performances carried out for the new action gesture yields completely different signals, which naturally confuses the classifier. This problem is expected to be solved when using the identity variable, previously presented in the extended model in section 2.2.3.2.
Each person has its own dataset, and the identity variable works as a stamp printed in their actions. During the classification process, this discriminatory method, would expectadely improve the classification. In the next chapter it will be presented the iPhone implementation of the classifier.

2.3 Iphone Implementation

In the scope of this work it will be used, for data acquisition and for online classifier, an iPhone 4S.

	Standing	Walking	Emergency Movement	Running	Ascending stairs	Descending stairs	Test Movement
Standing	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Walking	0.00	0.00	0.04785	0.00	0.00	0.00	0.9952
Emergency movement	0.00	0.2632	0.03	0.00	0.00	0.00	0.7068
Running	0.00	0.02343	0.00	0.8594	0.1094	0.00	0.0078
Ascending stairs	0.00	0.0391	0.00	0.00	0.9141	0.0234	0.0234
Descending stairs	0.00	0.00	0.00	0.00	0.2208	0.6753	0.1039
Test Movement	0.00	0.00	0.00	0.00	0.00	0.9186	0.0814

Table 2.2: Matlab results for one more action

2.3.1 iPhone as a device for collecting Data

The fist approach to the problem was to collect sample data, for all the movements and physical activities, previously defined.
A primary app was developed in the iPhone platform, that consists of an operating system and a set of tools developed by Apple, using Objective-C. The iPhone platform targets portable, multi-touch devices and is currently only available on the Apple iPhone, iPod touch and iPad.
Objective-C is a programming language, which is sleeted by Apple for developing the application for iPhone and Mac systems, and it's is defined as a small but powerful set of extensions to the standard ANSI C language. The app *collectData* was able to capture the device motion data with a frequency of 30Hz.

2.3.1.1 Sensors

In the last few years, the fast development in sensor technology made it possible to put small accelerometer sensors into everyday devices. The availability of MEMS (Micro-Electromechanical System) 3-axis linear accelerometers allows for the design of an inexpensive mobile gesture recognition system. A single integrated tri-axial accelerometer provides better accuracy than traditional orthogonally mounted accelerometers, and reduces re-calibration frequency. Wearable inertial sensors are a low-cost, low-power solution to track gestures and, more generally, movements of a person.[Pre08]

The embedded triaxial accelerometer inside the iPhone can continuously sample the experienced accelerations at each sampling interval and produce 3-D acceleration readings $A = (ax; ay; az)$, which are measures of the acceleration experienced in the three orthogonal axes: $X - axis$, $Y - axis$ and $Z - axis$ (Figure **??**)
The gyroscope measures the rate of rotation around the three axes.

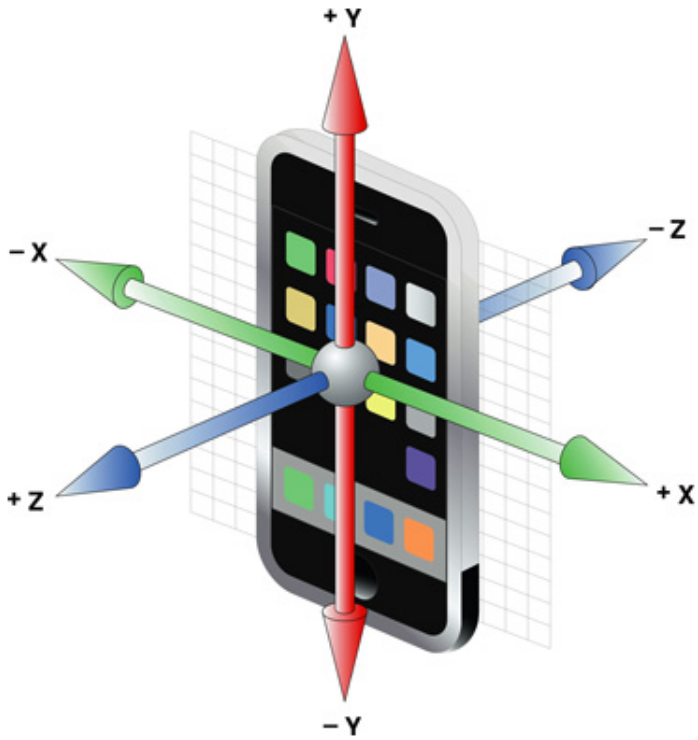


Figure 2.12: The three axes on the iPhone.

The Core Motion framework (CoreMedia.framework) provides a single set of interfaces for accessing all motion-based data available on a device. The framework supports accessing both raw and processed accelerometer data using a new set of block-based interfaces. For devices with a built-in gyroscope, it is possible to retrieve the raw gyro data as well as processed data reflecting the attitude and rotation rates of the device. Core Motion uses unique algorithms to process the raw data it collects, so that it can present more refined information. This processing occurs on the framework's own thread. Core Motion events are represented by three data objects, each encapsulating one or more measurements. [App13]

- A CMAccelerometerData object captures the acceleration along each of the spatial axes, from the three accelerometers.

- A CMGyroData object captures the rate of rotation around each of the three spatial axes, from the iOS device's gyroscope.
- The CMDeviceMotion object encapsulates data from the accelerometer and gyroscope.

  For iOS devices with a magnetometer, the magneticField property contains the magnetic field vector.

A CMMotionManager object is the gateway to the motion services provided by iOS. These services provide an app with accelerometer data, rotation-rate data, magnetometer data, and other device-motion data such as attitude.
Code belows shows how a CMMotionManager object is called.

```
// Start device motion updates
 if ([self.motionManager isDeviceMotionAvailable])
   {
   //update 30 times per second
       [self.motionManager setDeviceMotionUpdateInterval:1.0/30.0];
       [self.motionManager startDeviceMotionUpdatesToQueue:[NSOperationQueue mainQueue]
       withHandler: ^(CMDeviceMotion *deviceMotion, NSError *error)
        {
           CMAttitude *attitude = motionData.attitude;
           CMAcceleration gravity = motionData.gravity;
           CMAcceleration userAcceleration = motionData.userAcceleration;
           CMRotationRate rotate = motionData.rotationRate;
         // handle data here
        }
 }
```

If DeviceMotion is available we define the update interval, $tDeviceMotionUpdateInterval$: 1.0/30.0 for performing device motion updates at 30 Hz. Then we can access the properties of the CMDeviceMotion: A CMAttitude object, that represents a measurement of attitude; a CMAcceleration object, where we can have gravity and user acceleration (the total acceleration of the device is equal to gravity plus the acceleration the user imparts to the device (userAcceleration)) and a CMRotationRate structure that contains data specifying the device's rate of rotation around three axes.

2.3.1.2 Parse and app for collecting data

The app for collecting data is shown in Figure 2.13. After the inicialization of the app, the user can choose the activity to perform. For each activity a file is recorded containing all the information of the sensors. After all the activities were performed the user can push the button End in order to stop the updates from the Device Motion.

Parse (`http://www.parse.com/`) is a cloud app platform for iOS, Android, JavaScript, Windows 8, Windows Phone 8, and OS X.

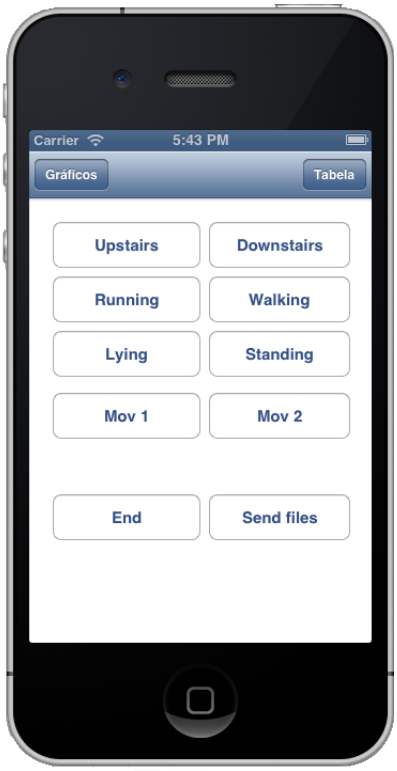


Figure 2.13: iPhone app for collecting data

The Parse platform provides a complete backend solution for mobile applications, totally eliminating the need for writing server code or maintaining servers.
The Parse, in this work, removed the need of connecting the iPhone to a computer, in order to access the files, everytime a new set of data is ready.
A connection with Parse was developed, allowing the synchronization of the files with a single button push (Button send data).

As we can see in Figure 2.14, all the files are available on the Parse cloud.

2.3.2 iBombeiro app

The iBombeiro app was then developed.
iBombeiro is continuously receiving data from device motion. The stream of data is then processed and classified. The resulted state is processed in three diferent ways that will be discussed in the next subsections.
The message for the CCO is constantly trying to be send to a near robot. When the firefighter is standing or lying for more that 30 seconds, the status becomes "danger"

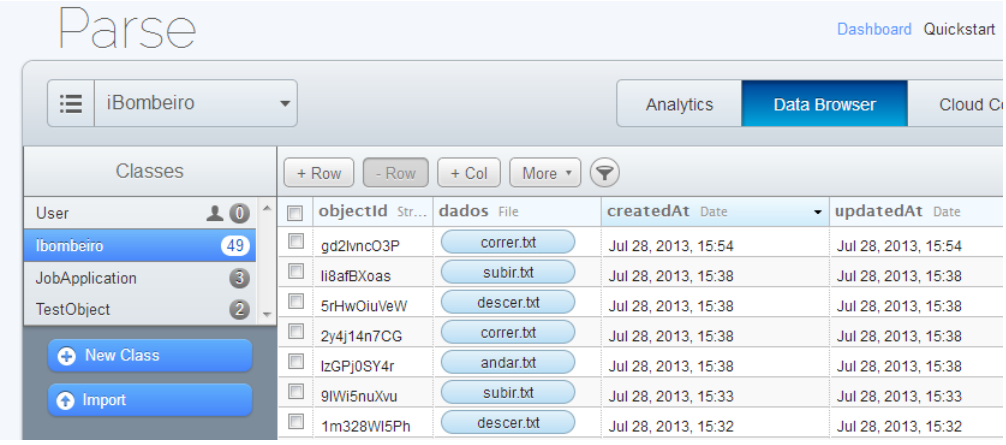


Figure 2.14: Parse cloud, where our files are stored

and an audible alarm is triggered.
If the state of the firefighter change, then a message is send to a WebService with that state (if network available), and that state is also recorded in the iPhone database (to act like a black box). To summarize:

- The iPhone will send a message to a near robot, via socket, in order to publish a topic in a ROS topic, with the state of the firefighter.
- If the state of the firefighter change the new state will be sent to a WebService, previously defined.
- If the state of the firefighter change the new state will be recorded in the iPhone database (black box approach)

In order to have a powefull app, without the system become unresponsive, it's necessary to work with multithreading. We used a core multithreading API available on iOS, Grand Central Dispatch.
Introduced in iOS 4.0, Grand Central Dispatch (GCD) is a BSD-level technology that is used to manage the execution of tasks in the application. GCD combines an asynchronous programming model with a highly optimized core to provide a convenient (and more efficient) alternative to threading.
GCD also provides convenient alternatives for many types of low-level tasks, such as reading and writing file descriptors, implementing timers, and monitoring signals and process events. This means that if there is a failure in some part of the code, either in the database recording process, either in the webservice process or in the interaction with the MANET, the system will be capable of continue, without crash.

2.3.2.1 Classifier implementation in IOS

After the implementation of the classifier in MATLAB the system was portated to the iPhone. The training set obtained in MATLAB was used, so the test phase was no needed to be implemented in iPhone.

To implement the FFT in the iPhone a recent API from apple was used. The vDSP API provides mathematical functions for applications such as speech, sound, audio, and video processing, diagnostic medical imaging, radar signal processing, seismic analysis, and scientific data processing.
The vDSP functions operate on real and complex data types. The functions include data type conversions, FFTs, and vector-to-vector and vector-to-scalar operations.[2]
A sliding window technique (window size of 32 values) is applied to the buffer of acceleration values to classify with an eight first-in first-out policy.
To each sliding window an independent FFT is aplied to the values of acceleration in x, y and z. We do not perform any data-preprocessing because, as mencioned above, Core Motion uses unique algorithms to process the raw data it collects, so that it can present more refined information. The absence of the preprocessing step allows better performance of the app.
This code allows the inicialization of a class, fft, where we create the methods to compute the fast fourier transform.

```
FFT *fft;
fft=[[FFT alloc]init];
```

Then we can call the method FFT, that exists in the fft class to return the first fourier coeficient

```
// return first fourier coeficient for acceleration data in x
fourier1_x=*[fft FFT:accel_x];
```

After the calculation of the fourier coeficient the methods that will handle the probabilities, are called, in order to compute probabilities from likelihood distributions.
In order to avoid the deadlock in bayesian model, due to 0 or 1 probabilities, we also developed a method $preventFlatRegions$, to prevent that a probability is greater than 0.99% and is less than 0.005%.

2.3.2.2 Comunication protocol

In its most basic form, a computer network is three or more computers connected via a communications system for the purpose of sharing data and/or resources, such as a printer. Although the communications systems used to build a computer network can

[2] https://developer.apple.com

vary, by far the most common types are based on the Open Systems Interconnection Basic Reference Model or OSI Model for short.

OSI Model is a standard description or reference model for how messages should be transmitted between any two points in a telecommunication network. Its purpose is to guide product implementors so that their products will consistently work with other products.

The main idea in OSI is that the process of communication between two end points in a telecommunication network can be divided into layers, with each layer adding its own set of special, related functions.
OSI divides telecommunication into seven layers. The layers are in two groups. The upper four layers are used whenever a message passes from or to a user. The lower three layers (up to the network layer) are used when any message passes through the host computer. Messages intended for this computer pass to the upper layers. Messages destined for some other host are not passed up to the upper layers but are forwarded to another host (Figure 2.15).
The OSI seven-layer model underlies every popular networking protocol.

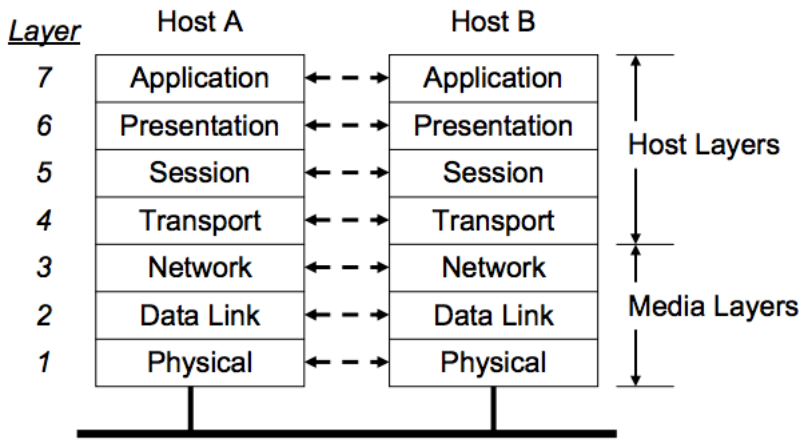


Figure 2.15: The Seven-Layer OSI Model

A MANET is an autonomous collection of mobile users that communicate over relatively bandwidth constrained wireless links. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, i.e., routing functionality will be incorporated into mobile nodes. Significant examples include establishing survivable, efficient, dynamic communication for emergency/rescue operations, disaster relief ef-

forts, and military networks. Such network scenarios cannot rely on centralized and organized connectivity, and can be conceived as applications of MANETs.

The protocol stack of MANET consists of five layers: physical layer, data link layer, network layer, transport layer and application layer. As can be seen, the OSI model's session, presentation and application layers are merged into one section, the application layer in MANET. [Har11, Pea07]

In the scope of the Chopin Project a MANET was developed, so all the robot agents and all the human agents must behave like a ROS node inside the MANET.

ROS (`http://wiki.ros.org/`) is a framework for robot software development, providing libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license. In the Chopin Project, a ROS node will be deployed for each agent, i.e. for each mobile robot and each hand-held device. Each node communicates with other nodes through Wi-Fi.

This assumption leads us to a problem: How can the iPhone be a ROS node inside the MANET?

Sockets A way to simulate the behavior of the iPhone as a ROS node (and since this issue was not envisaged in the initial requirements) was to dispatch messages over socket to a near robot, with a predefined format, in order to communicate with the CCO. A socket is one end-point of a two-way communication link between two programs running on the network. Socket classes are used to represent the connection between a client program and a server program.
For both the Network and Transport Layers, special attention is paid to connectionless- and connection-based communication. One of the major differences between these two forms of transmission is controlling the flow of information between endpoints. It is interesting that the two Layer-4 protocols used today—TCP and UDP (User Datagram Protocol)—are differentiated from each other in the exact same way, with TCP characterized as connection-oriented while UDP is connectionless (no session is established between hosts). TCP guarantees delivery through the use of acknowledgments and sequenced delivery of data. UDP does not guarantee or acknowledge delivery, or sequence data. UDP is fast, has low overhead requirements, and can support point-to-point and point-to-multipoint communication, while TCP is slower, has higher overhead requirements, and only supports point-to-point communication.
UDP sockets were chosen due to project constrains.

In the MANET each actor will behave like a ROS node, capable of communicating through a ROS topic, to be disseminated through all network members. The RosBridge (`http://wiki.ros.org/rosbridge_suite`) is a ROS package that emulates the mes-

sages that arrive via webSocket to the ROS network, as a ROS topic. The Figure 2.16 shows the communication structure of the different actors.

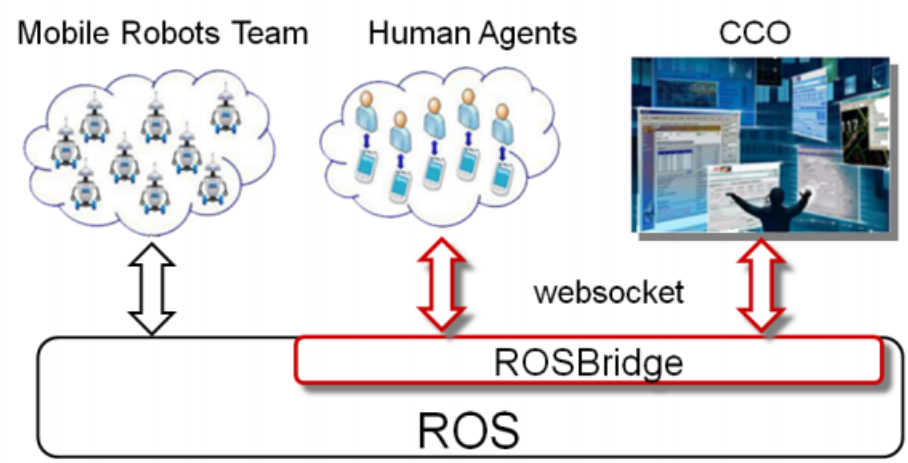


Figure 2.16: Communication structure of the different actors.

In the iPhone, when we want to send messages, the message in published in ROS topic *(/bombeiro1/datapub)*.
To receive data from CCO, the iPhone subscribe a ROS topic (*bombeiro1/datasub*).

In iOS, we used an asynchronous socket networking library, CocoaAsyncSocket (`https://github.com/robbiehanson/CocoaAsyncSocket`). CocoaAsyncSocket supports TCP and UDP. The AsyncSocket class is for TCP, and the AsyncUdpSocket class is for UDP. AsyncUdpSocket is a UDP/IP socket networking library that wraps CFSocket. It works almost exactly like the TCP version, but is designed specifically for UDP. This includes queued non-blocking send/receive operations, full delegate support, run-loop based, self-contained class, and support for IPv4 and IPv6.

As an example we show how to call the function that will send the UDP socket. The message is a string that the robot will publish in the topic *bombeiro1*.

```
    //call the function ROSPublish, and pass the firefighter status (statusString)
    [self ROSPublish:"/bombeiro1" withContent:statusString withType:"std_msgs/String"];

    //function ROSPublish
    - (void)ROSPublish:(char*)topic withContent:(char*)content withType:(char*)msg_type
{
    char end[] = "\xff";
    NSString *str = [NSString stringWithFormat:@"{\"receiver
    \":\"%s\", \"msg\": {\"data\": \"%s\"}, \"type\":\"%s\"}",topic, content, msg_type];
    NSData *strdata = [str dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:YES];
    NSMutableData *data = [NSMutableData alloc];
```

```
    [data appendData:strdata];
    [data appendBytes:end length:sizeof(end)];
    [sendSocket writeData:data withTimeout:-1 tag:2];
}
```

After defining the message, with the appropriate content, we call the function *sendSocket* that will handle with the dispatch of the UDP socket.

Webservices In addition to the communication protocol solution developed above, within the scope of the CHOPIN Project, we also developed a webservice. Thus, it is possible to have the system to operate independently, as for the tests, as for future usage, within the project.

Web services (sometimes called application services) are services (usually including some combination of programming and data, but possibly including human resources as well) that are made available from a business's Web server for Web users or other Web-connected programs. Providers of Web services are generally known as application service providers.
Users can access some Web services through a peer-to-peer arrangement rather than by going to a central server. Besides the standardization and wide availability to users and businesses of the Internet itself, Web services are also increasingly enabled by the use of the Extensible Markup Language (XML) as a means of standardizing data formats and exchanging data. XML is the foundation for the Web Services Description Language (WSDL).

In the scope of this work an webservice for the iOS app was developed (`http://www.ibombeiro.com/webservice.php`). The first step of the creation of the websevice was to create the database and the table (for this purpose only one table was needed, in the webserver (running MySQL and PHP).
The MySQL statement used to create the table was:

```
%criação da base de dados
DROP TABLE IF EXISTS ios;
CREATE TABLE ios (
    id int NOT NULL AUTO_INCREMENT PRIMARY KEY,
    status varchar(255) NOT NULL,
    timestamp varchar(20) NOT NULL
);
```

As the state of the firefighter changes, a message is send to a predefined url, via POST, i.e, the parameters are passed as part of the request body.
The status is then recorded in the mysql table.

```
if(isset($_POST['status'])){
    $status=$_POST['status'];
```

```
    $data=$_POST['data'];
    $query="insert into ios (status, data) values( '".$status."','".$data."')";
    mysql_query($query);
    mysql_close();
}
```

The div [3] that contains the status of the firefighter is refreshed second per second using asynchronous JavaScript and XML (AJAX).

Figure 2.17: Webservice

[3] div – generic flow container (`http://www.w3.org/TR/html-markup/div.html`)

In ios:

```
//function that send status to webservice
-(void) uploadDataToWebService:(NSString *) status
{
    NSString *post =[NSString stringWithFormat:@"status=%@",status];
    NSData *postData = [post dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:YES];
    NSString *postLength = [NSString stringWithFormat:@"%d", [postData length]];
    // Create the request
    NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:
        [NSURL URLWithString:postURL]];
    [request setHTTPMethod:@"POST"];
    [request setValue:postLength forHTTPHeaderField:@"Content-Length"];
    [request setValue:@"application/x-www-form-urlencoded" forHTTPHeaderField:@"Content-Type"];
    [request setHTTPBody:postData];
    NSHTTPURLResponse* urlResponse = nil;
    //process response
}
```

We created an instance of NSURLRequest, assigning it the URL of the webservice, and passing it to the webservice via POST

2.3.2.3 iPhone database: CoreData

Another upgrade to the system was to create a database on iPhone, in order to store the informations about the firefighter actions, for future reference. All the background is already developed enabling rapid integration with future services.

Introduced in iOS 3.0, the Core Data framework (CoreData.framework) is a technology for managing the data model of a Model-View-Controller application. Core Data is intended for use in applications in which the data model is already highly structured. Instead of defining data structures programmatically, the graphical tools in Xcode are used to build a schema representing the data model. At runtime, instances of the data-model entities are created, managed, and made available through the Core Data framework.

- managedObjectContext (of type NSManagedObjectContext) -This is the bridge between the programmer and the managed object model.
- managedObjectModel (of type NSManagedObjectModel) - This is the same concept as a schema in a database. This could represent the tables in a database or the different types of managed objects that it is possible to create in the app database.
  A Managed Object Model define the data we want to store.
  - Entity: This is an individual data object. In a database it'd be a table.

  - Atribute. This is an individual variable stored inside an entity. In a database it'd be a column.

- persistentStoreCoordinator (of type NSPersistentStoreCoordinator) - This is the bridge or the connection between the physical file that stores the data and the application. This bridge will be responsible for managing different object contexts. [GFG12, Nah12]

A database model was created in iOS, to record the status of the firefighter. Below is a code example to record data to the created database model (ibombeiro.sqlite). All The connections and configurations to the database are not shown here.

```
NSEntityDescription *entitydesc=[NSEntityDescription entityForName:@"IBombeiro"
    inManagedObjectContext:context];
NSManagedObject *newData=[[NSManagedObject alloc] initWithEntity:entitydesc
    insertIntoManagedObjectContext:context];
NSDateFormatter* dateFormatter = [[NSDateFormatter alloc] init];
[dateFormatter setDateFormat:@"yyyyMMdd HH:mm:ss"];
NSString *dateString = [dateFormatter stringFromDate:[NSDate date]];
[newData setValue:dateString forKey:@"timestamp"];
[newData setValue:newState forKey:@"coreStatus"];
NSError *error;
if (![context save:&error]) {
   //handle the error
}
```

In the code below first it's necessary to create a $NSEntityDescripton$, $*entitydesc$.

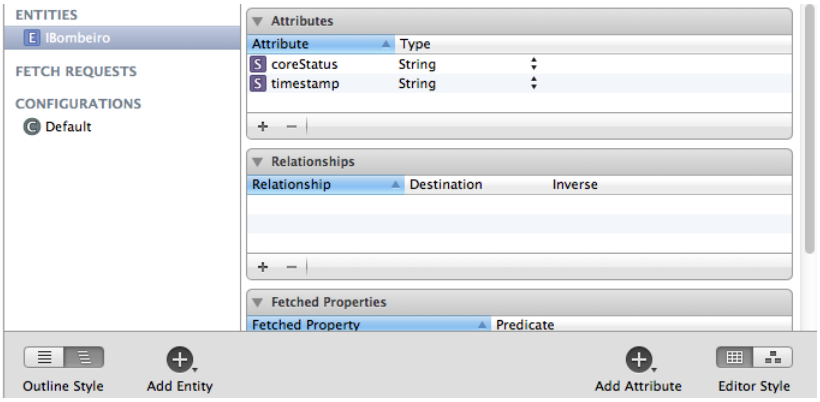


Figure 2.18: IBombeiro entity inside ibombeiro database

An $NSEntityDescription$ object describes an entity in Core Data. It means that we tell the managed object context what entity we want to retrieve the data from (in this case, IBombeiro entity, as shown in Figure 2.18).

Then the $NSManagedObject * newData$ is created. It's necessary associate the new managed object instance with the entity object that defines its properties and with the managed object context that defines its environment.
We can introduce new values to the table by typing $[newDatasetValue : newStateforKey : @"coreStatus"]$;.
$newState$, that is the status of the firefighter, in the current timestamp, is recorded in the IBombeiro entity, in the atribute $coreStatus$.
In order to not overcharge the iPhone (the data is processed in approximately a quarter of second) only will be saved new data when the status of the firefigter change. For

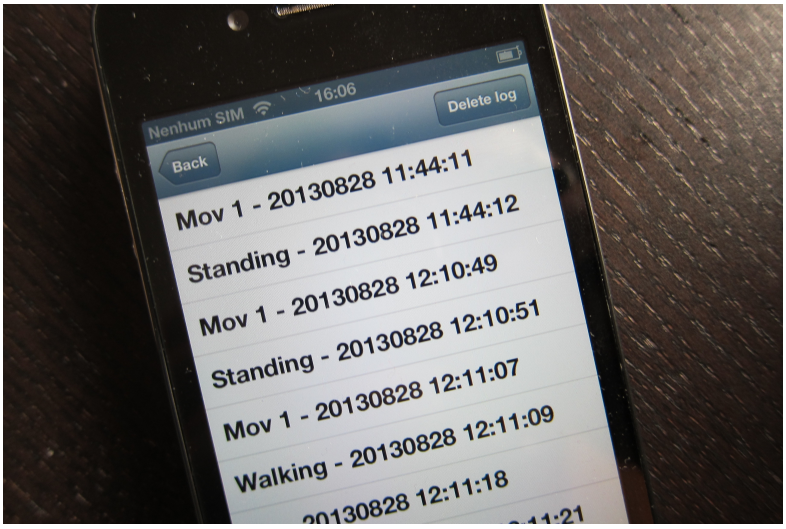


Figure 2.19: Tableview

the visualization of the data, a model of a tableview was added to the app, in order to make requests to the database and show the requests in the form of a table. The *tableview* with the logs showing the diferent status of the firefighter can be acessed within the *iBombeiro* app (Figure 2.19).
There is also a button in the tableview "Delete log", that allows to delete all the records in the IBombeiro entity.

2.3.2.4 Experiments with an iPhone integrated in a manet

in `www.ibombeiro.com/multimedia` it is possible to see the interaction of the iPhone working as a node in the MANET (Figure 2.20).
The videos were recorded earlier so the status is only if the firefighter is standing or moving. In the first video it is possible to see the iPhone sending a message to a ROS topic.

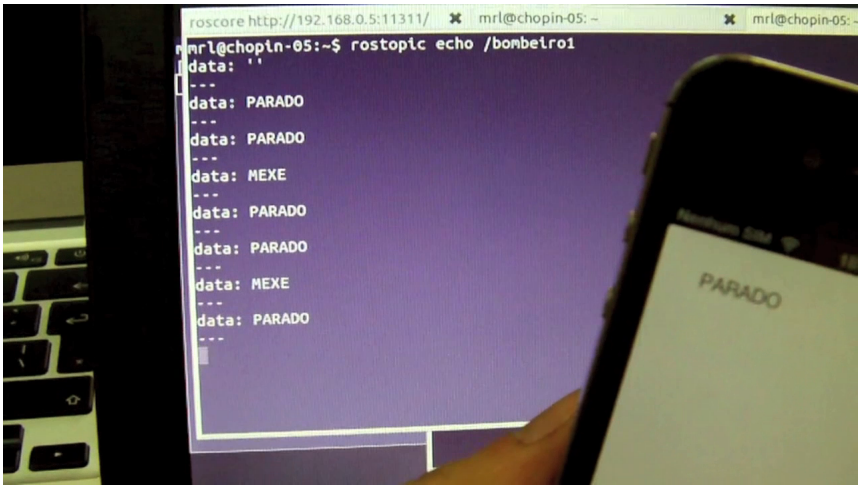


Figure 2.20: Interaction with the MANET - iPhone sending messages

The iPhone also receives mesages from CCO, via Sockets UDP. In the second video, we can see the iPhone receiving a message from the CCO (Figure 2.21).

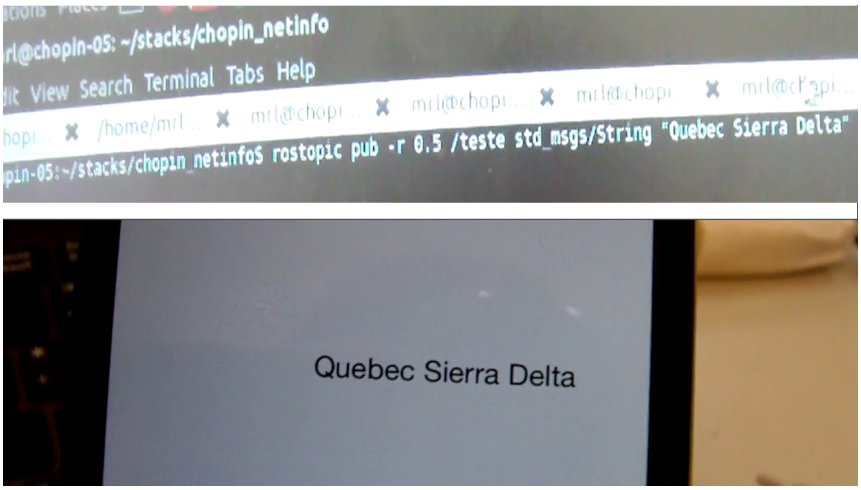


Figure 2.21: Interaction with the MANET - iPhone receiving messages

2.4 Conclusions and future work for iBombeiro

In this work we developed an action recognition model for implementation on a mobile system, for monitoring the state of firefighter, integrated in a mixed team of robotic and human agents. This work was developed in he scope of the CHOPIN Project.

In the system's implementation we used a classification model based on Bayesian techniques, in which inference is based on the *Maximum A Posteriori* (MAP) method.

An extended version of the model is also proposed. Time constrains alone have prevented its implementation, but given the complete formulation, this can be strongly suggested as future work, allowing a personalised version of the iBombeiro.

The model was successfully implemented in iPhone, and has generated promising results. Our approach to the problem had to be simplified, given to major restrictions: battery and CPU power autonomy. Although only the acceleration data and a reduced number of characteristics were used, due to computational limitations, the system has shown interessant results under the proposed conditions, and the models validated using methods of cross-validation. However, the device's orientation in the performance of the emergency gesture was found to have key impact during the classification.

Therefore, if device autonomy restrictions are relaxed, but not discared, and/or the models computationally optimised, we propose exploring additional features in order to improve the model robustness.

Another interesting solution is the potencial usage of an additional accelerometer, attached to the legs or hip in order to have relative accelerations and highly discriminant data, widening the range of actions to be identified.

We also developed a link between the firefighter and the MANET, where the firefighter behave like a ROS node, by sending a message to a near robot. This implementation poses some questions, due to the project constrains. We use UPD sockets to communicate to CCO via MANET, by sending a predefined message to be published as a ROS topic. But UDP sockets are not reliable; there may not be any robots nearby; it is not possible to know whether the message reaches the destination; the robot near can be stuck in an "island". Messages from the CCO to the firefighter, that are received in the same manner (UDP socket), are also liable to not be received.
Another problem is due to the lack of tests concerning multiple robots and firefighters acting as a team, communicating to the CCO inside the MANET.
MANETs suffer from the same limitations as fixed wireless mesh networks, but also are vulnerable to additional challenges resulting from their inherent mobility. Problems will arise, that will need to be resolved.

As alternative to the MANET communications a webservice was developed allowing the system to work independently.

The possibility of the system behave like a "black box", saving all relevant information was also covered.

As future work there are a wide range of improvements and new modules to be developed in the scope of iBombeiro, that can be added to the existing framework. The approach used in iBomeiro was a simplist one. Although the good results achieved in iBombeiro one interesting aspect would be the increase of the number of features. In this scenario rigorous tests should be conducted to conclude about the computational feasibility, for the benefits of the introduction of more features.

Another sugestion is the introduction of positioning with geolocation, taking advantages of of the iPhone features (Geolocation API). This can add another important service to iBombeiro, in areas where geolocation is avaliable.

Other important contribution for future work will be the integration of the developed Bayesian extended model, in order to take advantages of the personalisation by increasing system's accuracy. A corollary of this extended model would be the identification of persons by the way they perform their actions. The possibility of bulding a map with firefighters localization, without prior configuration, would have a great impact within CHOPIN Project.

Action recognition with wearable sensors has the potential to enhance existing applications as well as enable new ones, ranging from personal healthcare and assisted living, to industrial applications, and even entertainment and arts.

Beyond the scope of CHOPIN Project and of elderly care area, where already exists promising advances in the action recognition systems using smartphones, and as been mencioned in section 2.1.1 there are already smartphone applications capable of using GPS positioning to track user's activity. The integration of our work with a system like RunKeeper would allow the user to perform activities without redefining the activity in the smartphone. With an extra kit of sensors, attached to the smartphone, the accuracy of the system could be increased, and a new level of smartphone assisted personal trainer could be achieved.

Another interessant application area, would be in mountaineering (an area that was a great potencial and a personal interest).

With the excessive growth of adventure tourism enterprises, that lead unqualified persons to hazardous environments, the development of a system capable of performing action recognition to detect potentially dangerous actions (e.g. falling), integrated with GPS, temperature sensors, heartbeat sensors would be an interessant idea to explore.

Figure 2.22: A crowd of climbers slog up the Lhotse Face, heading toward Camp IV, last stop before the summit. Loose regulations and a boom in commercial guiding over the past two decades have made Everest far more accessible to experts and novices alike @Andy Bardon

3 Programming and integration in ROS of context-aware SaR mobile robots for urban firefighting

3.1 Introduction

Multi-Sensor information fusion is a process of information integration, merging data from disparate sources with differing conceptual and contextual representations[HL97]. This process allows to fulfil decision-making and estimation so as accomplish a certain mission based on the environmental characteristics. Nature has found a way to integrate information from multiple sources to a reliable and feature-rich recognition, such biological systems are able to compensate for the lack of information by reusing data obtained from the sensors. For instance, humans combine signals from the body senses (i.e, sight, sound, smell, taste, and touch) with knowledge of the environment to create and update a dynamic model of the surroundig world. The human way of fusing information can be compared to intelligent artificial sensor fusion systems. Both interact with the environment, perceive new information and interpret together with earlier experiences.
In the traditional method, information acquired by multiple sensors would have been processed separately, thus cutting off the connection between the acquired information and losing information characteristics[ZL09].
Since the computer technology can analyze and synthesize information from different nodes, the multi-sensor fusion as been widely used for real-time processing, i.e.[PNAL11, ZL09, BVB08]. The necessity to obtain a better understanding of some phenomenon in the development of intelligent system control functions to combine sensory data, thus generating methods that involves techniques such as statistical estimation, signal and image processing, artificial intelligence and information sciences.
Multi-Sensor information finds wide application in many areas of robotics such as object recognition, localization environment mapping, and monitoring environmental variables.
The fusion of multi-sensory information plays an important role in mobile robot perception over real-world. Robot perception requires system architecture support not found in simpler robot systems[SST].
In most cases integration of multi-sensory fusion in mobile robots is oriented to navigation, visual recognition and monitoring. This work is focus on monitoring the environment variables. The use of sensors with different characteristics allow to largely improve the performance of the overall system. The basic advantages of sensor fusion and integration are: redundancy, complementarity, timeliness, and cost [MMSV00].

3.1.1 Context and motivation

Mobile robots can be used on environments that humans cannot tolerate either due to contamination or very high risk of combustion
The CHOPIN project tries to level the cooperation between humans and robotic teams on catastrophic accidents or phenomenons, multi-sensor fusion can be used on mo-

bile robots teams to search and inspect so as to prevent catastrophes and monitor environments[1][CPR13]. Adopting a multi-sensor information method allows to retrieve a variety of information (e.g., temperature) to be fused and achieve an accurate judgment of contextual information (e.g., fire outbreak). This project as the objective to monitor environments and detect risk values with mobile robots teams with multi-sensor fusion integrated.
The mobile robots technologies in complex and unknown unstructured environments such as disaster scenes are still under study[ZLH08, DV96]. Currently, the main control mode of search and rescue (SaR) robotics is manual operation, also known as teleoperation, in which a complete autonomy is never achieved*[CM03, WJLSK12]*. It has been recently agreed in the literature that autonomous mobile robot require multi-sensor fusion to perceive the environmental information[WJLSK12], having a robotic intelligent sensor agent capable of storing data, processing data and act on the environment according to the context allows to improve the experience of information gathering. By doing this, first responders can percept the environment and focus on important parameters regarding the task while avoiding irrelevant data by having an efficient use of the group of sensors.

3.1.2 Objectives

The main contributions of this work are as follows:

1. Analyze the sensors response to several real laboratorial experiments in which Environmental Variables (e.g., gas, smoke, fire) different conditions with different are exposed to the sensors..

2. Study several classification methods and choose the most fitted one based on the current state-of-the-art. The training data will consist on readings from the previous point.

3. Sensors integration on mobile robotic team. Development of an algorithm to optimize the process of monitoring the environment with the previous selected classification method.

4. Construct a map of the environment variables in real time.

3.2 Multi-Sensor Information Fusion

3.2.1 Sensors

A sensor is a device that measures a physical quantity and converts it into a signal which can be read by an observer or by an instrument[Elm02]. Before the advent of microelectronics, the sensors or transducers used to measure physical quantities, such as temperature, pressure, and flow. They were usually coupled directly to a readout device, typically a meter, that was read by an observer. The transducer converted the physical quantity being measured to a displacement. However, microprocessor technology initiated the requirement for sensors to have an electrical output that could be more readily interfaced to provide unattended measurement and control. Therefore, nowadays, sensors help translating the real world of analog signals and varying voltages into the digital processing realm. Sensors typically convert non-electrical physical, biological or chemical quantities into electrical or optical signals. To be useful, the signal must be measured and transformed into digital format which can be processed and analyzed by processing units (e.g., computers). The information can be used by either a person or an intelligent device monitoring the activity to make decisions that maintain or change a course of action.
In power plants, automated vehicles, aircraft, and other complex systems, a large number of sensors are used for monitoring and control[AAM01]. Monitoring helps the operator in performing supervisory control tasks. A monitoring system receives information about the system through, sensors, and makes it available to the operator. By combining information from many different sources, it is possible to decrease the uncertainty and ambiguity inherent in processing the information from a single sensor source[AAM01]. A large number of sensors measuring many different variables can collectively achieve a high level of accuracy and reliability. Nevertheless, some steps are needed to make a multi-sensor system reliable. For instance, redundancy creation generates multiple values for the variable that is being estimated, thus improving the reliability of sensors. State prediction uses temporal information about the variable estimate for a specified time window to predict the value of the variable being measured at the next sampling point sensor. Data validation and fusion determine whether the information for the sensor can be trusted, thus associating a degree of belief in this measurement and combining the various redundant estimates to generate a "fused" value[Elm02].

3.2.2 Multi-Sensor Information Fusion

Since a single sensor generally can only perceive limited partial information about the environment, multiple similar and dissimilar sensors are required to provide sufficient local information with different focus and from different viewpoints in an integrated

manner. Information from heterogeneous sensors can be combined using data fusion algorithms to obtain observable data[ZLH08]. Multi-sensor system has the vantage to broaden machine perception and enhance awareness of the state of the world compared to what could be acquired with a single sensor system[AAMD11]. Multi-sensors are needed in a scenario in response to the increasingly learning nature of the environment to be sense. This motivates the emerging interest in research into contextual environmental information in catastrophic incidents (e.g., urban fires[1]). It is also beneficial to avoid overwhelming storage and computational requirements in a sensor and data rich environment by controlling the data gathering process such that only the truly necessary data is collected and stored.
The simplest task of a sensor management is to choose the optimal sensor parameter values, given one or more sensors, with respect to a given task. This is called active perception where in sensors need to be optimally configured for a specific purpose.
Multi-sensor management architectures is closely related to the form of data fusion unit. Typically, there are three alternatives for system structure, namely,
1.Centralized. In a centralized paradigm the data fusion unit is treated as a central mechanism. It collects information from all different platforms and sensors and decides jobs that must be accomplished by individual sensors. All commands sent from the fusion center to respective sensors must be accepted and followed with proper sensor actions.
2.Decentralized. In a decentralized system data are fused locally with a set of local robots rather than by a central unit. In this case, every sensor or plattform can be viewed as an intelligent asset having some degree of autonomy in decision-making. Sensor coordination is achieved based on communication in the network of robots, in which sensors share locally fused information and cooperate with each other.
3.Hierarchical. This can be regarded as a mixture of centralized and decentralized architectures. In a hierarchical system there are usually several levels of hierarchy in which the top level functions as the global fusion center and the lowest level consists of several local fusion centers[ZLH08] .

The basic purpose of sensor management is to adapt sensor behavior to dynamic environments. By having limited sensing resources, sensors may are not be able to serve all desired tasks and achieve all their associated objectives. Therefore, a reasonable process has to be made. More urgent or important tasks should be given higher priority in their competition for resources. The first step for the sensor management system should be to utilize evidences gathered to decide objects of interest and to prioritize which objects to look at in the time following. An interesting scenario requiring sensor coordination is shown in fig2.3. where 3 autonomous robots with multi-sensors cooperatively explore an area of interest.

Nevertheless, to achieve some sort of decision-making, each robot needs to be capable of assessing the contextual information. To that end, for this learning process,

patterns classification techniques are needed. Next chapter describes the most well-known used in the literature.

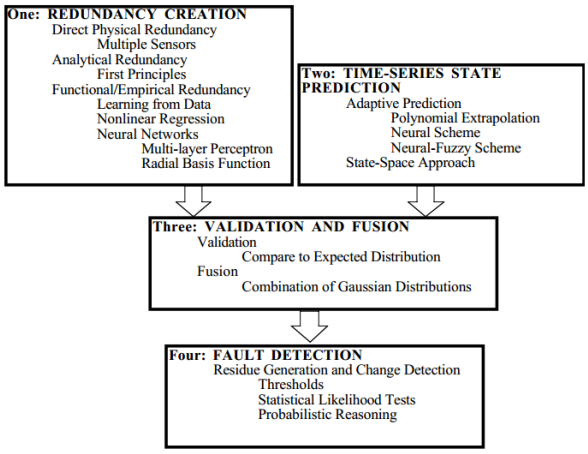


Figure 3.1: Flow chart representation of the four steps

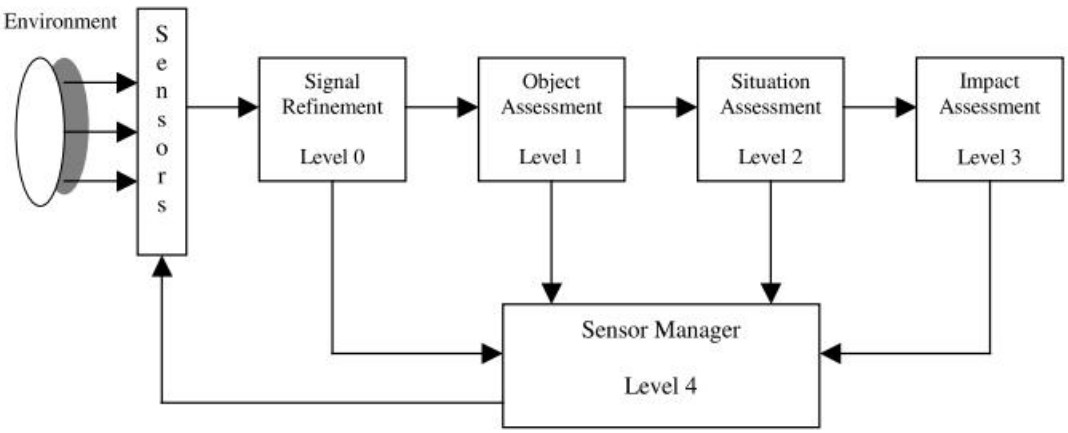


Figure 3.2: Feedback connection via sensor manager in a data fusion process

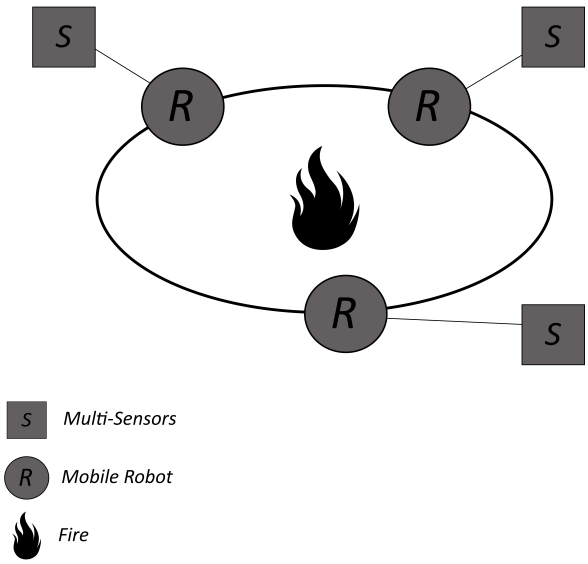


Figure 3.3: A team of mobile robots with multi-sensors cooperatively observing an area of fire in different points

3.3 Classification Methods

3.3.1 Neural Network

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information, a massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for use[Ste].
Artificial neural networks are composed of interconnecting artificial neurons (programming constructs that mimic the properties of biological neurons). Neural Networks provide the potential of an alternative information processing paradigm that involves large interconnected networks of processing units, units relatively simple and typically non-linear, units connected to each other by communication channels, connections carry data.
Artificial Neural Networks have a relationship with Statistics. Most Neural networks that can learn to generalize effectively from noisy data are similar or identical to statistical methods. Feed forward nets with no hidden layer (including functional-link neural nets and higher-order neural nets) are basically generalized linear models. Probabilistic neural nets are identical to kernel discriminant analysis. Kohonen nets for adaptive vector quantization are very similar to k-means cluster analysis. Hebbian learning is closely related to principal component analysis[Nic03].

3.3.1.1 Types of Artificial Neural Networks

1. *Supervised Learning*: where the network is supplied with a sequence of both input data and desired (target) output data network is told precisely by a "teacher" what should be emitted as output. The teacher can, during the learning phase, "tell" the network how well it should perform ("reinforcement learning") or what is the correct behavior ("fully supervised learning")[Gop98].

2. *Self-Organization or Unsupervised Learning*: a training scheme in which the network is given only input data. The network finds out about some of the properties of the data set and learns to reflect these properties in its output. This type of learning presents a biologically more plausible model of learning[Gop98].

3.3.1.2 Networks based on Feedback and Feedforward connections

Although neural network solutions for predictive analytics, pattern recognition and classification problems can be very different, they are always the result of computations that proceed from the network inputs to the network outputs. The network inputs are referred to as patterns, and outputs are referred to as *classes*.
Frequently, the flow of these computations is in one direction, from the network input

patterns to its outputs. Networks with forward-only flow are referred to as feedforward networks. Feedforward neural network is an artificial neural network where connections between the units do not form a directed cycle. This is different from recurrent neural networks. The feedforward neural network was the first and arguably simplest type of artificial neural network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network. Feedforward networks never contain feedback connections between units. Feedback (recurrent) networks always do. The presence of feedback connections in a network typically results in a network whose behavior is far more interesting and dynamic than a network composed of feedforward connections alone. Recurrent neural networks, allow data and information to flow in both directions.

3.3.1.3 Methodology: Training, Testing and Validation Datasets and Classification

In the artificial neural networks methodology, the sample data is often subdivided into training, validation, and test sets.

1. *Training set:* A set of examples used for learning, that is to fit the parameters [weights] of the classifier.

2. *Validation set:* A set of examples used to tune the parameters of a classifier, for example to choose the number of hidden units in a neural network.

3. *Test set:* A set of examples used only to assess the performance [generalization] of a fully-specified classifier.

4. *Classification:* Backpropagation algorithm,Fuzzy ARTMAP.

Multi-Layer Perception using Backpropagation is composed of layers of processing units that are interconnected through weighted connections. The first layer consists of the input vector The last layer consists of the output vector representing the output class. Intermediate layers called 'hidden' layers receive the entire input pattern that is modified by the passage through the weighted connections. The hidden layer provides the internal representation of neural pathways. Learning occurs in the perceptron by changing connection weights after each piece of data is processed, based on the amount of error in the output compared to the expected result.We want to adjust the network weights $w_{ij}(n)$ (n) in order to minimise the sum-squared error function:

$$E(w_{ij}^n) = \frac{1}{2}\sum_p \sum_j (targ_j^p - out_j^{(N)} - out_j^{(N)}(in_i^p))^2$$

and again we can do this by a series of gradient descent weight updates

$$\nabla w_{kl}^{(m)} = -\eta \frac{dE(w_{ij}^{n})}{dw_{kl}^{(m)}}$$

Note that it is only the outputs $out_j(N)$ of the final layer that appear in the error function. However, the final layer outputs will depend on all the earlier layers of weights, and the learning algorithm will adjust them all. The learning algorithm automatically adjusts the outputs $out_j(N)$ of the earlier (hidden) layers so that they form appropriate intermediate (hidden) representations.

Fuzzy ARTMAP is a supervised neural network architecture that is based on "Adaptive Resonance Theory", proposed by Stephen Grossberg in 1976. ART encompasses a wide variety of neural networks based explicitly on human information processing and neurophysiology. ART networks are defined algorithmically in terms of detailed differential equations intended as plausible models of biological neurons. In practice, ART networks are implemented using analytical solutions or approximations to these differential equations. Fuzzy ARTMAP is based on ART, internal control mechanisms create stable recognition categories of optimal size by maximizing code compression while minimizing predictive error during on-line learning. Fuzzy ARTMAP incorporates fuzzy logic in its ART modules,has fuzzy set-theoretic operations instead of binary set-theoretic operations. It learns to classify inputs by a fuzzy set of features (or a pattern of fuzzy membership values between 0 and 1)[S.S10].

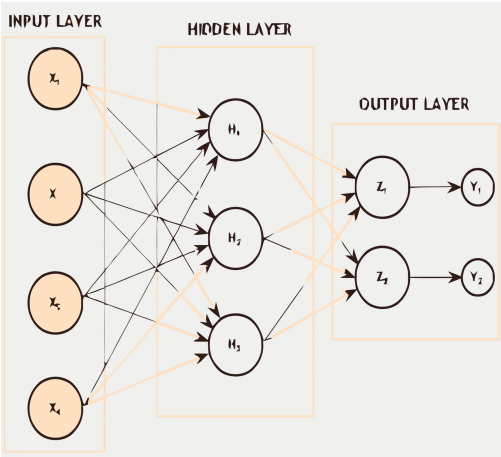


Figure 3.4: A Two-layer, Feed-Forward Network with Four Inputs and Two Outputs[2]

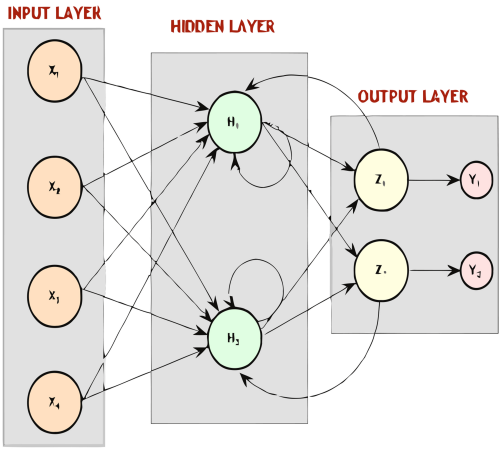


Figure 3.5: A Recurrent Neural Network with Four Inputs and Two Outputs[2]

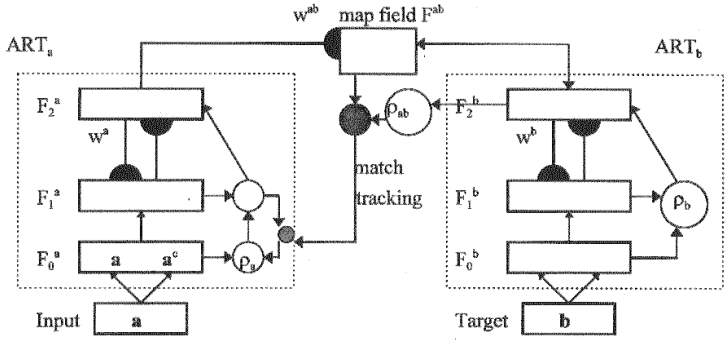


Figure 3.6: Fuzzy ARTMAP architecture [Carpenter et.al.,1992]

3.3.2 Fuzzy

3.3.2.1 Fuzzy Logic

Fuzzy logic is a form of many-valued logic or probabilistic logic; it deals with reasoning that is approximate rather than fixed and exact[L.A65]. In contrast with traditional logic they can have varying values, where binary sets have two-valued logic, true or false, fuzzy logic variables may have a truth value that ranges in degree between 0 and 1.
Fuzzy logic has found wide-spread popularity as a method for representing uncertainty particularly in applications such as supervisory control and high-level data fusion tasks,it provides an ideal tool for inexact reasoning, as example control, warning system and adaptive behaviour[CFLD12, CMRF12, CFM12].For the combination step in the fusion process, the advantages of fuzzy sets and possibilities rely in the variety of combination operators, which may deal with heterogeneous information (Dubois & Prade, 1985).An advantage of this approach is that it is able to combine heterogeneous information, which is usually the case in multi-source fusion (as in both examples given in the chapter), and to avoid to define a more or less arbitrary and questionable metric between pieces of information, since each piece of information is converted in membership functions or possibility distributions over the same decision space.
A main difference between fuzzy classification and possibilistic classification is that classes are generally considered as fuzzy sets in the first case and as crisp ones in the second case. In the following sections, we illustrate these two types of modelling.
Consider a universal set consisting of the elements x; $X = x$. Consider a proper subset$A \subseteq X$such that

$$A = \{x \,|\, x \;\; has\, some\, specific\, property\}$$

In conventional logic systems, we can define a membership function $\mu A(x)$ also called the characteristic function which reports if a specific element $x \in Xi$

$$A \rightleftharpoons \mu_A(x) = \Big\{ 1\; if\; x \in A \quad 0\; if\; x \notin A$$

In the fuzzy logic literature, this is known as a crispset.

$$A \rightarrow \mu_A \rightarrow [0, 1]$$

Composition rules for fuzzy sets follow the composition processes for normal crisp sets, for example

$$A \cap B \rightleftharpoons \mu_{A \cap B(x)} = min[\mu_A(x), \mu_B(x)]$$

$$A \cup B \rightleftharpoons \mu_{A \cup B(x)} = max[\mu_A(x), \mu_B(x)]$$

The normal properties associated with binary logic now hold; commutativity, associativity, idempotence, dis-tributivity, De Morgan's law and absorption. The only exception is that the law of the excluded middle is no longer true $A \bigcup A = X$, $A \bigcap A = \phi$. Together these definitions and laws provide a systematic means of reasoning about inexact values.
Fuzzy logic and probabilistic logic are mathematically similar – both have truth values ranging between 0 and 1 – but conceptually distinct, owing to different interpretations—see interpretations of probability theory. Fuzzy logic corresponds to "degrees of truth", while probabilistic logic corresponds to "probability, likelihood"; as these differ, fuzzy logic and probabilistic logic yield different models of the same real-world situations.
A basic application might characterize subranges of a continuous variable. For instance, a temperature measurement for fire detection. Each function maps the same temperature value to a truth value in the 0 to 1 range. These truth values can then be used to determine how the fire should be controlled.

In this image, the meanings of the expressions harmless, warning , and danger are represented by functions mapping a temperature scale. A point on that scale has three "truth values"—one for each of the three functions. The vertical line in the image represents a particular temperature that the three arrows (truth values) gauge. Since the red arrow points to zero, this temperature may be interpreted as "not hot". The orange arrow (pointing at 0.2) may describe it as "slightly warm" and the blue arrow (pointing at 0.8) "fairly cold"[BW07].

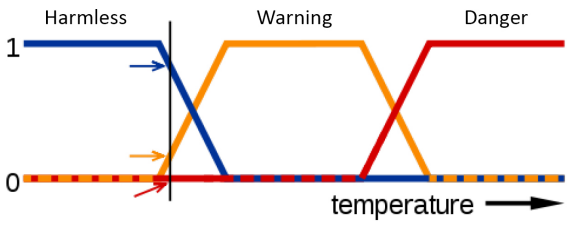


Figure 3.7: Fuzzy logic temperature

3.3.3 Support Vector Machines

3.3.3.1 SVM

In machine learning, support vector machines (SVMs, also support vector networks[1]) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis[Bur98, PNAL11, RNO+08, ZL09]. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on. In addition to performing linear classification, SVMs can efficiently perform non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.
SVM is a hybrid technique of statistical and deterministic approaches. This means that to find the best space for classification hypothesis, a probability distribution is determined from the input space. The technique originated from the work of Vapnik on the Principle of Risk Minimization, in the area of Statistical Learning[Tie08, Bur98]. The technique is applied in the following way: in the case of linear space, determine the hyperplanes of separation by an optimization problem; in the case of non-linear space, a kernel function is applied and the new space obtained is denominated the feature space. fig. 5.1 illustrates the application of a kernel in the input space. In the feature space, a hyperplanes is obtained for separation. In its simplest form, SVMs are linear

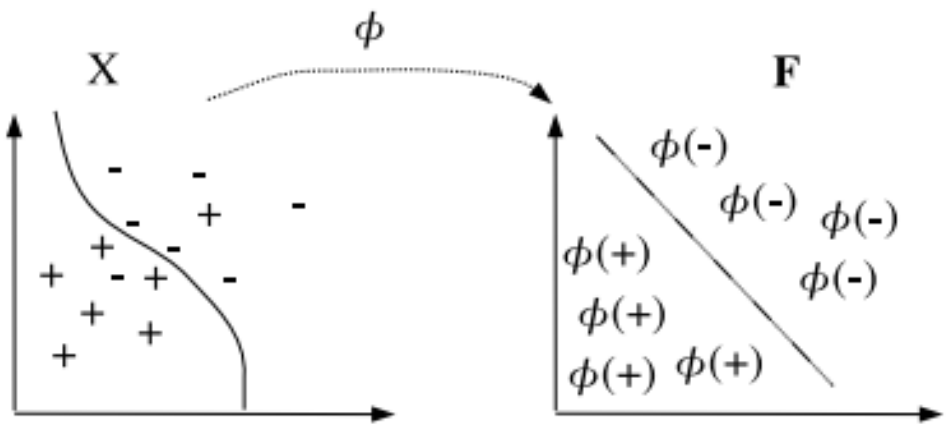


Figure 3.8: Mapping of an input space non-linearly separable for a feature space

binary classifiers that assign a given test sample a class from one of the two possible

labels. An instance of a data sample to be labeled in the case of remote sensing classification is normally the individual pixel derived from the multi-spectral or hyperspectral image. Elements of the feature vector may also include other discriminative variable measurements based on pixel spatial relationships such as texture. An important generalization aspect of SVMs is that frequently not all the available training examples are used in the description and specification of the separating hyperplane. The subset of points that lie on the margin (called support vectors) are the only ones that define the hyperplane of maximum margin The implementation of a linear SVM assumes that the

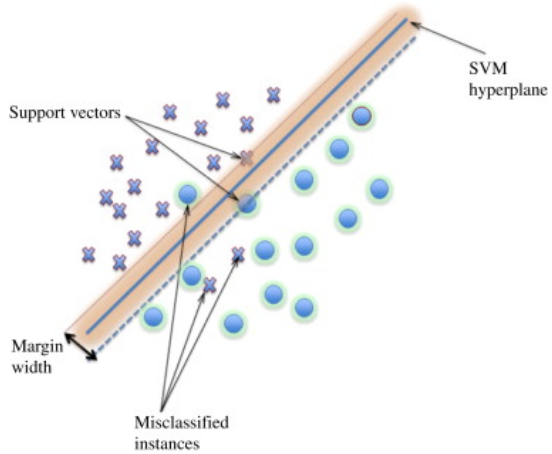


Figure 3.9: Linear SVM example

multi-spectral feature data are linearly separable in the input space. In practice, data points of different class memberships (clusters) overlap one another. This makes linear separability difficult as the basic linear decision boundaries are often not sufficient to classify patterns with high accuracy.VC dimension and capacity of functions: If you

$$\text{Test Error} \leq \text{Training Error} + \text{Complexity of set of Models}$$

take a high capacity set of functions (explain a lot) you get low training error. But you might "overfit". If you take a very simple set of models, you have low complexity, but won't get low training error.

3.3.3.2 SVM kernel

The solution of SVM is mapped the x-domain to a high-dimensional feature space with a nonlinear function ϕ, and then take on linear regression in high-dimensional feature space, to obtain the effect of original non-linear space regression, so its optimal function is expressed as:

$$f(x) = \omega \times \phi(x) + n$$

w is a vector and n a scalar. The dimensionality of $\phi(x)$ can be very large, making w hard to represent explicitly in memory,

$$\omega = \sum_{i=1}^{m} \alpha_i \phi(x_i)$$

So the decision function:

$$f(x) = \sum_i \alpha_i \phi(x_i).\phi(x) + b = \sum_i \alpha_i K(x_i, x) + b$$

and the dual dormation

$$min\, P(w,b) = \underbrace{\frac{1}{2}||\sum_{i=1}^{m} \alpha_i \phi(x_i)||^2}_{maximize\, margin} + \underbrace{C\sum_i H_1[y_i f(x_i)]}_{minimize\, training\, error}$$

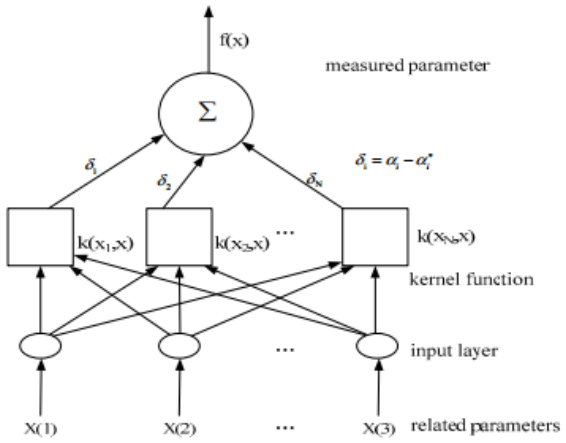


Figure 3.10: The model of multi-sensor information fusion based on SVM[ZL09]

3.3.4 Bayesian

3.3.4.1 Bayesian Probability

The Bayesian theory has the possibility to make predictions on future events an provides an embedded scheme for learning[JR08]. The Bayesian interpretation of probability can be seen as an extension of logic that enables reasoning with propositions whose truth or falsity is uncertain. To evaluate the probability of a hypothesis, the Bayesian probabilist specifies some prior probability, which is then updated in the light of new, relevant data.The use of hierarchical models and marginalization over the values of nuisance parameters. In most cases, the computation is intractable, but good approximations can be obtained using Hidden Markov Models (HMMs), Kalman filters an Particle filters. The sequential use of the Bayes' formula: when more data become available after calculating a posterior distribution, the posterior becomes the next prior. For the frequentist a hypothesis is a proposition (which must be either true or false), so that the frequentist probability of a hypothesis is either one or zero. In Bayesian statistics, a probability can be assigned to a hypothesis that can differ from 0 or 1 if the truth value is uncertain.

3.3.4.2 Sensor Models and Multisensor Bayesian Inference

Bayes' rule provides a means to make inferences about an object or environment of interest described by a statex, given an observation z.Bayes' rule requires that the relationship betweenxand zbe encoded as a joint probability or joint probability distribution P(x,z) for discrete and continuous variables respectively. The chain-rule of conditional probabilities can be used to expand a joint probability in two ways:

$$P(x, z) = P(x|z)P(z) = P(z|x)P(x)$$

$$P(x|z) = \frac{P(z|x)P(x)}{P(z)}$$

The value of this result lies in the interpretation of the probabilities P(x|z), P(z|x), and P(x). In this fusion process, the marginal probability P(z) simply serves to normalize the posterior and is not generally computed. The marginal P(z) plays an important role in model valida-tion or data association as it provides a measure of how well the observation is predicted by the prior. The value of Bayes' rule is that it provides a principled means of combining observed information with prior beliefs about the state of the world.

3.3.4.3 Sensor Models and Multisensor Bayesian Inference

$$P(z_1, ..., z_n|x) = P(z_1|x)...P(z_n|x) = \prod_{i=1}^{n} P(z_i|x)$$

The conditional probability $P(z|x)$ serves the role of a sensor modeland can be thought of in two ways. first, in building a sensormodel, the probability is constructed by fixing the value of $x = x$ and then askingwhat probability density $P(z|x = x)$ on z results. Conversely, when this sensor model is used and observations are made, $z = z$ is fixed and a likelihood function $P(z = z|x)$ on xis inferred. The likelihood function, while not strictly a probability density, models the relative likelihood that different values ofxgave rise to the observed value of z. The multisensor form of Bayes' rule requires condi-tional independence so that

$$P(x|Z^n) = CP(x)\prod_{i=1}^{n} P(z_i|x)$$

$$P(x|Z^k) = \frac{P(z_k|x)P(x|Z^{k-1})}{P(Z_k|Z^{k-1})}$$

whereCis a normalising constant. This states that the posterior probability onxgiven all ob-servations Z^n , is simply proportional to the product of prior probability and individual likelihoods from each information source.

3.3.4.4 Naive Bayes classifier

A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions[PR12]. A more descriptive term for the underlying probability model would be "independent feature model". The probability model for a classifier is a conditional model

$$P(C|F_1, ..., F_n)$$

Using Bayes theorem

$$p(C|F_1, ..., F_n) = \frac{p(C)p(F_1, ..., F_n|C)}{p(F_{1,....,}F_n)}$$

is equivalent to

$$posterior = \frac{prior \times likelihood}{evidence}$$

wich can be writen using the chain rule for repeated applications

$p(C, F_1, ..., F_n)$
$\alpha\, p(C)\, p(F_1, ..., F_n|C)$
$\alpha\, p(C)\, p(F_1|C)\, p(F_2, ..., F_n|C, F_1)$
$\alpha\, p(C)\, p(F_1|C)\, p(F_2|C, F_1)\, p(F_3, ..., F_n|C, F_1, F_2)$
$\alpha\, p(C)\, p(F_1|C)\, p(F_2|C, F_1)\, p(F_3|C, F_1, F_2) ...p(F_n, |C, F_1, F_2, F_3, ..., F_{n-1})$

so the distribution over the class variable C can be expressed like this

$$p(C|F_1, ..., F_n) = \frac{1}{Z} p(C) \prod_{i=1}^{n} p(F_i|C)$$

3.4 A Comparative Study of Classification Methods

3.4.1 Discussion

In [AKP+11]a comparative study of Support Vector Machine, Artificial Neural Network and Bayesian Classifier for mutagenicity prediction was made. Performance of the classifiers developed using ANN, SVM, Bayesian classifier, were compared to determine the best model for prediction of mutagenicity for present dataset. Sensitivity of SVM (69.14%) was found to be better than that of ANN (40.20%) and Bayesian classifier (58.44%). Precision of SVM model (74.9%) is comparatively higher than ANN (70.00%) and Bayesian (72.38%) models. Moreover, SVM predicts 15% and 5.5% less false negatives than ANN and Bayesian classification models respectively. ANN based model gave the highest specificity value (approx. 81%) as compared to the other two models. However, it lacks behind the other two models in terms of sensitivity, accuracy and precision values. Overall accuracy of SVM was found to be 71.73%, whereas accuracy of ANN and Bayesian was 59.72% and 66.14% respectively. These statistical studies indicate that SVM based model performance is comparatively better than the other two classifiers. In [LCP+] a comparative study between Linear Discriminant Analysis (LDA),Quadratic Discriminant Analysis (QDA), Naive Bayes with Normal (Gaussian) distribution (NV), Naive Bayes with Kernel Smoothing Density Estimate (NVK) and Least Squares Support Vector Machines with Radial Basis Function Kernel (LS-SVM), for golf putting performance analysis. The five classification methods were compared through the analysis of the confusion matrix and the area under the Receiver Operating Characteristic curve (AUC). From figure 7.2 it was possible to confirm that the LS-SVM has the most consistent results.

3.4.2 Conclusion

The most widely used data fusion methods employed in robotics originate in the fields of statistics, estimation and control. However, the application of these methods in robotics has a number of unique features and challenges. In particular, most often autonomy is the goal and so results must be presented and interpreted in a form from which autonomous decisions can be made; for recognition or navigation, for example. Classification is a process of supervised learning where data is separated into different classes on the basis of one or more characteristics inherent in data. In this study it is possible to see there are efficient alternatives to heavy probabilistic methods. SVMs are recent technique suitable for binary classification tasks, which is related to and contains elements of non-parametric applied statistics, neural networks and machine learning. Like classical techniques, SVMs also classify a company as solvent or insolvent according to its score value, which is a function of selected financial ratios. As we can see in[AKP+11] and [LCP+] the results of these models were acceptable, but the SVM

based model was found to be more efficient. Since SVM use kernel , it contains a non-linear transformation, no assumptions about the functional form of the transformation, which makes data linearly separable, is necessary. The transformation occurs implicitly on a robust theoretical basis and human expertise judgment beforehand is not needed. SVMs provide a good out-of-sample generalization, if the parameters C and r (in the case of a Gaussian kernel) are appropriately chosen. This means that, by choosing an appropriate generalization grade, SVMs can be robust, even when the training sample has some bias.

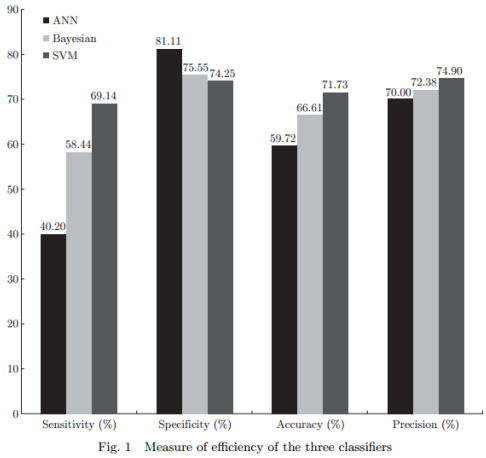


Figure 3.11: Measure of eficiency of the three classifiers [AKP+11]

Class	LDA	QDA	NV	NVK	SVM
1	0.619	0.601	0.671	0.680	0.744
2	0.650	0.623	0.692	0.685	0.737
3	0.566	0.582	0.634	0.761	0.734
4	0.507	0.585	0.574	0.675	0.690
5	0.622	0.651	0.692	0.766	0.797
6	0.493	0.602	0.650	0.718	0.745

Figure 3.12: Avarage value of the AUC [LCP+]

4 Requirements analysis and design of a command center for USAR robotics

4.1 Introduction

In an Urban Search and Rescue (USAR) mission, firefighters' actions are planned and controlled in the Command Center (CC), so it's imperative to provide this central node with all information available to support the commander decisions presenting a global vision of the state of the mission on the ground and individual state of the agents and resources available on the Theatre of Operations (TO).

Decision making is largely impacted by the external environment and the decision maker is affected by several factors, e.g. *time stress* which results in fatigue and loss of vigilance, forcing the decision maker to use less complicated reasoning strategies in order to save time.

Despite the evolution in information and communication technologies and the developments in ubiquitous computing, firefighters on ground level still make use of archaic methods to register incident occurrences depending on the expertise and resilience of the decision makers.

Some steps have been taken and Portugal adopted the *Portuguese Integrated Security and Emergency Network System ("Sistema Integrado das Redes de Emergência e Segurança de Portugal")* (SIRESP) to integrate communications of all civil defense agencies, while the *District Command ("Comando Distrital de Operações de Socorro")* (CDOS) of *National Authority for Civil Protection ("Autoridade Nacional de Protecção Civil")* (ANPC) is equipped with the *Operational Decision Support System ("Sistema de Apoio à Decisão Operacional")* (SADO) software, which does not give direct support to firefighters and commanders on the TO.

Firefighter Fatality Investigation & Prevention Program of NIOSH[1] identifies *inadequate risk assessment*, *lack of accountability or SCBA air management (BA Control)*, *inadequate communication* and *lack of situational awareness* as some of the most common factors associated with Line Of Duty Death (LODD).

Further study by the IAFF[2] (USA) quantifies *lack of situational awareness* as responsible for 37.3% firefighters' injuries [Gri09], therefore, the use of technology to enhance situational awareness and to automate and control time-depending tasks, by alerting the decision-makers can save firefighters lives.

The use of robot teams in cooperation with human teams for search and rescue operations allows decisions to be made closer to the place to which they relate, based on information provided by various sensors of the robots and the human team experience.

[1] National Institute for Occupational Safety & Health
[2] International Association of Fire Fighters

Robotic teams can assist human teams browsing in dangerous areas and with low visibility, looking for survivors and providing important information to define the context in which the teams are operating, allowing more rapid, efficient and safer operations of the human teams in USAR operations, and allowing the CCO to anticipate hazardous events for agents on the ground. However, the whole operation will always be planned, monitored and controlled from the CC, so this work aims to develop a computer interface capable of providing an overview of events in the theatre of operations, monitoring the status, location and actions of all agents operating in the terrain, as well as the logistical resources mobilized within the mission, providing this central node with all information available to support the commander decisions

This interface aims to automate some tasks that are nowadays carried out manually by the commander, including recording and monitoring information coming from every agent in the TO, and generating operational warnings when critical situations are detected. In this way, the person in charge of the mission can avoid being distracted with collecting and organizing such information, being focused in analyzing it and making decisions.

The cooperative exchange of information in order to decentralize some decisions is achieved by using the latest technologies, such as fixed large touch screens, hand held devices, internet access, and robots with several sensors, communicating over a Mobile ad hoc Network (MANET) providing best-effort communication to the search and rescue teams.

Due to communication constrains in unpredictable and dynamic environments, tagging mission context information for later retrieval represents a relevant feature to define context in time and space, therefore, the main events during USAR operations are registered in a database for prior context definition during ongoing actions and subsequent analysis of the course of the mission.

To understand the concept of operations and model first responders operations on the ground, several meetings were promoted with the domain experts, firefighters corporations and members of the National Authority for Civil Protection ("Autoridade Nacional de Protecção Civil") (ANPC) structure: *Bombeiros Sapadores de Coimbra*, *Bombeiros Voluntários da Mealhada* and *Bombeiros Sapadores de Faro*; with the district command of ANPC: *Paulo Palrilha, 2.º Comandante Distrital de Operações de Socorro (CODIS) de Coimbra* and *José Bismark, CODIS de Aveiro*; and documentation review of the manuals of the Firefighter National School (Firefighters National School ("Escola Nacional de Bombeiros") (ENB)), ANPC standard operating procedures (NOPs) and portuguese legislation.

The gathered material allows to identify the mission goals, the tasks performed by first responders and acting patterns in unpredictable environments, leading to the elicitation of user requirements and the design of the user interfaces, supported by interaction design principles.

The integration of technologies introduced by the Cooperation between Human

and rObotic teams in catastroPhic Incidents (CHOPIN) project with CCO were tested in controlled laboratory environment, namely local maps transmission, robots' pose, environmental variables readings, victims and fire-outbreaks location and firefighters status.

4.2 Requirements analysis

4.2.1 Application Context

Operational model and procedures of civil defense agencies and fire departments of different countries are similar. Usually, the incident response mission is triggered by an emergency call to the control centre which mobilizes first responders to the incident zone. The first responders' commander starts to gather as much information as possible about the incident (reconnaissance phase) and defines an appropriate response tactics. On ground level, the command post is settled and the incident commander must get an understanding of the amount of additional resources and the level of command required [NOP09] [GHC09].

4.2.2 Operation Mode of Portuguese Firefighters

The structure of the post of command is evolutive, and changes as the incident takes other proportions and evolves to a different phase, thus changing the structure of command and the layout of resources on the ground. The phasing of the incident depends on the complexity and on the number of resources involved. In phase *I*, for non complex operations, up to six teams are involved, independently of their typology and the firefighter with the highest rank on the ground takes the role of Incident Commander ("Comandante das Operações de Socorro") (COS). In phase *II*, with resources equivalent to three groups, with six teams each, the Operations Section ("CElula de Combate/OPerações") (CECOP)[3] is activated. For phases *III* and *IV*, the Operations Management System ("Sistema de Gestão de Operações ") (SGO)[4] is divided into three cells: Planning Section ("CElula de PLANeamento") (CEPLAN)[5], Logistics Section ("CElula de LOGística") (CELOG)[6] and CECOP. This structure is used in complex and large catastrophic scenarios and, in this last phase, the operation is conducted by a member of the ANPC[7] structure. The CC uses nowadays a vehicle with magnetic white boards to manually record all incidents on the ground [Figure 4.1], coordinate the operations of the USAR teams and manage the resources on the TO.

[3]CECOP - Célula de Combate/Operações
[4]SGO - Sistema de Gestão de Operações
[5]Célula de Planeamento
[6]CELOG - Célula de Logística
[7]ANPC - Autoridade Nacional de Protecção Civil

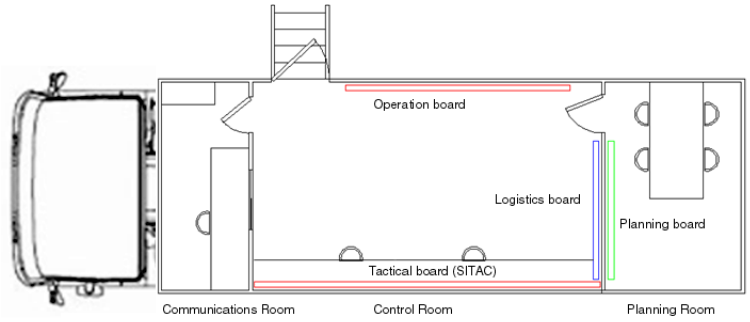


Figure 4.1: Layout of the CC vehicle.

As the level of command changes, the new COS[8] should identify clearly and unambiguously all set of resources, structures, type of ongoing and planned actions on the TO, therefore ANPC adopted a common operational symbology for the management of USAR operations. These tools allow representing graphically and dynamically all main events and key information concerning the ongoing occurrence, and understanding the transmission of operational information at different levels of command, control and execution (strategic, tactical and manoeuvre levels)[NOP09]. All the planned and ongoing actions, as well as teams and resources (planned and available), are represented in a white board designated by SITAC[9], which is updated in the course of operations, providing mission global awareness.

Other predefined forms are used to record the structure of command, damages, victims, communications plan, mobilized resources and weather reports distributed through the respective boards in the CC, as shown in Figure 4.2.

All communications are centralized in the CC based on analog radio communications and on the Portuguese Integrated Security and Emergency Network System ("Sistema Integrado das Redes de Emergência e Segurança de Portugal") (SIRESP)[10], which is based on the European European Terrestrial Trunked Radio (TETRA) communication system. Therefore, the update of the boards relies on information that is requested to agents on the ground and on their reply.

The information that reaches the commander normally exists in large quantity and quality, but may not arrive in time, due to limitations of communications equipment, existence of shadow zones, and the accumulated stress of the team commanders. This time delay, as well as the amount of information the CCO have to process, may prevent the right decisions in the right time, thus putting at risk the mission or delaying predefined tactical actions to be taken.

[8]COS - Comandante das Operações de Socorro
[9]SITAC - Situação Tática
[10]SIRESP - Sistema Integrado de Redes de Emergência e Segurança de Portugal

(a) SITAC on magnetic board in CCO. (b) Forms on magnetic boards in CCO.

Figure 4.2: SITAC main board and forms to register incidents on the ground in the PCO.

4.2.3 Decision making

Command and Control (C2) in firefighters operations is based on a hierarchical military model with pre-defined acting rules. Nonetheless, if the decision maker is an expert, he or she interacts with the situation at a *skill-level* of cognitive control, where the decisions are generated *subconsciously*. At a *rule-based* level of cognitive control, the decision maker is familiar with the situation, forming his decision on prior experiences or learned from another person's know-how, where the control of behaviour at this level is goal-oriented, knowing that other alternatives of action are available. On a novel or unfamiliar situation, the decision maker acts on a *knowledge-based* (also called *model-based*) level of cognitive control, involving the transformation of declarative knowledge into procedural knowledge [Ras93].

The command and control model in a USAR mission is similar to the one used in military actions, Boyd's OODA loop (*observe* - *orient* - *decide* - *act*). Brehmer in [Bre05] propose a work flow model to describe how members of the command teams perform their tasks, merging the OODA loop and cybernetic approaches to Command and Control [Figure 4.3].

The data is collected from visual contact, information received from firefighters and survivors and from the system. The mission is planned based on gathered information, pre-defined USAR actions and the commander experience, transmitting the orders to accomplish the mission and waits feedback effects to reformulate the plan of action until the mission is completed.

In dynamic and unpredictable environments such as USAR missions, all three levels of control can interact with the situation depending on the decision maker an its training level and expertise.

Figure 4.3: The dynamic OODA loop (DOODA loop). functions are given in black, products in red, and inputs in green

4.2.4 Situation Awareness

Generally, Situation Awareness (SA) is a human's understanding of what is happening around the human operator at the current point in time and in the near future. Endsley et al. in [EBJ03] divides the formal definition of SA into three separate levels:

- Level 1 - *perception* of the elements in the environment;
- Level 2 - *comprehension* of the current situation;
- Level 3 - *projection* of the future status.

USAR missions occur in highly dynamic and unpredictable environments where the situation is always changing forcing operator's SA to change constantly or become inaccurate. Human operators have limited attention capacity to attend to multiple items simultaneously, and Miller in [Mil56] states that only seven (plus or minus 2) chunks of unrelated information can be held and manipulated in humans' working memory, which consist in two major constraints for SA. In these dynamic environments, where information changes constantly and is hard to obtain, other factors can deteriorate significantly SA, like *attentional-tunnelling*, *data overload*, *workload*, *anxiety*, *fatigue*, *misplaced salience*, *errant mental models* and *out-of-the-loop syndrome*.

The system design must take these factors into account to avoid them and create and maintain high levels of SA and the fact that decision makers form mental models of how the system works to form higher levels of SA (comprehension and projection) providing default information even when data is incomplete or missing.

For a stronger situational awareness, the messages passed among agents and the CCO should be clear to influence the effectiveness of information.

4.2.5 Risk Assessment

One of the main problems that firefighters face in USAR missions, is the rapid fire phenomenon, which have three basic types, with accepted scientific terminology: *Flashover*, *Backdraught* and *Fire Gas Ignition*, which conducts to high rising temperatures in a small period of time, injuring and killing nearby firefighters. Then, becomes of utmost importance the recognition of the warning signs of these impending events [SOG06], sharing the information with the surrounding agents so they can try to leave the hazard area. Some of these warnings comprise changes in smoke conditions, like color alterations (particulary darkening), high velocity through an exit, sudden lowering layer and pulsing (raising & lowering cycle in the layer); heat alterations forcing agents to crouch low; detection of hazard conditions with inflammable gases. These changes in the environment can be detected by fusing several sensors readings of different systems, like robots and firefighters wearable devices, augmenting situation awareness.

The three main tactics used in preventing or counter rapid fire development are *tactical ventilation*, *fire confinement* and *3D water-fog tactics*, whilst the use of these tactics are decided in the CCO, incident commander should make his/her decision based on the better knowledge of the context where the teams are working in.

4.2.6 User Classes

The use cases in appendix A demonstrates the distinct roles of robots, firefighters and commanders within the system. These diagrams also mention the various boards which are updated during the operations. The strategic and tactical commanders are outside the hot zone with access to these boards controlling the progress of ground operations. According to their function and location, systems' user classes are identified.

- The **incident commander** and the **supporting officers** are located in the same physical space with access to fixed large touch screens, sharing the same information.
- The **sector commanders** are located within the limits of the accident area, monitoring the course of actions within its sector, using tablets connected to the network to interact with the system and the agents on the ground.
- The **breath apparatus entrance control officer** coordinates the Self-Contained Breathing Apparatus (SCBA) activities, when adverse conditions in the building require a tight entrance control to limit the time of exposure to the incident. This can be achieved by registering SCBA activities and the firefighters information in the respective board with a tablet connected to network. SCBAs with level control are too expensive and the activities must be time controlled, echoing an alarm, when the defined limit is reached, in the firefighters board and in the mobile device attached to the firefighter.

- **Firefighters** operates inside the sinister area, to search and rescue potential victims, determine hazardous conditions and inform the COS about their actions and course of the mission. The firefighters use a mobile device with sensors, e.g. smart phone, connected to the system by a MANET to update the firefighter status and send/receive some small messages and signals (e.g. if the firefighter stay still for more than thirty seconds an alarm will echo to warn the command center and the other firefighters nearby.

- **Robots** provide crucial information, building temperature, gas and 2D maps and identifying potential victims and hazardous conditions for the agents on the ground. This information is registered in the robot's board, the SITAC board, victims board and the operations board. Robots are connected to the Base Station (BS) through the MANET and communicate with the CCO using a predefined set of messages.

- ANPC control center receives request for new resources and status reports updates through the support commander officers and is external to the system.

4.3 Interface requirements

4.3.1 Hardware Interfaces

The interaction interface is supported by the hardware interface where several views of the Human machine interface (HMI) are distributed over fixed touch screens for the tactical operators, mobile devices (e.g iPADs) for the Sector Commander (SC), smart phones (e.g. iPhones) for the firefighters and multiple sensors (lasers, sonars, dust, alcohol and pyroelectric sensors) for the robots on the TO. All these devices can be used to share information and implement collaborative context awareness.

The use of computers, tablets and smart phones increases the ability to transmit other data types, complementing the standard audio channel used in search and rescue scenarios. Robots provide important information for strategic decision and enable a better context definition. The visual information for representing tasks status, planning and ongoing actions are transmitted more accurately than verbal communication [Ash07].

Adding "Touch" technologies functionality provides a more intuitive operation mode. Moreover, the development of online mapping websites has become popular, facilitating access to geospatial maps that were only available through a Geographic Information System (GIS), allowing a detailed view of the surrounding TO. Databases with Global Positioning System (GPS) coordinates, combined with mapping web sites, can be a helpful tool to indicate several crucial points (e.g. Hydrants) and the location of available resources within the TO.

Devices indicated here only serve as a proof of concept, and should be replaced by more rugged and appropriate devices in real USAR missions.

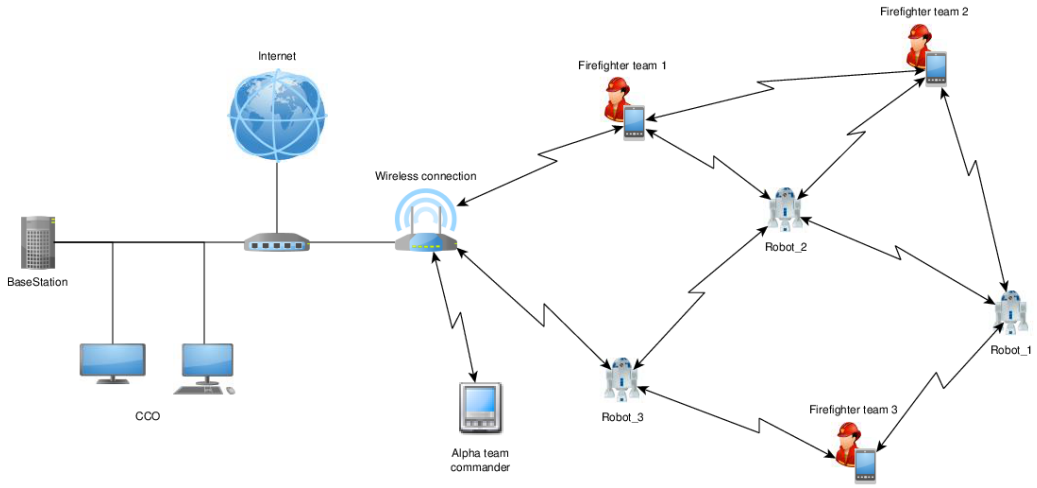


Figure 4.4: System architecture

4.3.2 User interface requirements

The interface design development is focused on finding a suitable representation that engages the end-user inputs to achieve the desired system output, making the application usable and useful.

To meet these criteria the CCO interface development addresses to some interaction design laws that should be observed for a good *Human machine interface* (HMI). Therefore, *clickable* objects, like operating buttons, are designed to have reasonable sizes and placed preferentially on the edges and corners of the screen to make it easier and faster to operate, since they are huge targets, with theoretically infinite height or width, because the mouse stops at the edge of the screen even with a fast movement, by referring to Fitts' law which states *"the time it takes to move from a starting position to a final target is determined by two factors: the distance to the target and the size of the target"*.

Gestures recognition is added to the application combined with large touch screens to meet the Fitts' law statement, because it is more natural and faster for a user to point to an object with his finger rather than with the *mouse* pointer. The selectable objects has an associated information window or pop-up menu next to them, with no more then ten options to allow a direct manipulation of the object and a faster decision of which action to take. Hick-Hyman law says that the time it takes the user to make

a decision is determined by the number of options available, which is consistent with George Millers *Magical Number Seven* rule, where he states that the human mind is best able to remember information in chunks of seven pieces, plus or minus two, having some difficulties to keep more than that information amount in short-term memory [Mil56].

The design of an application to operate in a dynamic and unpredictable environment, capable of enhancing SA and maintaining knowledge common ground for decision makers and operating agents presents large complexity, so according to Teslers law of *Conservation of Complexity* it is important to transfer some complexity to the system. Teslers law states that there is a point beyond which its not possible to simplify a process any further, and some complexity can only be transferred from one place to another. This is achieved automating some processes, drawing alarm signs and eliminating redundant tasks to focus the decision makers attention in really compelling operations.

The system interaction design must be driven by the *Poka-Yoke* principle, avoiding (*yokeru*) inadvertent errors (*poka*), to adjust some behaviors of the end users and prevent eventual errors. This principle was created in 1961 by the japanese industrial engineer and quality guru Shigeo Shingo. To accomplish this principle the system must inform the end user after a performed action (*feedback*) to trace operation success and before an action is taken (*feedforward*) so the operator know what is about to do, limiting the options of the end-user.

The system interaction design should be directed by users' concerns to make it more attractive and easier to use (*user-centered*) but also focused on the tasks the user must accomplish (*activity-centered*).

Changing methodologies always find some resistance, therefore the use of the ANPC forms and the analysis of its usual procedures may contribute to a greater acceptance of the proposed model for search and rescue operations. Using the same symbology defined by ANPC will improve the acceptance of the system because users will feel comfortable with the vocabulary used. The graphical environment should be easy to operate, using informative icons with large dimensions representing actions, since they are better remembered than *text commands*, allowing *drag&drop* operations, resizing and rotating objects.

The actions on the tactical situation board will have a side effect on the forms of the ANPC to create a level information automation eliminating redundant tasks and enhancing situation awareness with the use of *dynalinks* where information depicted in one window explicitly changes in relation to what happens in another [SR96].

Endsley et al. in [EBJ03] describe 50 principles of designing for SA, where four are directly intended to the display of information to support shared SA:

- Principle 45: Build a common picture to support team operations
- Principle 46: Avoid display overload in shared displays

- Principle 47: Provide flexibility to support SA across functions
- Principle 48: Support transmission of different comprehensions and projections across teams.

The defined HMI must have these principles into account to support shared understanding, to interact with all agents on the ground and characterize awareness. Situation awareness in this HMI have five components: human-robot, robot-human, human-human, robot-robot and humans overall mission awareness [DHYS04].

Therefore, the HMI should:

- Provide maps of where the firefighters and robots have been, providing spatial information of the immediate surroundings, thus enhancing awareness;
- Avoid the "mental" (or manual) data fusing by human agents by providing automated multi-sensor information fusion;
- Display multiple robots on a single window, increasing efficiency and providing the CC operator with the most appropriate level of automation at any given time [DHYS04].

However, the use of a higher level of automation may lead to a poorer acceptance by end users. Therefore, it is important to seek a balance between human/manual and automated operation; a mixture of human and automated control is desirable [MP07].

4.3.3 Context-awareness requirements

The overall system design is context-aware, exploiting decentralized collaboration between human and robotic teams, sharing the surrounding context with the nearby agents and the CCO.

Abowd et al. in [ADB+99] define *context* as any information that can be used to characterize the situation of an entity, where an entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves. The most relevant context types to characterize the situation of a particular entity, according to this author, are **location**, **identity**, **time** and **activity** defining the primary level of context, responding to the questions, *where*, *who*, *when* and *what*. So every feature in the system presents this primary information content to characterize the context.

The definition of Context-aware is not a consensual issue, requiring application's behavior be modified to be considered context-aware, Abowd et al. in [ADB+99] states that "a system is *context-aware* if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task". This context-aware definition is more general, therefore applications that only displays context information

about the mission's environment without modifying its behavior, are also classified as context-aware application.

Context-aware applications support features depending on the context, so Abowd et al. in [ADB+99] propose a *context-awareness* taxonomy, fusing the taxonomy proposals of Schilit et al. [SAW94b] and Pascoe [Pas98b], into three categories:

1. **presentation of information and services to a user** :
   - agents and the CCO have the knowledge of discovered features locations and status, thus enhancing SA;
   - robot agents fuse data sensors readings to activate CCO and nearby agents application widgets when relevant features in the environment changes, alerting end-users in a convenient form for context alteration, reducing data overload;
   - also visible options to the end-user reflects the possible actions to be taken in accordance with the surrounding context;
2. **automatic execution of a service** :
   - using an appropriate level of automation and decentralizing some decisions from tactical actors, the system responds to particular context changes activating services, which without any reasonable doubt, the human actor should take (e.g. firefighter with a SCBA should receive a retreat order after half time work of the used device capacity;
   - after an evacuation command, if a human agent is moving towards a blocked exit or a hazard area, agents on the ground with different SA should warn and provide an alternative exit for the firefighter);
3. **tagging of context to information for later retrieval** :
   - information context sharing provides the users with contextual augmentation with additional digital information. Environmental alterations and new discovered features are registered in the database to encompass contextual events of the surroundings and forwarded to the agents enhancing SA.

4.3.4 Human-Robot Interaction requirements

USAR missions are classified as *synchronous* (at the same time) and *non-collocated* (in different spaces) in the canonical Computer-Supported Cooperative Work (CSCW) Ellis et al. "Time-space taxonomy" [EGR91], but taxonomy does not cover the dimension of collaborators communication mode, which is an important feature in USAR missions, which can also be classified as *collocated* (in the same space) in our approach

to collaborative work between human and robotic teams. Yanco and Drury in [YD04] present a taxonomy to classify HRI and overcome the "Time-space" and other proposed taxonomies constrains. In the scope of the CHOPIN project, this taxonomy is adapted to support the situation awareness's five components of the HMI [Figure 4.5]: *human-robot*, *robot-human*, *human-human*, *robot-robot* and *humans overall mission awareness* [DHYS04].

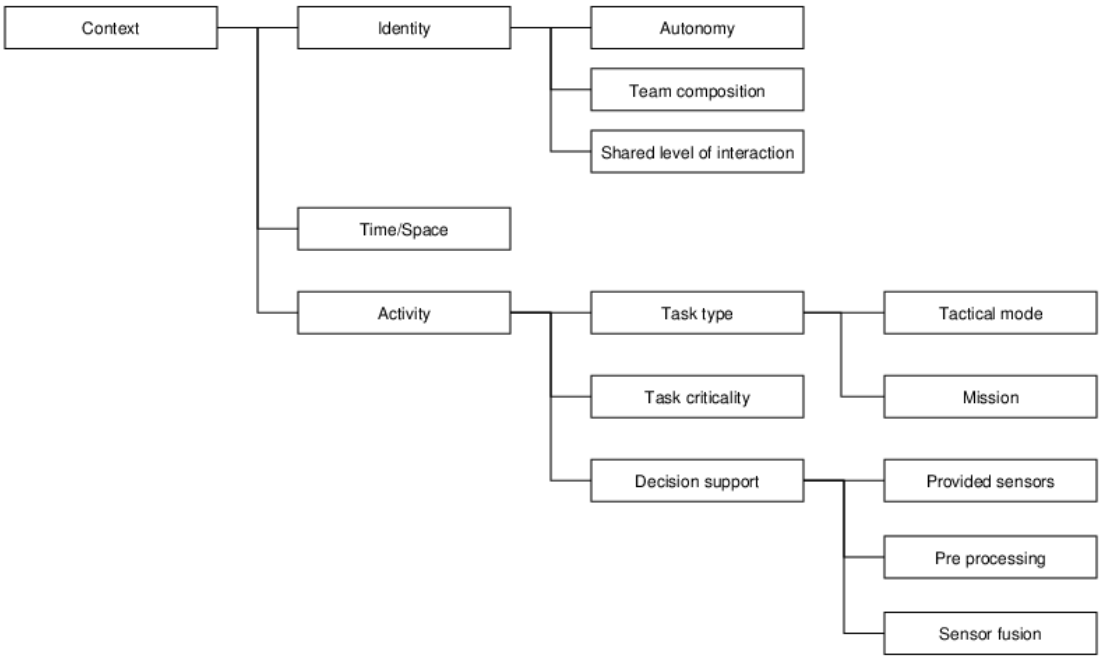


Figure 4.5: Taxonomy for multi-agent HRI.

The relevant categories for the USAR system are :

1. **Task Type** [TT] : robots and firefighters on the ground perform different tasks as victim search, detection of fire outbreaks and hazard conditions and reading environmental variables, reflecting the assigned objective of the agent. These tasks have different procedures or are incompatible with the declared tactical mode

2. **Task Criticality** [TC] : the first aim of USAR missions is to protect human life, rescuing incident victims and assuring firefighters physical integrity, therefore some tasks have higher priority than others, by giving primacy to the protection of human life on the protection of property. Task criticality of an agent's mission can only be changed by the CCO.

3. **Composition of Teams** [CRT] : agents on the ground, should know whom their teammates are. Although they can collaborate with other team members it is important, for context characterization, that robot agents can recognize their teammates, as human agents do.

4. **Level of Shared Interaction Among Teams** [LSI] : in the scope of the CHOPIN project, CCO gives orders to a team of robots [Figure 4.6.B] that need robot-robot HRI awareness to coordinate themselves, although, in order to decentralize some decisions, the remain combinations are possible to occur in USAR scenarios [Figure 4.6].

5. **Decision Support for Operators** [DSO] : The agents on the ground should have:

   - a list of provided sensors - includes a list of sensing type to the operator for decision making.
   - a list of the type of sensor fusion, e.g. $\{\{sonar, ladar\} \rightarrow map\}$
   - the amount of pre-processing of sensors for decision support, e.g. $\{ladar \rightarrow map\}$

6. **Time/Space** [TS] : One of the major features to define *context* is the interveners location and time, therefore both robots and firefighters should communicate these aspects within the incident area, during the assigned mission.

7. **Autonomy Level** [AL] : In this work, the level of autonomy is considered to be the capacity (battery level for robots and SCBA level for firefighters) to ensure a safe exit for the agent, considering the duration of working time, the distance to the nearest safe exit, the consumption rate and the remaining time to travel the escape path.

4.3.5 Functional Requirements

The system will interact with all the actors on the ground through transactions on the screen by the end user meeting the functional requirements listed in tables following tables:

4.3.5.1 Search and Rescue requirements

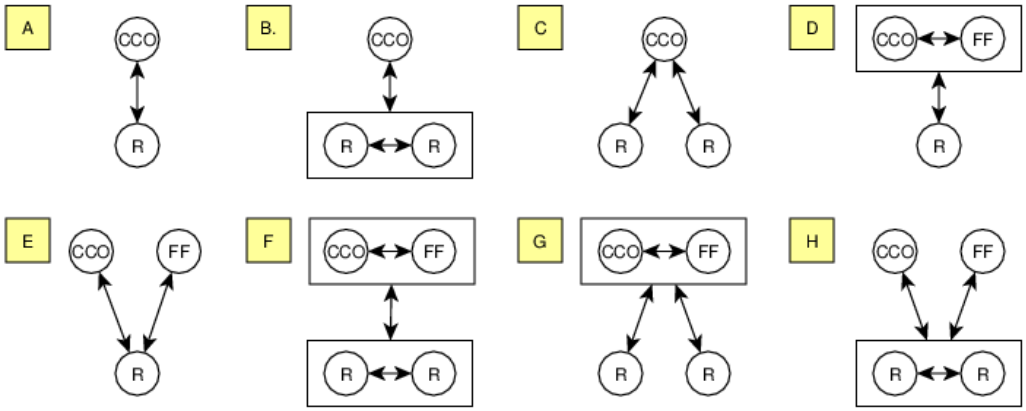


Figure 4.6: The possible combinations of CCO, firefighters and robots, acting as individuals or in teams. (adapted from [YD04])

Table 4.1: Search and Rescue Functional Requirements.

N.º	Requirement
SRFR 01	The system will record automatically founded victims by the robots, as well as their location and status, with an associated warning to the end-user with a visual and acoustic alarm.
SRFR 02	The end-user is be able to mark founded victims communicated by the firefighters, as well as their location and status publishing information with agents on the ground.
SRFR 03	The end-user must be able to activate camera view to identify a potential victim and evaluate their state.
SRFR 04	The end-user must be able to define a new goal for a robot or a team of robots publishing a new area to scan.

4.3.5.2 Command and Control requirements

Table 4.2: Command and Control Functional Requirements.

N.º	Requirement
C2FR 01	Change tactical mode of operations and communicate it to all the involved agents (Offensive, Defensive and Transitional).
C2FR 02	Classify the incident and fill an appropriate form to help him to make a risk assessment.
C2FR 03	Write a situational report and should be able to send it.
C2FR 04	Define initial coordinates of deployed robots on the ground and initiate their mission
C2FR 05	Record the developments of tactical situation: all tactical actions will be recorded in the command board to get an overview of the incident evolution
C2FR 06	Write/draw task information and communicate it to the agents.

4.3.5.3 Situation Awareness requirements

Table 4.3: Situation Awareness Functional Requirements.

N.º	Requirement
SAFR 01	Track robots on the ground
SAFR 02	Track firefighters on the ground
SAFR 03	Know the firefighter status
SAFR 04	Know the robot status (battery level, mission,...)
SAFR 05	Display local plant of activity
SAFR 06	Display temperature map
SAFR 07	Display gas map
SAFR 08	Know the location of hazardous materials
SAFR 09	Know the location of fire outbreaks, their state of evolution and the class of fire
SAFR 10	Record areas where searches were carried out
SAFR 11	Display images supplied by the robot
SAFR 12	Record existing damage on site
SAFR 13	Display incident time line

4.3.5.4 Self-Contained Breathing Apparatus requirements

Table 4.4: Self-Contained Breathing Apparatus Functional Requirements.

N.º	Requirement
BAFR 01	Know the SCBA users and team composition
BAFR 02	Know the SCBA characteristics (capacity, level,..)
BAFR 03	Know the mission time of the SCBA users
BAFR 04	The system should know the level of the SCBA, the consumption rate and the distance from the exit to estimate the turn-around-time.
BAFR 05	Calculate the turn-around-time (TAT), display it on the screen and alert the SCBA Entry Control Officer and the firefighter when that time is reached.

4.3.5.5 Resources Management requirements

Table 4.5: Resources Management Functional Requirements.

N.º	Requirement
PMFR 01	Know the resources deployed in action and in reserve, and their commitment within the mission.
PMFR 02	Know the communication plan and the assigned communication channels.
PMFR 03	Record the existing command structure on site.

4.3.5.6 User Warnings requirements

Table 4.6: User Warnings Functional Requirements.

N.º	Requirement
UWFR 01	Status of all visual aids should be displayed to the controller
UWFR 02	Alarms and warnings to alert the end user of risk situations

Continued on next page

Table 4.6 – *Continued from previous page*

N.º	Requirement
UWFR 03	Alarms to warn the user: 1. Acoustic alarms 2. Visual alarms
UWFR 04	The user can disable alarms, but they will be activated again if no action is taken and the risk associated with alarm increases.

4.3.6 Non functional requirements

In a search and rescue scenario, all operations are controlled and coordinated by the commander of the rescue operations (COS) that should be aware of the conditions of the incident and register it in the SITAC form using adequate symbology [NOP09].

The situation of the incident should be represented according to the forms, diagrams and symbols used by firefighters for easy reading and operation.

4.3.6.1 End User Operation

Table 4.7: End User Operation non Functional Requirements.

N.º	Requirement
EUnFR 01	A sequence of actions should be designed to be logical from the perspective of the user, not from the perspective of computer processing or ease of programming
EUnFR 02	Interactive control logic should permit completion of a task with the minimum number of actions. However, this should not be to the detriment of situation awareness or consistency

4.3.6.2 Information Display

Information should be presented simply and in a well-organised. Ways to achieve simplicity include the following non functional requirements listed in Table 4.8:

Table 4.8: Information Display non Functional Requirements.

N.º	Requirement
IDnFR 01	The screen should be orderly
IDnFR 02	Information should be presented in consistent, predictable locations
IDnFR 03	The language used should be plain and simple
IDnFR 04	Data items on a screen should be grouped on logical principles basis
IDnFR 05	Information shall be presented to a user in directly usable form; a user shall not have to decode or interpret data.
IDnFR 06	Data should be gathered logically to help the user in his tasks
IDnFR 07	Fonts used for texts should be legible with adequate size and contrast
IDnFR 08	The data needed for an interaction shall be displayed in a directly usable form
IDnFR 09	Information density should be minimised, in particular, for displays used for critical task sequences
IDnFR 10	Paging or scrolling can be used when the amount of information to display does not allow display on a single page
IDnFR 11	A system shall interrupt a user only when necessary to guide the user for a response, to provide essential feedback, or to inform the user of errors
IDnFR 12	The wording of displayed data and labels shall be chosen to reflect the user's point of view and shall correspond to the user's operational language
IDnFR 13	Words in the command language dialog shall reflect the user's point of view and shall correspond to the user's operational language
IDnFR 14	The response time of a system to a user action shall be appropriate to this type of action.
IDnFR 15	A maximum response time shall be determined for each type of action.
IDnFR 16	In case of a system failure during an interaction, the system should keep, as far as possible, the maximum of information in order not to have to start again the same interaction.

Continued on next page

Table 4.8 – *Continued from previous page*

N.º	Requirement
IDnFR 17	Colours, different levels of brightness, blinking colours shall be used to the following aims: 1. Show the agents status on the ground 2. Draw attention to emergencies 3. Identify critical situations 4. Improve readability and perception of the end user
IDnFR 18	The information should be presented in a way that non critical objects do not overlap with critical objects
IDnFR 19	To provide a clear awareness of the situation on the ground, the following information shall be displayed: 1. Different symbols for firefighters and robots 2. Target information (position, speed, identification, task, status) on labels and windows 3. Conflict alerts (visual and audible)
IDnFR 20	The user should be able to select the view he/she want, but the critical warnings of other views should appear to draw the attention to a new event.

4.3.7 Taxonomy of information flows

The information sharing between human and robotic teams and the CC is extremely important and it is required to use a common base for information validation and communication between the used platforms. This common base should be appropriate for data-interchange, language independent and human-readable, with two possible solutions to solve this issue, eXtensible Markup Language (XML) and JavaScript Object Notation (JSON).

XML is a subset of the Standard Generalized Markup Language (SGML) and is a universal data representation format. It is a user-defined hierarchical data format and is used to create user-defined markups for documents and encoding schemes, applied in object serialization for data transfer between applications.

JSON is designed to be a light-weight data-interchange format and easy for humans

to understand. Although based on a subset of the JavaScript Programming Language, JSON is a text format that is language-independent, and is used for transmit data objects between applications, consisting of *attribute-value* pairs.

In an USAR scenario, fast and reliable communication is required and JSON claims a lightweight payload reducing bandwidth needs and allowing faster transfers, leading to lesser processing time [Cro06], but according to Lee in [Lee13], given a document object, one can produce identical sized JSON and XML representations. Furthermore, this representations compress to nearly identical size which is an indicator that they contain approximately the same entropy or information content, so transferring these documents to a wide variety of devices takes effectively the same time per device [Lee13].

The ROSBridge protocol grants access to the underlying Robot Operating System (ROS) messages and services as serialized JSON objects, providing communication among the different agents in the system through a single web socket connection (Figure 4.7) enabling non-ROS users to publish and subscribe topics and call services as a true ROS node, consequently the messages are described in JSON format.

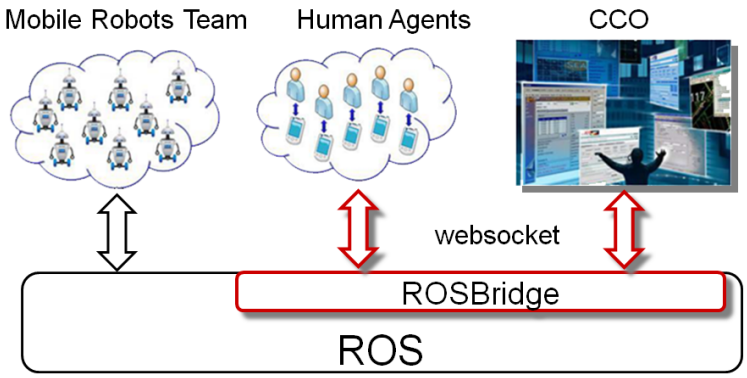


Figure 4.7: Communication structure of the different actors.

The analysis of the actions performed by agents on ground level, the relevant information to the performance of the mission and their interaction with the command center, leads to the identification of an initial set of messages for controlling and monitoring operations allowing collaborative context awareness. Messages are classified into five classes: *command & control*, *location*, *status*, *conditions* and *mapping*, according with their content, priority and function. The tree structure of the messages is presented in Figure 4.8.

The messages are published/subscribed as ROS topics, with a defined *namespace* to identify the topic publisher and a field to indicate the receiver classes with their respective id. The CC will always have *id*="0" and the other agents *id's* will start with "1". If a message is sent to a robot or firefighter with *id*="0", the message should

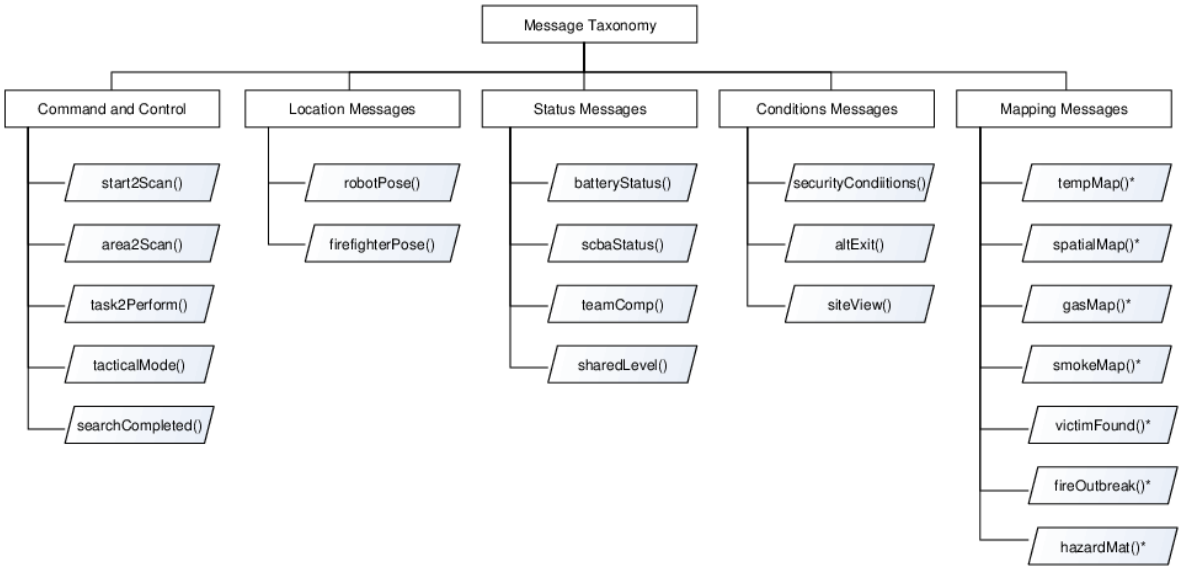


Figure 4.8: Tree structure of the messages between the CC, firefighters and robots (* indicates priority messages)

be broadcasted to all instances of the class. If an agent receive a ROS message with an $id \neq 0$ and different from its own id the message should be ignored.

A complete description of the messages is presented in B.

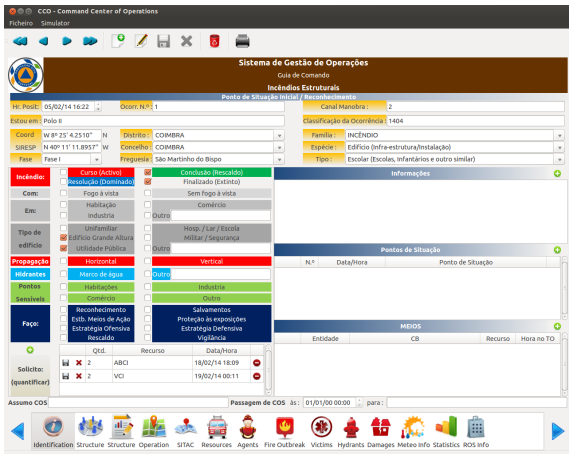
4.4 Design of the CHOPIN Command Center

4.4.1 Design of the CC human-machine interface

The incident commander size-up the incident and evaluate risk assessment by fulfilling the screen in Figure 4.9(a), this screen is based on an ANPC form, depending on the incident classification. This screen adopts the same layout and colors of the form to meet information display non-functional requirements.

The operations aiming the definition of the TO areas and sectors of the hot zone, the deployment of resources on the ground and the adopted tactical mode are achieved in the SITAC screen, adopting the ANPC form and graphical symbology already in use by the firefighters in their missions [Figure 4.9(b)].

The defined locations and designated resources are stored in the database and the information necessary to characterize the context is transmitted to the correspondent agents, minimizing one of the DOODA loop introduced delay.



(a) Incident identification window

(b) Graphical SITAC

Figure 4.9: Human Machine Interface of the CC.

As stated before, the incident structure is modified when the incident phase changes, therefore the database reflects the current structure of the incident with all the resources involved and the communication structure

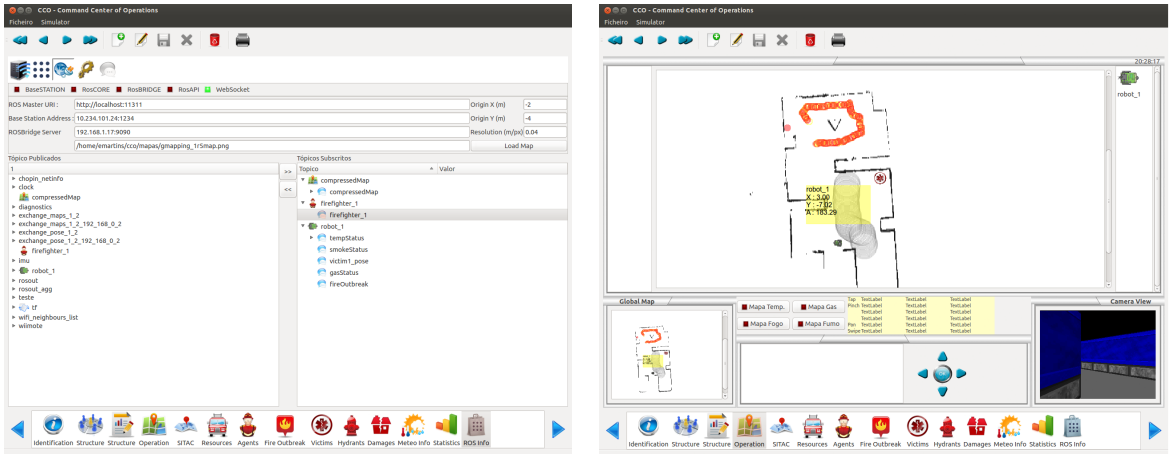
During the course of the mission, several occurrences are found and reported to the CCO, which are grouped according with their category and presented in proper layouts and registered in the database, alerting and asking the incident commander to take proper actions.

4.5 Interface with human and robotics first responders

The interface with human and robotic first responders in urban search and rescue activities represent a fundamental role in obtaining situation awareness. Whatever are the robots capabilities, the interface must provide the command center with sufficient information to make the correct decisions [MWH04]. High level in tasks autonomy of the robot relieves the commander of the operations to focus his attention in more high-level aspects.

The different requirements of the USAR mission determines different interfaces for different tasks, so it's critical that the commander of the operations keep a status awareness of the mission, so it's proposed an interface based on strategic games which have been developed in the last years for a better and intuitive interaction of the user and the scenario and their interveners, where it's possible to identify the agents, their location and status, as presented in Figure 4.10(b).

Communications with the BS and the ROS framework is achieved through the layout in Figure 4.10(a), where all the published topics and their publisher appear on the left and the subscribed topics appear on the right. The end user selects the desired topic to subscribe and move it to the subscribe window to start receiving information from the agent. Whenever a topic is subscribe, the publisher agent is added to the operational operational map, reflecting the agent status on the ground allowing the interaction between CCO and the agents.



(a) ROS layout (b) Operational Map

Figure 4.10: Human Robot Interaction of the CC

These automated tasks simplify the command and control of SCBA teams defining a time-based turning point for of the SCBA users.

Although robots try to find victims with their sensors, and the humans are commonly viewed as commanders, planners and managers, they are more noteworthy for their sensory feats [LW09] and could be helpful determining the victim condition, especially if the victim is trapped or unconscious, and only an arm or a leg is visible. So the interface is based on First-Person Shooter (FPS) video game which provides the primary source of surroundings awareness, as presented on the lower right corner of Figure 4.10(b).

Video and image transmission can overload the communication channel, therefore, the end user have the ability to connect/disconnect the video transmission.

4.6 Database requirements analysis

4.6.1 Conceptual cluster models

An USAR mission can involve a large number of resources and personnel, from different entities, and can attain a high degree of division complexity and communications channels can be overloaded by the information flow to report occurrences originated in the incident. Capability to register events is added to the CCO with a database to capture the structure of the mission and maintain historical data.

Due to the large number of entities involved, one can follow a top-down approach to define the conceptual data model using the *clustering concept* where collections of entities and relationships comprise higher-level objects, allowing to represent the entire database conceptual schema [Figure 4.11], where the entity clusters are depicted in yellow and maintain the same relationships between entities inside and outside the entity cluster, as occur between the same entities in the lower-level diagram. No attributes are presented in this higher-level diagram to provide a clear view of the major functionalities areas of the database.

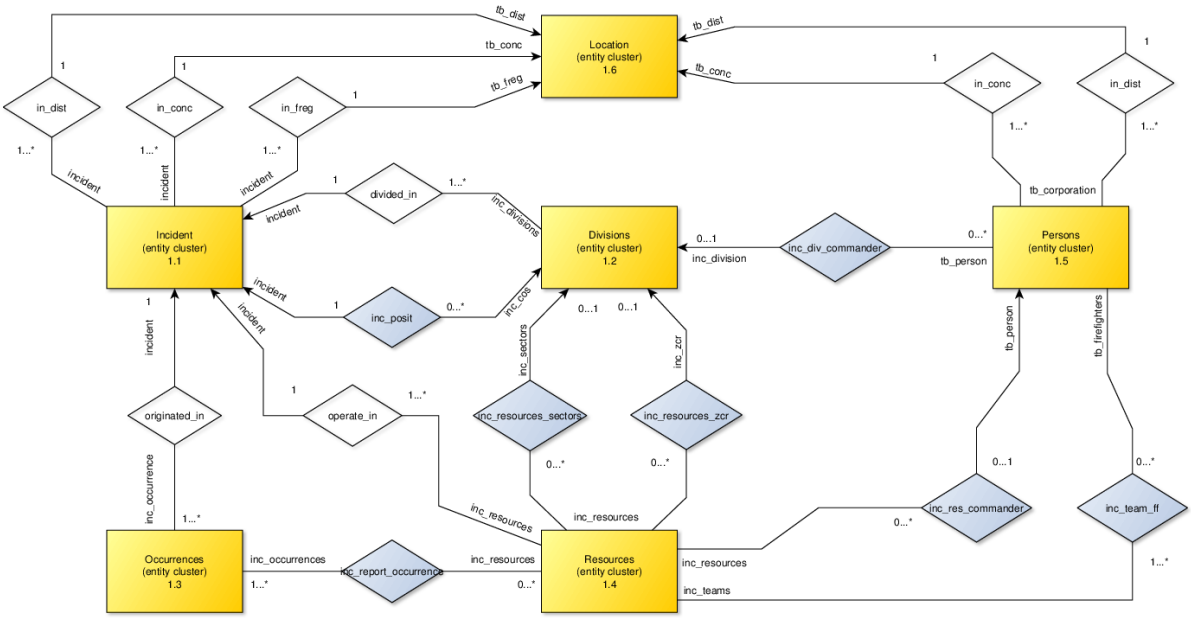


Figure 4.11: Database conceptual schema - *root entity cluster*.

From the database requirement analysis, six major groups and their relationships are identified; the incident itself, the deployed resources (teams, vehicles, robots), the personnel (decision makers, specialists, firefighters), the incident divisions (sectors, cells,

sections, areas), the occurrences originated by the incident (victims, fire outbreaks, tactics) and the location of the incident.

The incident occurs in a location classified by the portuguese regional country division, characterized by three relationships with the location cluster

Each division of the incident is commanded by a person of a firefighters corporation or an external entity and has a location in TO, changing over time depending on the incident evolution (*inc_div_commander*).

The resources working in the incident are located in a sector (*inc_sectors*) or in the concentration and reserve area (*inc_zcr*). The resources are commanded by a person, and eventually this commander can change over time, yielding a *1:M* relationship optional on both sides, therefore *inc_res_commander* holds the *tb_person* and *inc_resources primary keys*. The same rationale applies to the *all "many-to-many"* (*N:M*) relationships *inc_team_ff* and *inc_report_occurrence* regardless the involved entities are mandatory or optional. All actions are coordinated by the COS which have to communicate the *inc_posit* in a predefined time span. All the database clusters are expanded in appendix C.

4.6.2 Data modeling

The conceptual data model is implemented using Structured Query Language (SQL), by transforming the Entity-Relationship Diagrams (ERDs) into three different kind of tables addressing to the following rules:

- Tables with the same information content as the original entity, when the binary relationships are *many-to-many* (M:M), *one-to-many* (1:M) on the *one* side; or *one-to-one* (1:1) on either side - the only exception to this rule is when the relationship is 1:1 mandatory on both sides, and in this case the the two entities are transformed in only one table, and entities with any ternary or higher-degree relationship or generalization hierarchy.

- Tables with foreign key from the mandatory side of the relationship, when the binary relationships are *one-to-many* (1:M) for the entity on the M side; or *one-to-one* (1:1) for the entity that is not mandatory.

- Tables derived from relationship containing the foreign keys of all the entities in the relationship, when the binary relationships are *many-to-many* (M:M), relationships binary recursive and all relationships that are ternary or higher degree.

Whilst there are some dependent *weak* entities and relationships, unique identifiers are defined with an auto incremented field to overcome this constrain and streamline the programming work with the *model-view-delegate* in *Qt*.

4.7 Future work

To meet the requirements that guide the CHOPIN project and provide a wide and integrated context-awareness becomes of major relevance know the position of the firefighters on the ground, supported by the availability of multiple indoor positioning technologies and techniques and taken advantage of link quality parameter with the connected neighbours made available by the Mobile ad hoc Network developed in the project.

Aside the use of video cameras, one should consider the use of thermal cameras and/or other group of sensors with the ability to read and interpret classic fire behavior warnings, providing an augmented context-awareness useful to save firefighters lives.

5 Conclusion

This document reports the activities related with deliverable D2.2, which corresponds to the aggregation of four deliverables of mentioned on the project proposal: D2.1.2, D2.2.1, D2.3.1, D5.3. It presents the results of the work in the following points: the definition of context-awareness for this project, the use of a mobile phone for acquiring signals about the firemen activities, the integration SaR mobile robots in ROS, and the requirements analysis and design of the command center for this project.

A Use Cases and System Models

A.1 Use-case diagrams

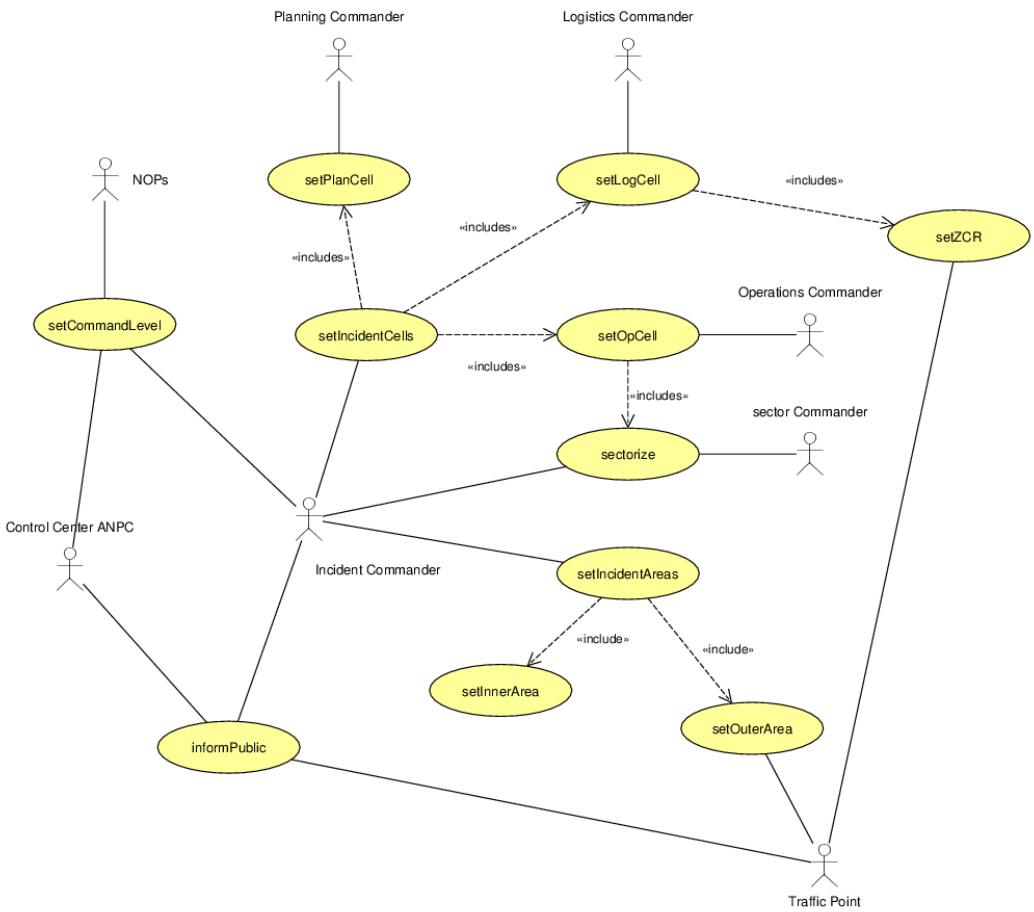


Figure A.1: Theatre of operations organization use-cases

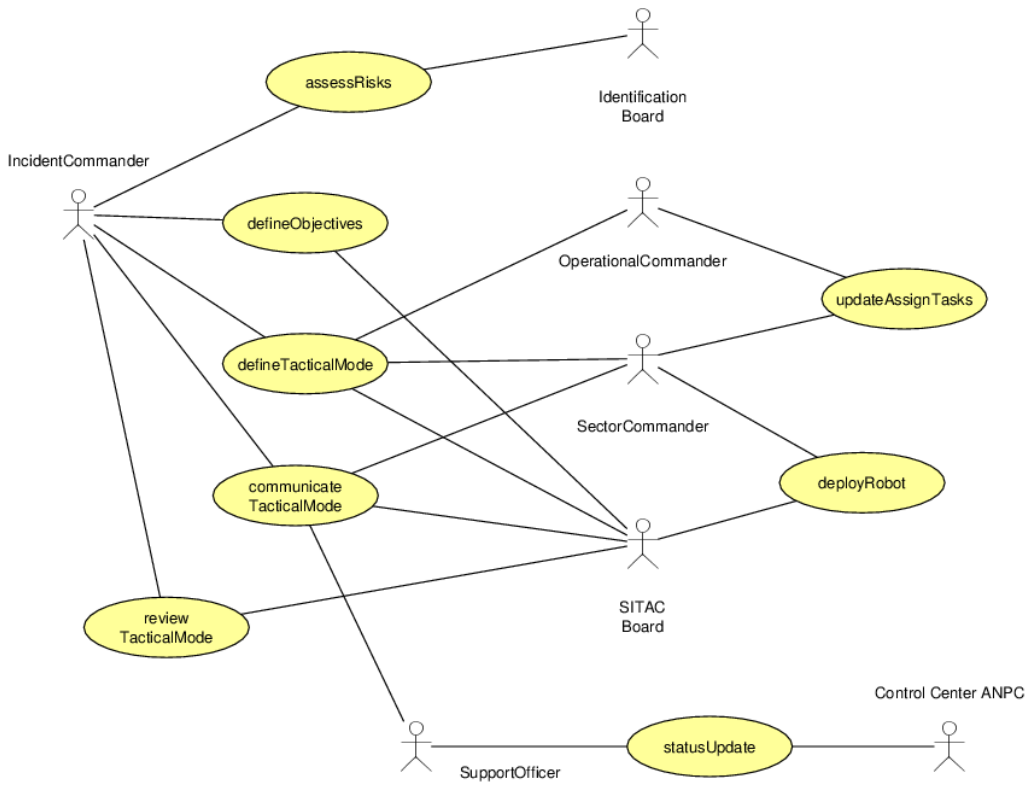


Figure A.2: Tactical Mode and Risk Assessment use-cases.

A.2 Class diagrams

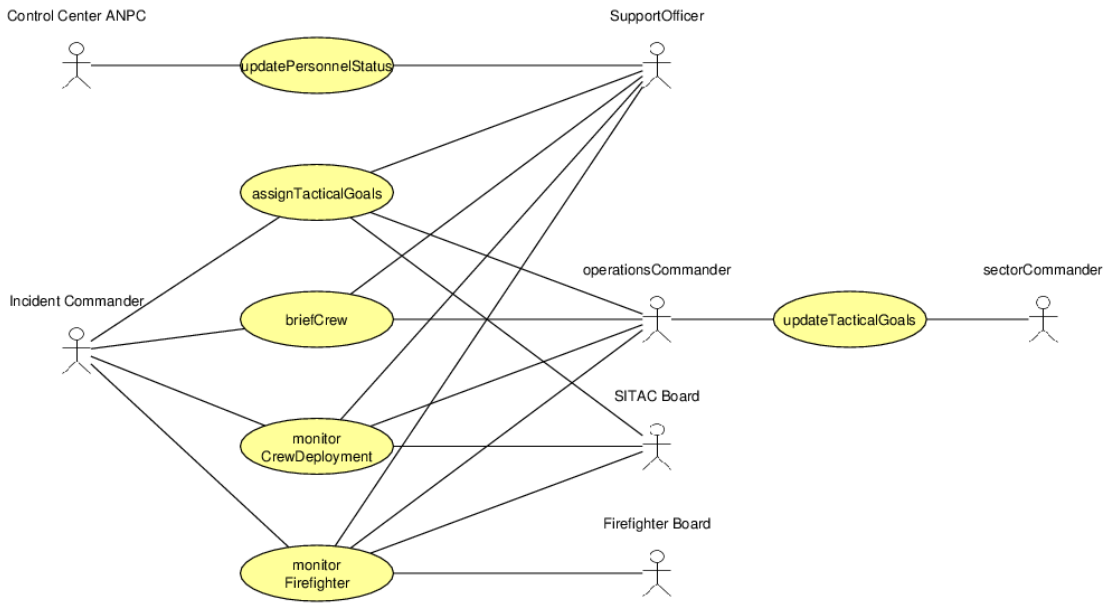


Figure A.3: Personnel Management use-cases

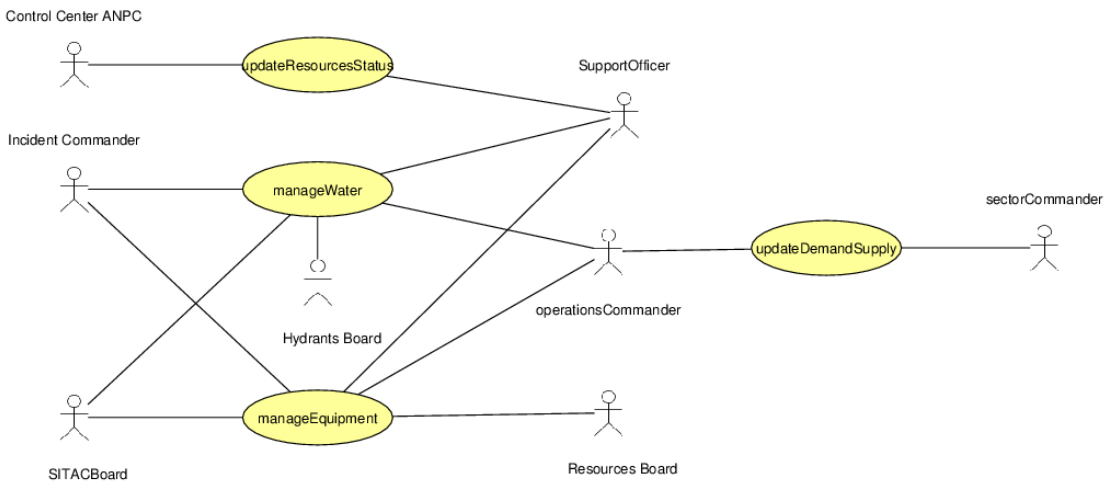


Figure A.4: Resources Management use-cases

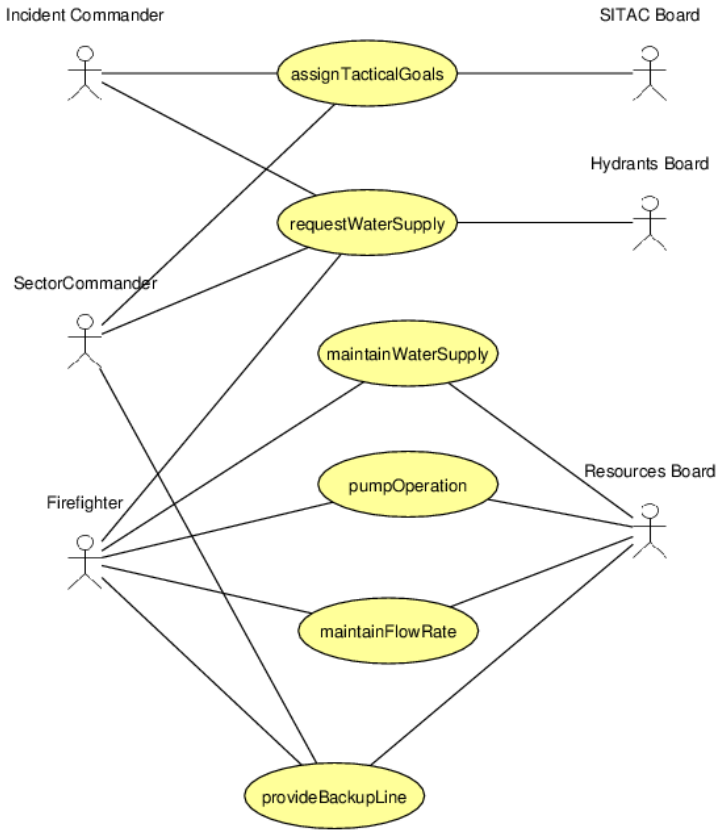


Figure A.5: Manage Water supply use-cases

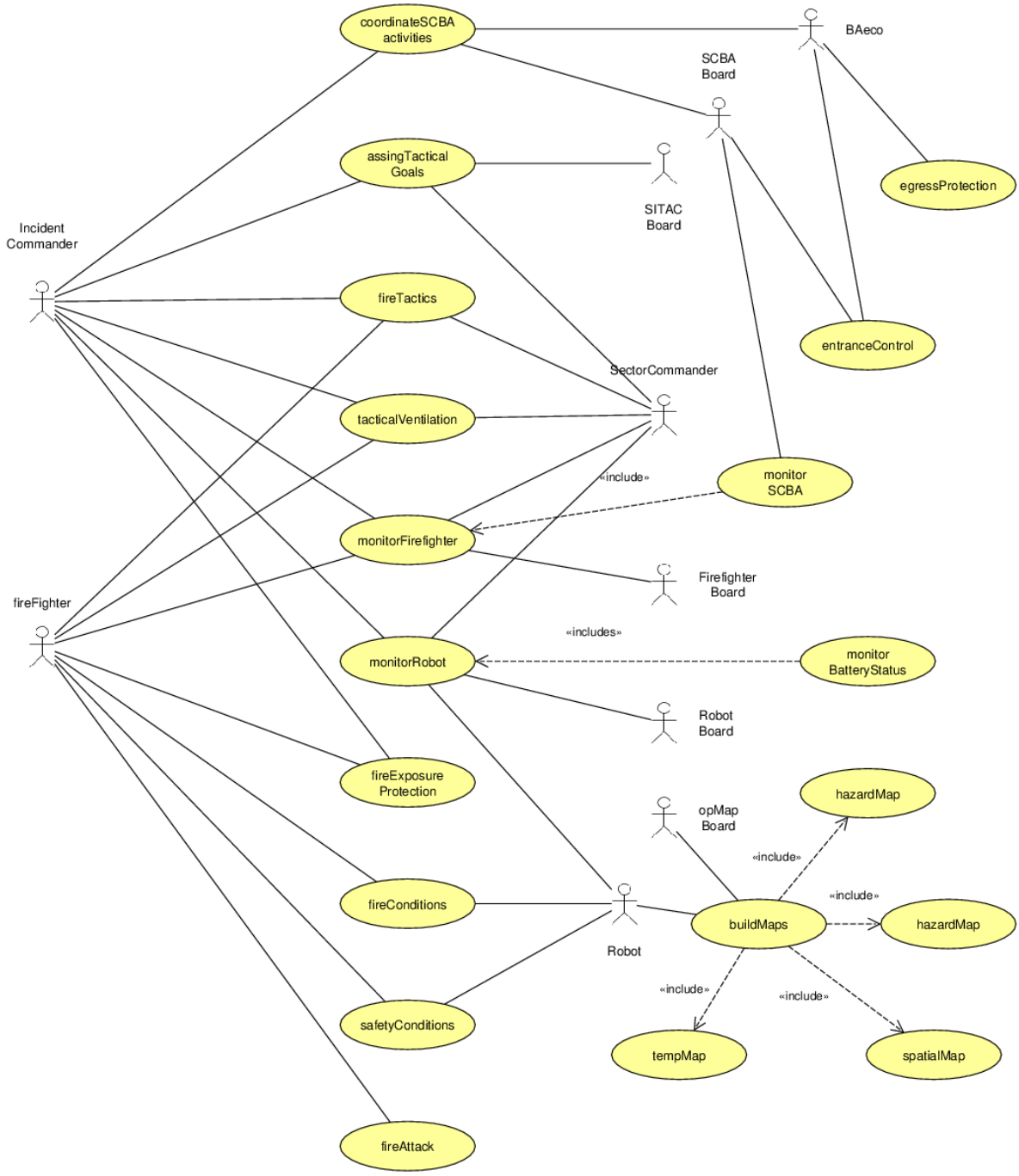


Figure A.6: Fire attack use-cases

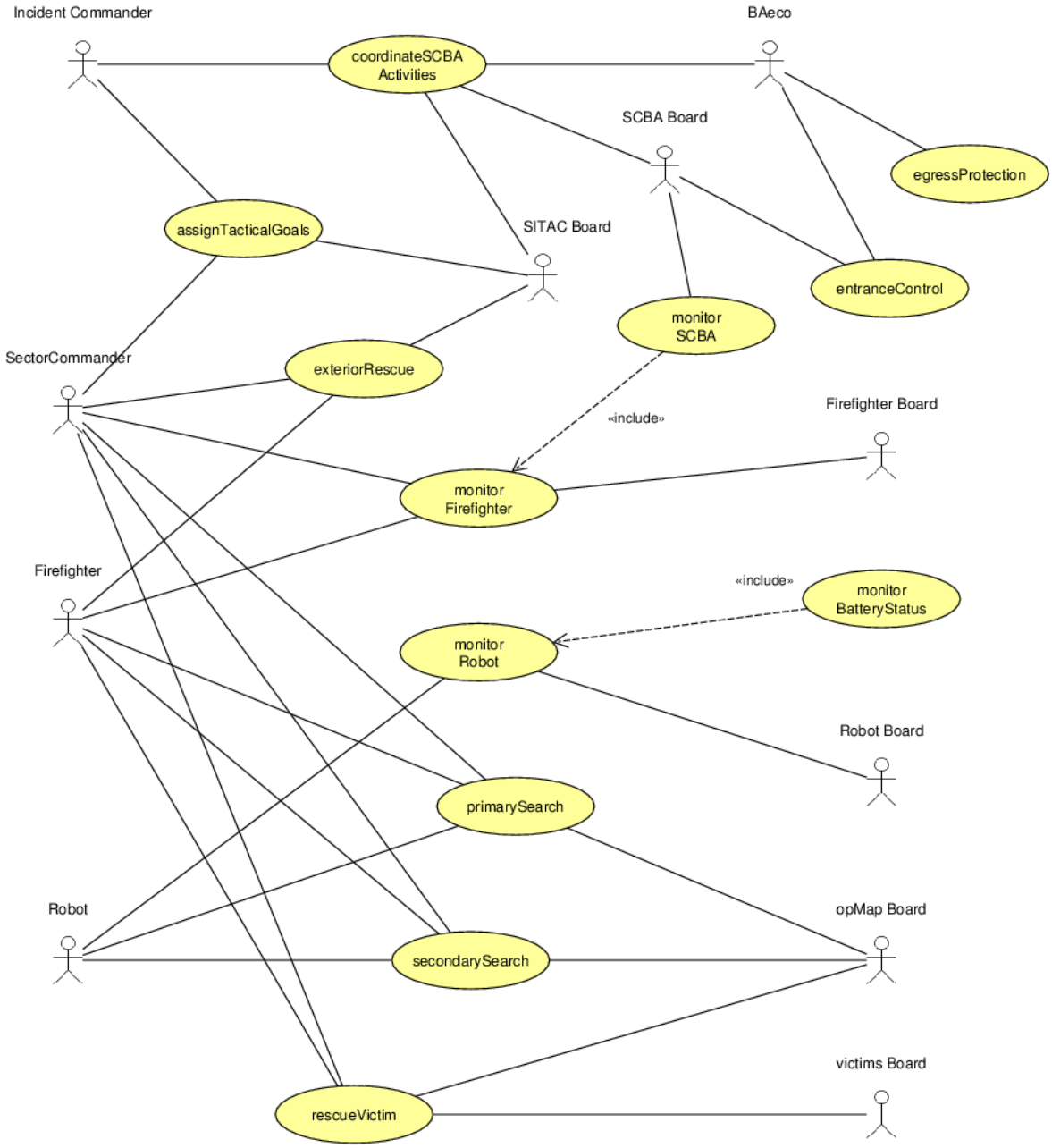


Figure A.7: Search and Rescue use-cases

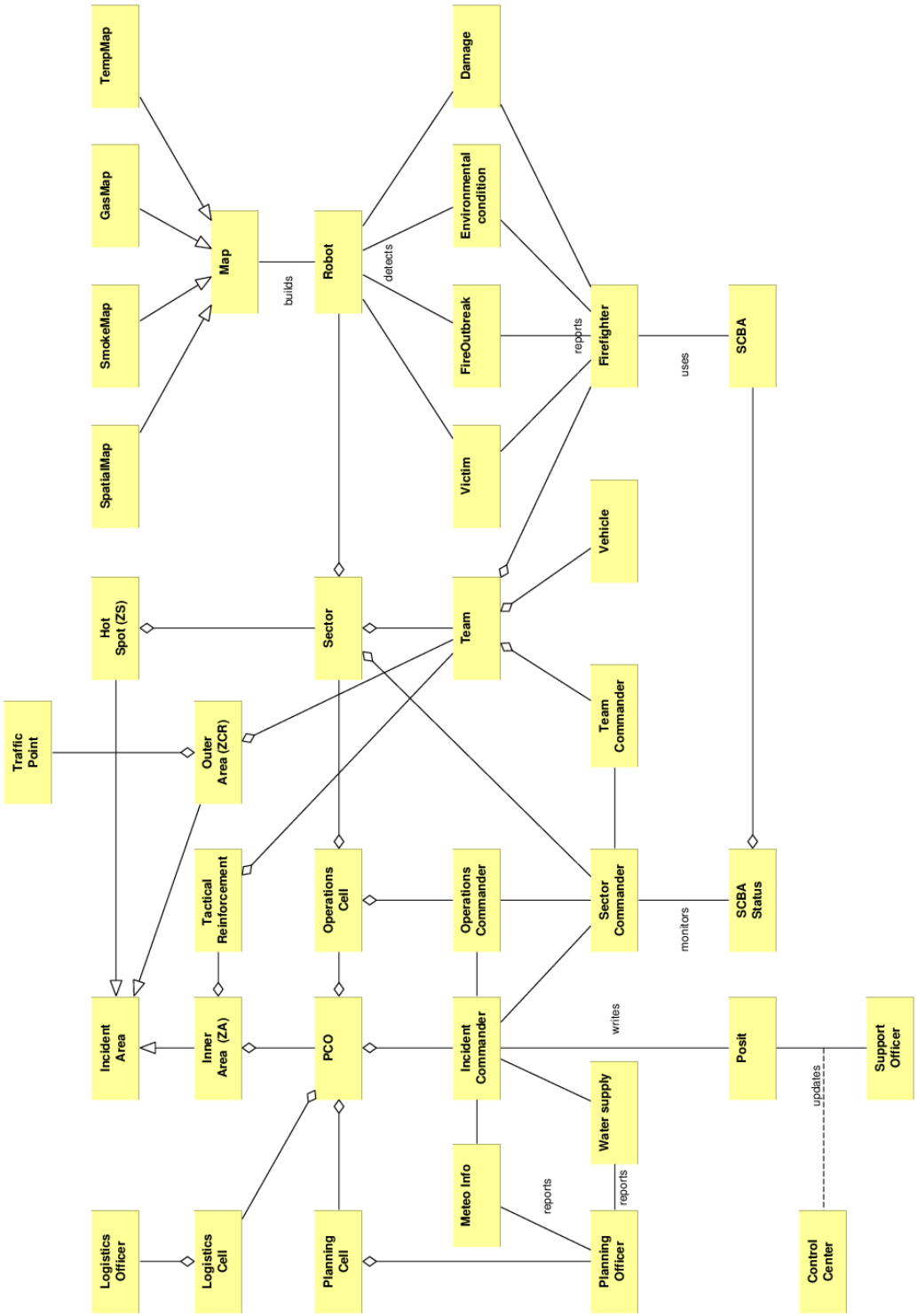


Figure A.8: Domain class Diagram

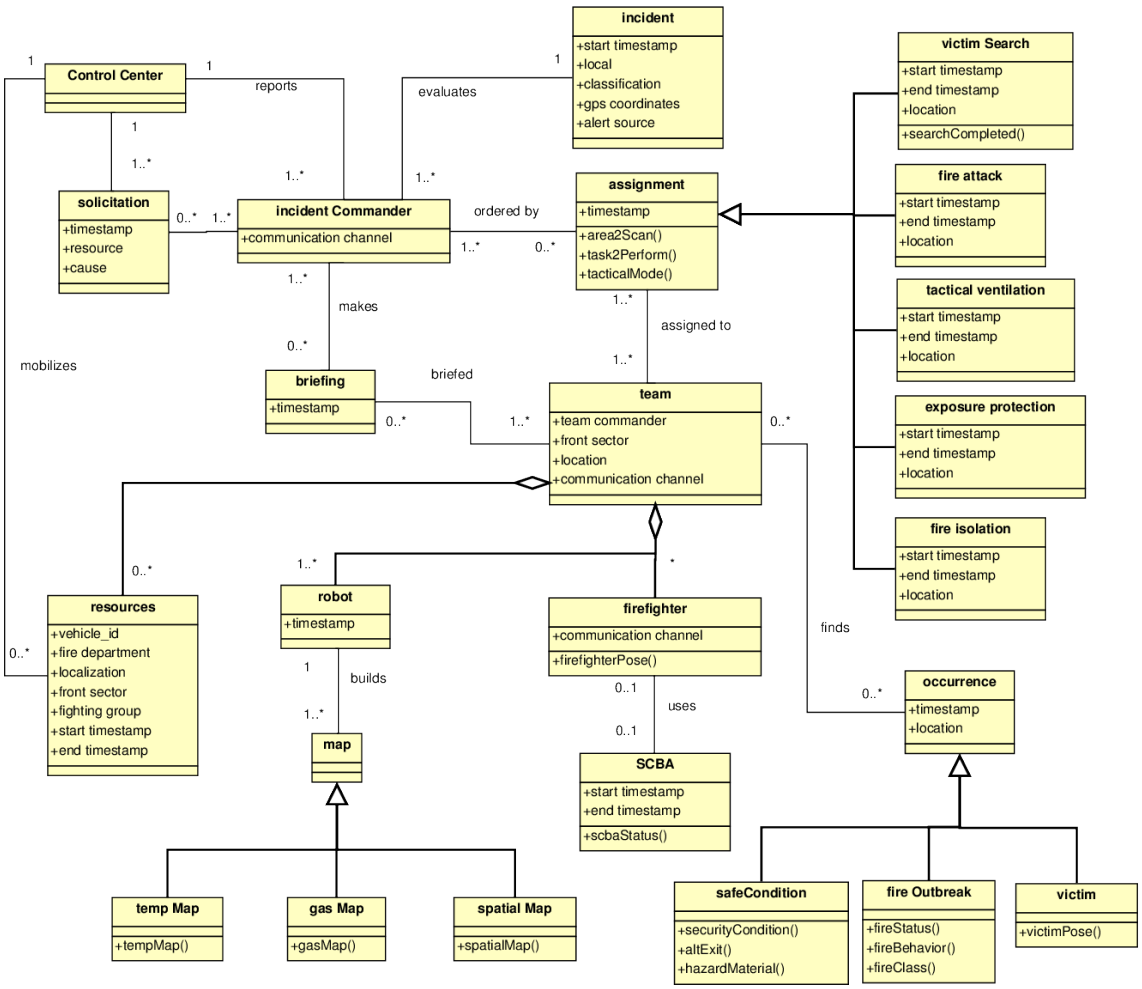


Figure A.9: Fire Attack and Search and Rescue class diagram

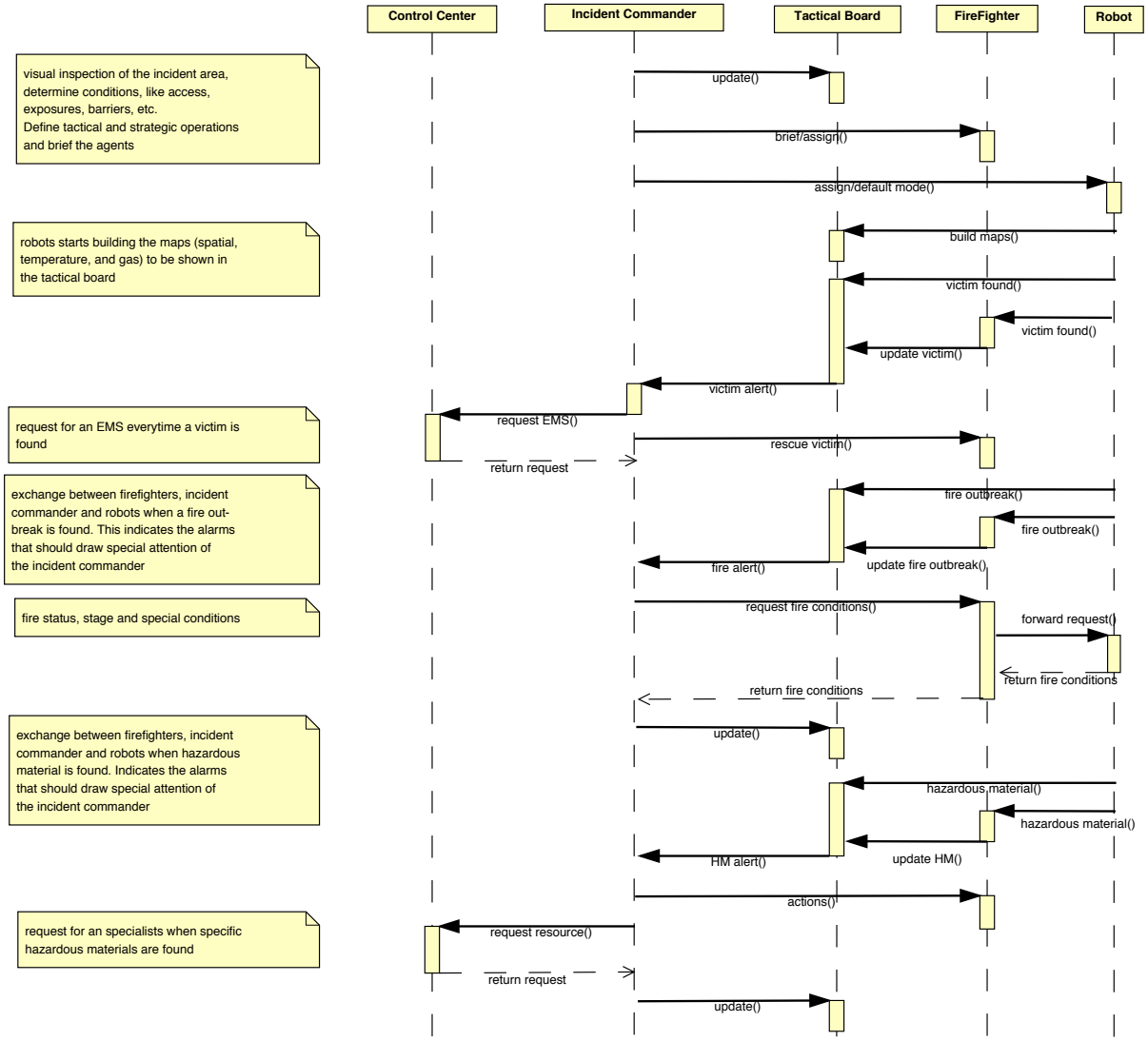


Figure A.10: Sequence Diagram of the Fire Attack and Search and Rescue operations

B Messages Definition

B.1 Command and Control Messages

Command and Control messages class allow the CC to change the goals of the mission, define new areas of interest and start a new mission for a robot, from a predefined location. During a search and rescue mission, scanned areas are signalized to avoid duplicated tasks.

B.1.1 *start2Scan()*

Description

Coordinates of the starting point of the robot in its exploit mission.

CC ⇒ robot

Message to define the starting point of the robots' mission when its deployed on an entrance of the structure to exploit, submitted only when required by the CC. The coordinates of the starting point is obtained by clicking on the desired point on Google Maps, and the robot must complete the initial pose from its magnetic compass.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	A Pose with reference coordinates frame and time stamp.

Listing B.1: *ROS message: start2scan()*

```
    Header                  header
    std_msgs/String         agent_class
    std_msgs/Int8           id
    geometry_msgs/Pose      pose
```

Json Message

Listing B.2: *Json message: start2scan()*

```
{ "topic": "cc_0/start2scan",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
```

```
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": 1,
            "seq": 232
        },"pose": {
            "position": {
                "x": 0.0017761230701580644,
                "y": -0.0019882202614098787,
                "z": 0.0
            },"orientation": {
                "y": ,
                "x": ,
                "z": ,
                "w":
            }
        }
    },"op": "publish"
}
```


B.1.2 *area2Scan()*

Description

Coordinates of an area of interest that should be scanned immediately. If one robot receives two or more messages of areas to scan, it should prioritize the orders addressing to the defined level of shared interaction and task criticality.

CC firefighter; **CC robot**; **firefighter robot**

Message to define a new area to scan, submitted only when required by the CC or the firefighter in the hot zone. This message is also used with the *site_view()* message to scan the area with the robot video camera.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	A Pose with reference coordinates frame and time stamp.

Listing B.3: *ROS message: area2scan()*

```
    Header                  header
    std_msgs/String         agent_class
    std_msgs/Int8           id
    geometry_msgs/Pose      pose
```

Json Message

Listing B.4: *Json message: area2scan()*


```
{ "topic": "robot_1/area2scan",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": 1,
            "seq": 232
        },"pose": {
            "position": {
                "x": 0.0017761230701580644,
                "y": -0.0019882202614098787,
                "z": 0.0
            },"orientation": {
                "y": ,
                "x": ,
                "z": ,
                "w":
            }
        }
    },"op": "publish"
}
```


B.1.3 *task2Perform()*

Task to perform - Action that should be carried out by the agent on the ground, whether it is a firefighter or a robot.

CC_firefighter; **CC_robot**

Action to be performed by firefighters and/or robots. Submitted only when required.
Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.

Continued on next page

Table B.3 – *Continued from previous page*

Ros type	Description
nav_msgs/Path	An array of poses that represents a path for a robot/firemen to follow.
actionlib_msgs/GoalStatus	*GoalID, status, PENDING, ACTIVE, PREEMPTED, SUCCEEDED, REJECTED, ABORTED, ...*

Listing B.5: *ROS message: task2perfom()*

```
Header                        header
std_msgs/String               agent_class
std_msgs/Int8                 id
nav_msgs/Path                 path
actionlib_msgs/GoalStatus     goal_status
```

Json Message

Listing B.6: *Json message: task2perform()*


```
{   "topic": "robot_1/task2perform",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "string",
            "seq": 232
        },"path": [{
                "pose": {
                    "header": {
                      "stamp": {
                        "secs": 1358783297,
                        "nsecs": 415561250
                      },"frame_id": "string",
                        "seq": 232
                    },"position": {
                        "x": 0.0017761230701580644,
                        "y": -0.0019882202614098787,
                        "z": 0.0
                    },"orientation": {
                        "y": 0.0,
                        "x": 0.0,
                        "z": -0.0001439173933738437,
                        "w": 0.9999999896438919
                    }
                }
            }
        ],"goal_status": {
            "id": "string",
            "status": 5,
            "text": "string"
```

```
        }
    },"op": "publish"
}
```

B.1.4 *tacticalMode()*

Ground teams operation mode (e.g. "defensive", "offensive" or "transactional") define by the CCO.

CC_firefighter; **CC_robot**

Message submitted only when required, to change the teams operating mode due to tactics change.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
actionlib_msgs/GoalID	The stamp should store the time at which this goal was requested.

Listing B.7: *ROS message: tacticalMode()*

```
Header                  header
std_msgs/String         agent_class
std_msgs/Int8           id
actionlib_msgs/GoalID   goal_id
```

Json Message

Listing B.8: *Json message: tacticalMode()*

```
{
    "topic": "cc_0/tactical_mode",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "goal_id": {"stamp" : "time",
                    "id"    : "string"
```

```
                    }
    },
    "op": "publish"
}
```

B.1.5 *searchCompleted()*

Notifies a complete area search, with or without victims, to prevent other agents to duplicate the task.

robot_firefighter; **robot_CC**; **firefighter_CC**

Message responsible to prevent duplicated searches, optimizing the use of resources. Should be sent whenever the area search is completed.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
nav_msgs/GridCells	An array of cells in a 2D grid.
actionlib_msgs/GoalStatus	*GoalID, status, PENDING , ACTIVE, PREEMPTED, SUCCEEDED, REJECTED, ABORTED, ...*

Listing B.9: *ROS message: searchCompleted()*

```
    Header                      header
    std_msgs/String             agent_class
    std_msgs/Int8               id
    nav_msgs/GridCells          searched_area
    actionlib_msgs/GoalStatus   goal_status
```

Json Message

Listing B.10: *Json message: searchCompleted()*

```
{
    "topic": "robot_1/searchCompleted",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
```

```
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "searched_area": {
            "header": {
                "seq"  : uint32,
                "stamp": time,
                "frame_id": string
                },
            "cell_witdh" : float32,
            "cell_height": float32,
            "cells" : [{"x":float64,
                        "y":float64,
                        "z":float64
                      }]
            },
        "goal_status": {
            "id": "string",
            "status": 5,
            "text": "string"
        }
    },
    "op": "publish"
}
```


B.2 Location messages

Information class messages are divided into two classes: conditions messages - to help classify the incident and define tactical strategies of operation; and location messages - to locate the agents, victims and relevant events in the incident zone.

B.2.1 *robotPose()*

Pose coordinates of the robot on the ground.

robot_firefighter; **robot_CC**

Message to send pose coordinates of the robot. This is a priority message and should be sent frequently, to provide CC and fire fighters with situation awareness of the robot.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	A Pose with reference coordinates frame and time stamp.

Listing B.11: *ROS message: robotPose()*

```
Header              header
std_msgs/String     agent_class
std_msgs/Int8       id
geometry_msgs/Pose  pose
```

Json Message

Listing B.12: *Json message: robotPose()*


```
{
    "topic": "robot_1/robot_pose",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": 1,
            "seq": 232
        },
        "pose": {
            "position": {
                "y": -0.0019882202614098787,
                "x": 0.0017761230701580644,
                "z": 0.0
            },
            "orientation": {
                "y": 0.0,
                "x": 0.0,
                "z": -0.0001439173933738437,
                "w": 0.9999999896438919
            }
        }
    },
    "op": "publish"
}
```

B.2.2 *firefighterPose()*

Firefighter pose - pose coordinates of the firefighter on the ground and its way of progression (e.g. standing, climbing, descending, crouched, "on all fours").

firefighter_robot; **firefighter_CC**

Message to send pose coordinates of the firefighter. This is a priority message and should be sent frequently, to provide the CC and robots with situation awareness of the firefighter.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	A Pose with reference coordinate frame and time stamp.
std_msgs/Int16.	Integers type message, to enumerate fireman state, id, etc.

Listing B.13: *ROS message: firefighterPose()*

```
Header                  header
std_msgs/String         agent_class
std_msgs/Int8           id
geometry_msgs/Pose      pose
float32                 situation
```

Json Message

Listing B.14: *Json message: firefighterPose()*


```
{
    "topic": "firefighter_1/firefighterPose",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "pose": {
            "position": {
```

```
                "y": -0.0019882202614098787,
                "x": 0.0017761230701580644,
                "z": 0.0
            },
            "orientation": {
                "y": 0.0,
                "x": 0.0,
                "z": -0.0001439173933738437,
                "w": 0.9999999896438919
            }
        },
        "situation": "number"
    },
    "op": "publish"
}
```


B.3 Status messages

B.3.1 *batteryStatus()*

Robot battery level. Message defined to monitor the robot battery level, to allow the CC to provide a safe exit for the robot.

robot CC

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
pr2_msgs/PowerState	This message communicates the state of the robot's power system.

Listing B.15: *ROS message: batteryStatus()*

```
std_msgs/Header      header
std_msgs/String      agent_class
std_msgs/Int8        id
pr2_msgs/PowerState  status
```

Json Message

Listing B.16: *Json message: batteryStatus()*


```
{
    "topic": "robot_1/batteryStatus",
    "msg": {
        "agent_class" : "robot",
```

```
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "status": {
            "power_consumption": "number",
            "time_remaining": "number",
            "prediction_method": "string",
            "relative_capacity": "number"
        }
    }
    "op": "publish"
}
```

B.3.2 *scbaStatus()*

Firefighters SCBA level and mission time using SCBA.

firefighter_CC

Message defined to monitor the firefighters SCBA level and mission time, to allow the CC to provide a safe exit for the firefighter.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
pr2_msgs/PowerState	This message communicates the state of the scba system.

Listing B.17: *ROS message: scbaStatus()*

```
    Header                  header
    std_msgs/String         agent_class
    std_msgs/Int8           id
    pr2_msgs/PowerState     status
```

Json Message

Listing B.18: *Json message: scbaStatus()*

```
{
    "topic": "firefighter_1/scbaStatus",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "status": {
            "power_consumption": "number",
            "time_remaining": "number",
            "action": "string",
            "relative_capacity": "number"
        }
    }
    "op": "publish"
}
```

B.3.3 *teamComp()*

Description

Composition of teams

CC_robot,**CC_firefighters**

Message with HRI purposes, to indicate the teammates of the robot or firefighter.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
std_msgs/String[]	An array of string with the identifiers of the teammates of the agent.

Listing B.19: *ROS message: teamComp()*

```
    Header              header
    std_msgs/String     agent_class
```

```
    std_msgs/Int8       id
    std_msgs/String[]   team
```

Json Message

Listing B.20: *Json message: teamComp()*

```
{
    "topic": "robot_1/teamComp",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "team": ["robot_2",
                 "robot_5",
                 "firefighter_3"
                ]
    },
    "op": "publish"
}
```

B.3.4 *sharedLevel()*

Description

Shared level of interaction between humans and robots

CC_robot

Message to define the shared level of interaction between robots and human teams

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
std_msgs/Int8	An integer to indicate the shared level of interaction with the operators. *e.g. 1- 1H:1R; 2- 1H:RT; 3- 1H:MR; 4- HT:1R; . . .*, where 1H: one human, 1R: one robot, MR: multiple robots, RT: robot team and HT: human team.

Listing B.21: *ROS message: sharedLevel()*

```
Header              header
std_msgs/String     agent_class
std_msgs/Int8       id
std_msgs/Int8       sharedlevel
```

Json Message

Listing B.22: *Json message: sharedLevel()*

```
{
    "topic": "robot_1/gas_map",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "string",
            "seq": 232
        },
        "sharedLevel": number
    },
    "op": "publish"
}
```

B.3.5 *netStats()*

Network Stats - message with network connection metrics from the node to the CCO and from the node to their neighbours.

robot_CC; **robot_firefighter**

This message should be sent whenever there is a network connection.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.

Listing B.23: *ROS message: netStats()*

```
Header                  header
std_msgs/String         agent_class
std_msgs/Int8           id
chopin/Vector3Stamped   quality
```

```
    chopin/Path              metrics
    wifi_comm/WiFiNeighbors neighbors
```

Json Message

Listing B.24: *Json message: netStats()*

```
{
    "topic": "network/stats",
    "msg": {
        "agent_class" : "cco",
        "id" : 0,
        "quality": {
            "link": "number",
            "level": "number",
            "noise": "number"
        },
        "metrics": {
            "tx": "number",
            "rx": "number"
        },
        "neighbors":{[
                "items":{
                "ip": "string",
                "quality":{
                    "link": "number",
                    "level": "number",
                    "noise": "number"
                },
                "metrics": {
                    "tx": "number",
                    "rx": "number"
                }
            }
        ]}
    },
    "op": "publish"
}
```

B.4 Conditions messages

B.4.1 *securityCondition()*

Security conditions - Indicates the firefighter if it is safe to move in a direction or enter an area for the known fire conditions. (E.g. *"climb above a fire without a charged hose - unsafe, do not proceed"*).

robot firefighter; **robot CC**; **firefighter CC**

Message to define safety conditions of an area which is about to be scanned by a firefighter. This message should be sent while firefighters move forward on the ground.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	A Pose with reference coordinate frame and time stamp.
std_msgs/Int16	Simple float type message, to enumerate situation.

Listing B.25: *ROS message: securityConditions()*

```
Header header
std_msgs/String    agent_class
std_msgs/Int8      id
geometry_msgs/PoseStamped[] poses
```

Json Message

Listing B.26: *Json message: securityConditions()*


```
{
    "topic": "robot_1/securityConditions",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "pose": {
            "position": {
                "y": -0.0019882202614098787,
                "x": 0.0017761230701580644,
                "z": 0.0
            },
            "orientation": {
                "y": 0.0,
                "x": 0.0,
                "z": -0.0001439173933738437,
                "w": 0.9999999896438919
            }
        } ,
        "security_conditions": 2
    },
    "op": "publish"
}
```

B.4.2 *altExit()*

Detection of doors and/or windows that allow the escape of fire in adverse situations. Always depends on the maps defined by agents nearby.

robot firefighter; **robot CC**; **firefighter CC**

Information message to signal a room secondary exit or an egress exit determining a safety path for firefighters and victims. This message should be sent whenever one of these exits is found.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	A Pose with reference coordinates frame and time stamp.
std_msgs/Int16	Simple float type message, to enumerate situation.

Listing B.27: *ROS message: altExit*

```
Header header
    std_msgs/String    agent_class
    std_msgs/Int8      id
geometry_msgs/Pose pose
float32 gas
```

Json Message

Listing B.28: *Json message: altExit()*

```
{
    "topic": "robot_1/altExit",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "pose": {
            "position": {
                "y": -0.0019882202614098787,
                "x": 0.0017761230701580644,
```

```
                "z": 0.0
            },
            "orientation": {
                "y": 0.0,
                "x": 0.0,
                "z": -0.0001439173933738437,
                "w": 0.9999999896438919
            }
        },
        poses: [
            {"pose": {
                "header": {
                    "stamp": {
                        "secs": 1358783297,
                        "nsecs": 415561250
                    },
                    "frame_id": "string",
                    "seq": 345
                },
                "position": {
                    "y": -0.0019882202614098787,
                    "x": 0.0017761230701580644,
                    "z": 0.0
                },
                "orientation": {
                    "y": 0.0,
                    "x": 0.0,
                    "z": -0.0001439173933738437,
                    "w": 0.9999999896438919
                }
            },
            "exit": 0
        }]
    },
    "op": "publish"
}
```


B.4.3 *siteView()*

Image sequence to allow the human user the fire conditions and/or identify a victim condition.

robot firefighter; **robot CC**

Image message to provide primary source of surroundings awareness to the CC and firefighters, for victim search and safety conditions.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
sensor_msgs/CompressedImage	This message contains a compressed image.

Listing B.29: *ROS message: siteView()*

```
Header                      header
std_msgs/String             agent_class
std_msgs/Int8               id
sensor_msgs/CompressedImage image
```

Json Message

Listing B.30: *Json message: siteView()*

```
{
    "topic": "robot_1/siteView",
    "msg": {
        "agent_class" : "cco",
        "id" : 0,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "string",
            "seq": 232
        },
        "image":{
            "header": {
                "stamp": {
                    "secs": 1358783297,
                    "nsecs": 415561250
                },
                "frame_id": "string",
                "seq": 232
            }
            "format": "string",
            "data":["number"]
        }
    },
    "op": "publish"
}
```

B.5 Mapping messages

Mapping messages class are used only by robots providing information about the interior of the building, aiding the CC to establish eventual hazardous conditions for the firefighters and victims.

B.5.1 *compressedMap()*

Spatial map - 2D grid map to identify the location of the agents in the building, entry and egress accesses and respective conditions.

robot CC; **robot fireghter**; **CC firefighter**

Priority message to build the interior map and provide the CC with situation awareness of the incident. This message should be sent whenever there is a network connection.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
nav_msgs/MapMetaData	This represents the map *metadata* information.
geometry_msgs/Point	A point to represent top-left coordinate of the transmitted window
geometry_msgs/Point	A point to represent bottom-right coordinate of the transmitted window
std_msgs/ByteMultiArray	A byte array with the occupancy-grid compressed data.

Listing B.31: *ROS message: compressedMap()*

```
Header header
std_msgs/String    agent_class
std_msgs/Int8      id
nav_msgs/MapMetaData info
geometry_msgs/Point iniwindow
geometry_msgs/Point endwindow
std_msgs/ByteMultiArray data
```

Json Message

Listing B.32: *Json message: compressedMap()*


```
{
    "topic": "robot_1/compressedMap",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "info": {
            "map_load_time": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "resolution": float64,
            "width": uint32,
            "height": uint32,
            "origin":{
                "position": {
                    "y": -0.0019882202614098787,
                    "x": 0.0017761230701580644,
                    "z": 0.0
                },
                "orientation": {
                    "y": 0.0,
                    "x": 0.0,
                    "z": -0.0001439173933738437,
                    "w": 0.9999999896438919
                }
            }
        } ,
        "iniwindow": {  "x": number,
                        "y": number,
                        "z": 0 },
        "endwindow": {  "x": number,
                        "y": number,
                        "z": 0 },
        "data": [uint8]
    },
    "op": "publish"
}
```


B.5.2 *tempMap()*

Temperature map - 2D grid map with temperature levels to identify hazardous conditions in the area.

robot CC; **robot firefighter**; **CC firefighter**

Priority message to build the temperature map and provide the CC with heat and hazardous awareness. This message should be sent whenever there is a network connection.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
nav_msgs/OccupancyGrid	This represents a 2-D grid map, in which each cell represents the probability of occupancy.

Listing B.33: *ROS message: tempMap()*

```
Header header
    std_msgs/String    agent_class
    std_msgs/Int8      id
geometry_msgs/Pose pose
float32 temp
```

Json Message

Listing B.34: *Json message: tempMap()*


```
{
    "topic": "robot_1/tempMap",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "info": {
            "map_load_time": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "resolution": float64,
            "width": uint32,
            "height": uint32,
            "origin":{
                "position": {
                    "y": -0.0019882202614098787,
                    "x": 0.0017761230701580644,
                    "z": 0.0
```

```
                },
                "orientation": {
                    "y": 0.0,
                    "x": 0.0,
                    "z": -0.0001439173933738437,
                    "w": 0.9999999896438919
                }
            }
        } ,
        "data": [uint8]
    },
    "op": "publish"
}
```


B.5.3 *gasMap()*

Gas map - 2D grid map with the type of gas and respective concentration (e.g. CO, CO_2[organic material], HCN [acrylic fibers, polyurethane or nylon], HCI and $COCI_2$ [PVC, vinyl wallpaper and cable installation].

robot_CC; **robot_firefighter**; **CC_firefighter**

Priority message to build the temperature map and provide the CC with gas concentration and hazardous awareness. This message should be sent whenever there is a network connection.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
nav_msgs/OccupancyGrid	This represents a 2-D grid map, in which each cell represents the probability of occupancy.

Listing B.35: *ROS message: gasMap()*

```
Header header
std_msgs/String     agent_class
std_msgs/Int8       id
geometry_msgs/Pose  pose
float32 gas
```

Json Message

Listing B.36: *Json message: gasMap()*


```
{
    "topic": "robot_1/gasMap",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "info": {
            "map_load_time": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "resolution": float64,
            "width": uint32,
            "height": uint32,
            "origin":{
                "position": {
                    "y": -0.0019882202614098787,
                    "x": 0.0017761230701580644,
                    "z": 0.0
                },
                "orientation": {
                    "y": 0.0,
                    "x": 0.0,
                    "z": -0.0001439173933738437,
                    "w": 0.9999999896438919
                }
            }
        } ,
        "data": [uint8]
    },
    "op": "publish"
}
```


B.5.4 *victimFound()*

Victim pose - pose coordinates of the victim on the ground and its state condition (e.g. *"aware"*, *"semi-aware"*, *"unconscious"*, *"unknown"*), situation condition (e.g. *"surface"*, *"trapped"*, *"entombed"*, *"unknown"*).

robot CC; **robot firefighter**; **firefighter CC**

Message to send pose coordinates of the victim. This is a priority message to be sent whenever a victim is found, and provide the CC and firefighters with situation awareness of the victim.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	An array of Poses with reference coordinates frame and time stamp.
std_msgs/Int16MultiArray	Array of integers type message, to enumerate victim state,id,etc.

Listing B.37: *ROS message: victimFound()*

```
Header header
    std_msgs/String    agent_class
    std_msgs/Int8      id
    geometry_msgs/PoseStamped[] poses
    std_msgs/Int16MultiArray status
```

Json Message

Listing B.38: *Json message: victimFound()*


```
{
    "topic": "robot_1/victimFound",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "victims": [{
                "header": {
                    "stamp": {
                        "secs": 1358783297,
                        "nsecs": 415561250
                    },
                    "frame_id": "map",
                    "seq": 232
                },
                "position": {
                    "y": -0.0019882202614098787,
                    "x": 0.0017761230701580644,
                    "z": 0.0
                },
                "orientation": {
                    "y": 0.0,
                    "x": 0.0,
                    "z": -0.0001439173933738437,
```

```
                    "w": 0.9999999896438919
                }]
            } ,
        "status":[{
            "state"     : number,
            "situation": number,
            "condition": number,
            "type"      : number
        }]
    },
    "op": "publish"
}
```

B.5.5 *smokeMap()*

Hazard condition - position of materials and hazardous conditions (e.g. containers of flammable or explosive materials near fire outbreak that requires special attention or imminent "*BLEVE*").

robot_CC; **robot_firefighter**; **firefighter_CC**

Message to send location coordinates of the hazardous materials and conditions. This is a priority message and should be sent whenever these conditions are found, to provide the CC and firefighters with situation awareness of the overall safety conditions.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	An array of Poses with reference coordinates frame and time stamp.
std_msgs/Int16MultiArray	Array of integers type message, to enumerate victim state,id,etc.

Listing B.39: *ROS message: smokeMap()*

```
Header header
std_msgs/String     agent_class
std_msgs/Int8       id
geometry_msgs/PoseStamped[] poses
std_msgs/Int16MultiArray status
```

Json Message

Listing B.40: *Json message: smokeMap()*


```
{
    "topic": "robot_1/smokeMap",
    "msg": {
        "agent_class" : "cco",
        "id" : 0,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "info": {
            "map_load_time": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "resolution": float64,
            "width": uint32,
            "height": uint32,
            "origin":{
                "position": {
                    "y": -0.0019882202614098787,
                    "x": 0.0017761230701580644,
                    "z": 0.0
                },
                "orientation": {
                    "y": 0.0,
                    "x": 0.0,
                    "z": -0.0001439173933738437,
                    "w": 0.9999999896438919
                }
            }
        } ,
        "data": [uint8]
    },
    "op": "publish"
}
```


B.5.6 *hazardMat()*

Hazard condition - position of materials and hazardous conditions (e.g. containers of flammable or explosive materials near fire outbreak that requires special attention or imminent "*BLEVE*").

robot CC; **robot firefighter**; **firefighter CC**

Message to send location coordinates of the hazardous materials and conditions. This is a priority message and should be sent whenever these conditions are found, to

provide the CC and firefighters with situation awareness of the overall safety conditions.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	A Pose with reference coordinate frame and time stamp.
std_msgs/Int16MultiArray	Array of integers type message, to enumerate victim state,id,etc.

Listing B.41: *ROS message: hazardMat()*

```
Header header
std_msgs/String    agent_class
std_msgs/Int8      id
geometry_msgs/Pose[] pose
std_msgs/Int16MultiArray status
```

Json Message

Listing B.42: *Json message: hazardMat()*


```
{
    "topic": "robot_1/hazardMap",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "info": {
            "map_load_time": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "resolution": float64,
            "width": uint32,
            "height": uint32,
            "origin":{
                "position": {
                    "y": -0.0019882202614098787,
                    "x": 0.0017761230701580644,
                    "z": 0.0
                },
                "orientation": {
```

```
                    "y": 0.0,
                    "x": 0.0,
                    "z": -0.0001439173933738437,
                    "w": 0.9999999896438919
                }
            }
        } ,
        "data": [uint8]
    },
    "op": "publish"
}
```


B.5.7 *fireOutbreak()*

Fire outbreak - coordinates of the fire outbreak, which requires immediate attention of the CC and the firefighters.

robot_CC; **robot_firefighter**; **firefighter_CC**

Message to send location coordinates of a fire outbreak. This is a priority message and should be sent whenever a new fire outbreak is found, to allow the CC and firefighters to take adequate measures.

Ros Message

Ros type	Description
std_msgs/String	A string with the class of the agent (team, robot or firefighter).
std_msgs/Int8	An integer with the number id of the agent.
geometry_msgs/PoseStamped	A Pose with reference coordinates frame and time stamp.

Listing B.43: *ROS message: fireOutbreak()*

```
Header              header
std_msgs/String     agent_class
std_msgs/Int8       id
geometry_msgs/Pose  pose
```

Json Message

Listing B.44: *Json message: fireOutbreak()*


```
{
    "topic": "robot_1/fireOutbreak",
    "msg": {
        "agent_class" : "robot",
        "id" : 1,
```

```
        "header": {
            "stamp": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "frame_id": "map",
            "seq": 232
        },
        "info": {
            "map_load_time": {
                "secs": 1358783297,
                "nsecs": 415561250
            },
            "resolution": float64,
            "width": uint32,
            "height": uint32,
            "origin":{
                "position": {
                    "y": -0.0019882202614098787,
                    "x": 0.0017761230701580644,
                    "z": 0.0
                },
                "orientation": {
                    "y": 0.0,
                    "x": 0.0,
                    "z": -0.0001439173933738437,
                    "w": 0.9999999896438919
                }
            }
        } ,
        "data": [uint8]
    },
    "op": "publish"
}
```

B.6 Messages and HRI taxonomy

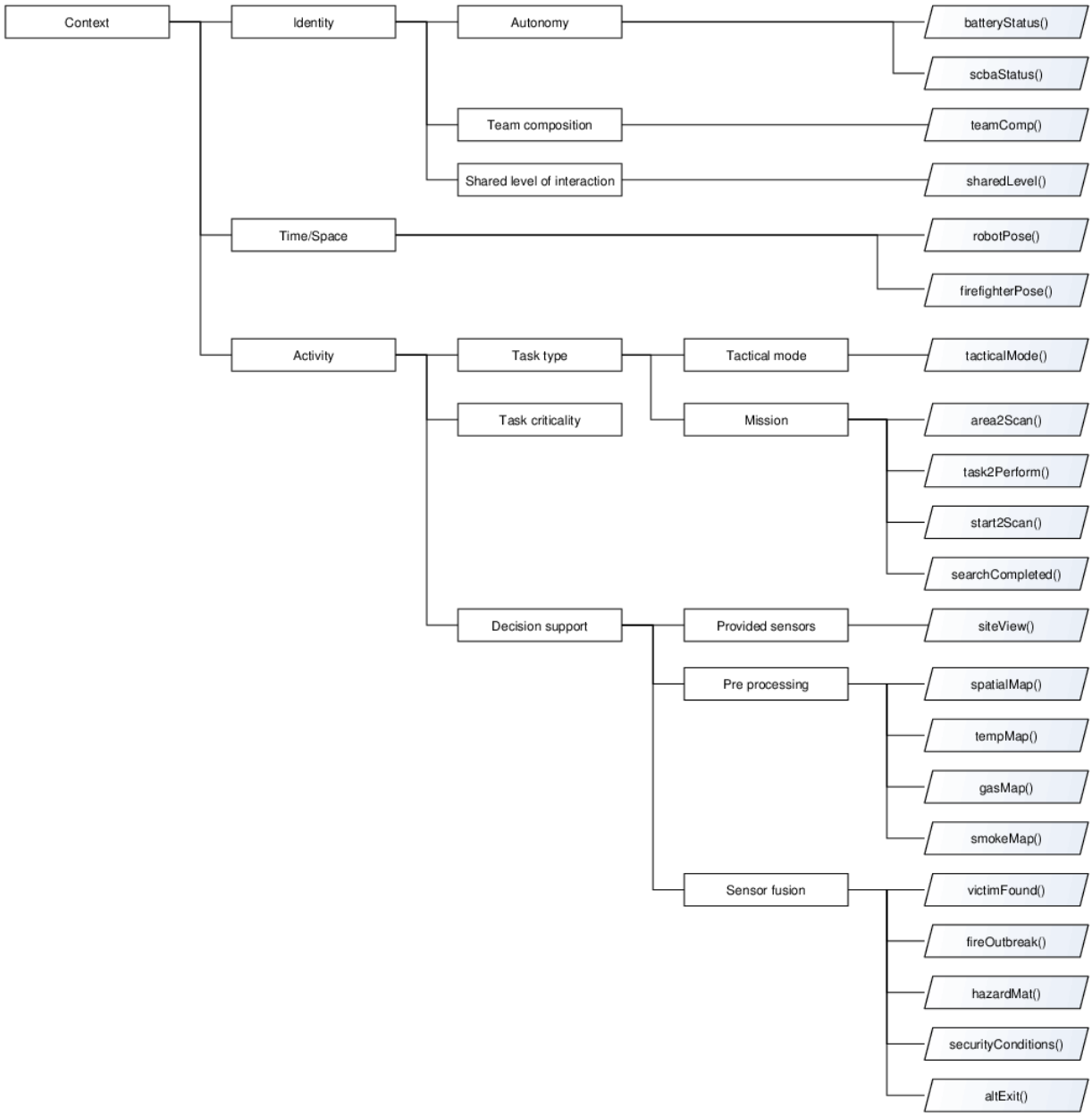


Figure B.1: Messages affecting the HRI taxonomy

C Entity-Relationship Diagram

C.1 Conceptual models

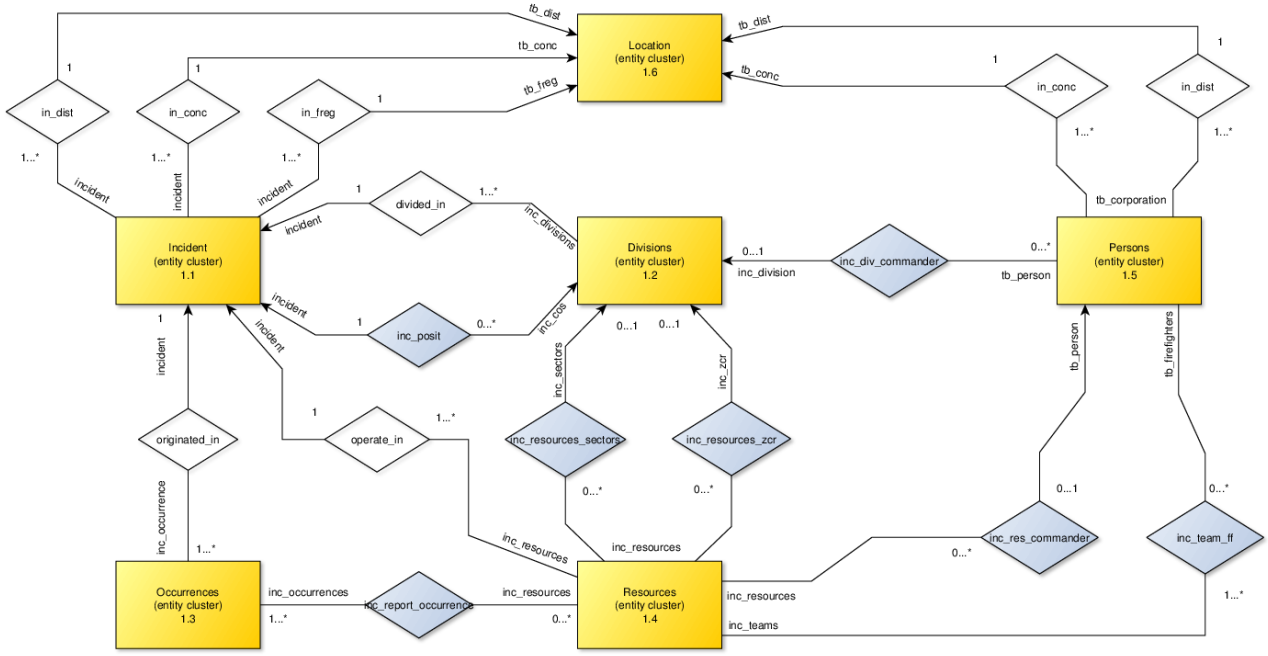


Figure C.1: Database conceptual schema - *root entity cluster*.

C.1.1 *Incident cluster*

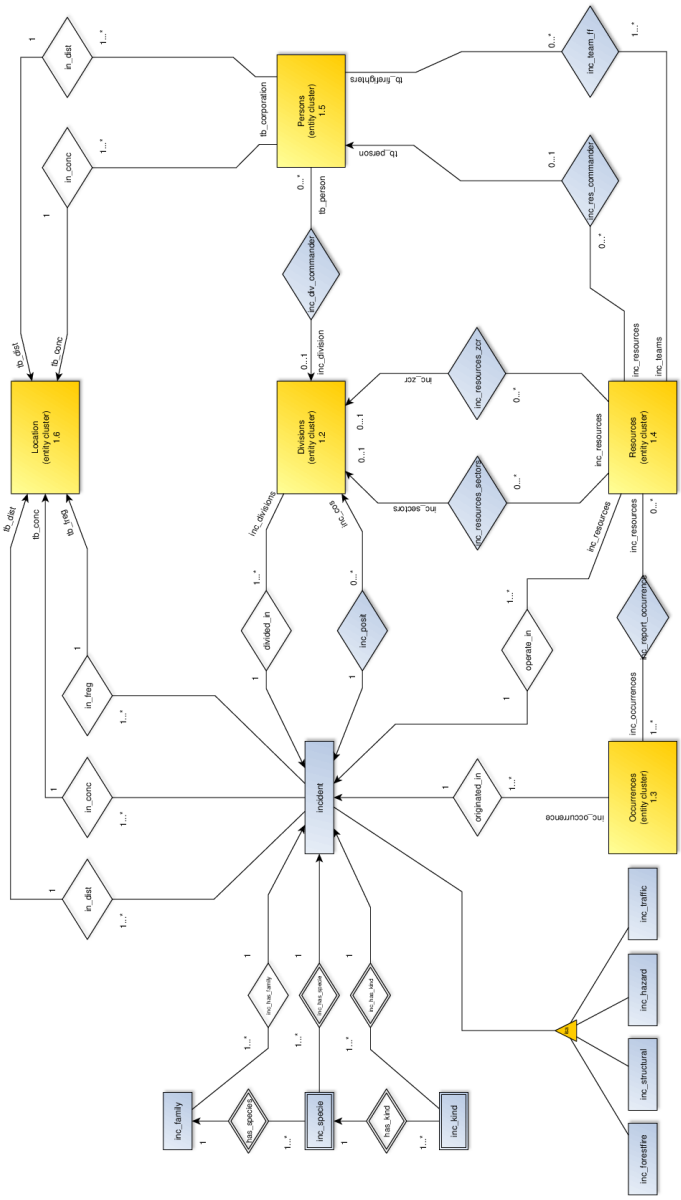


Figure C.2: Cluster 1.1 - Incident conceptual model

C.1.2 *Incident divisions cluster*

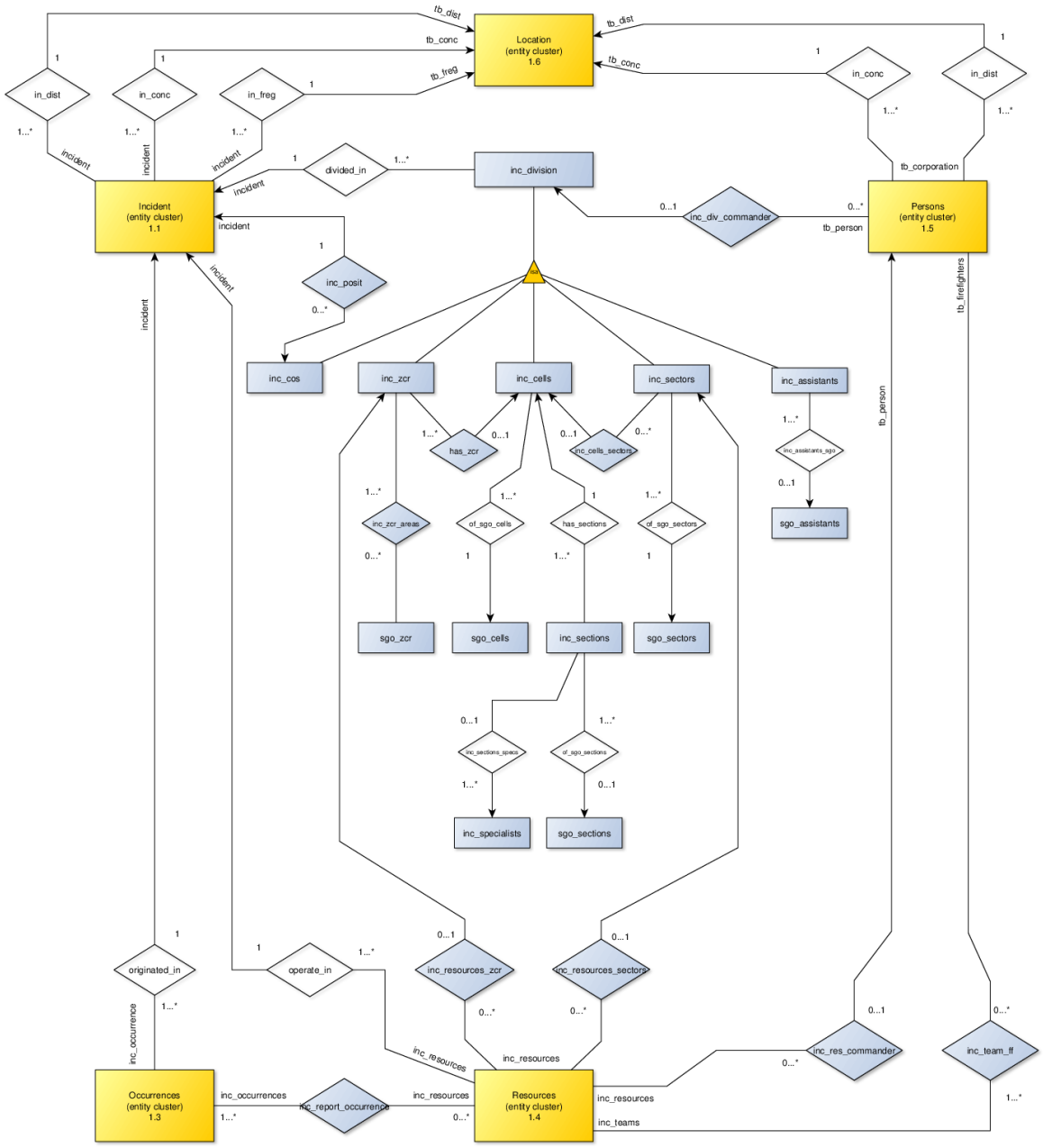


Figure C.3: Cluster 1.2 - Incident divisions conceptual model

C.1.3 *Incident occurrences cluster*

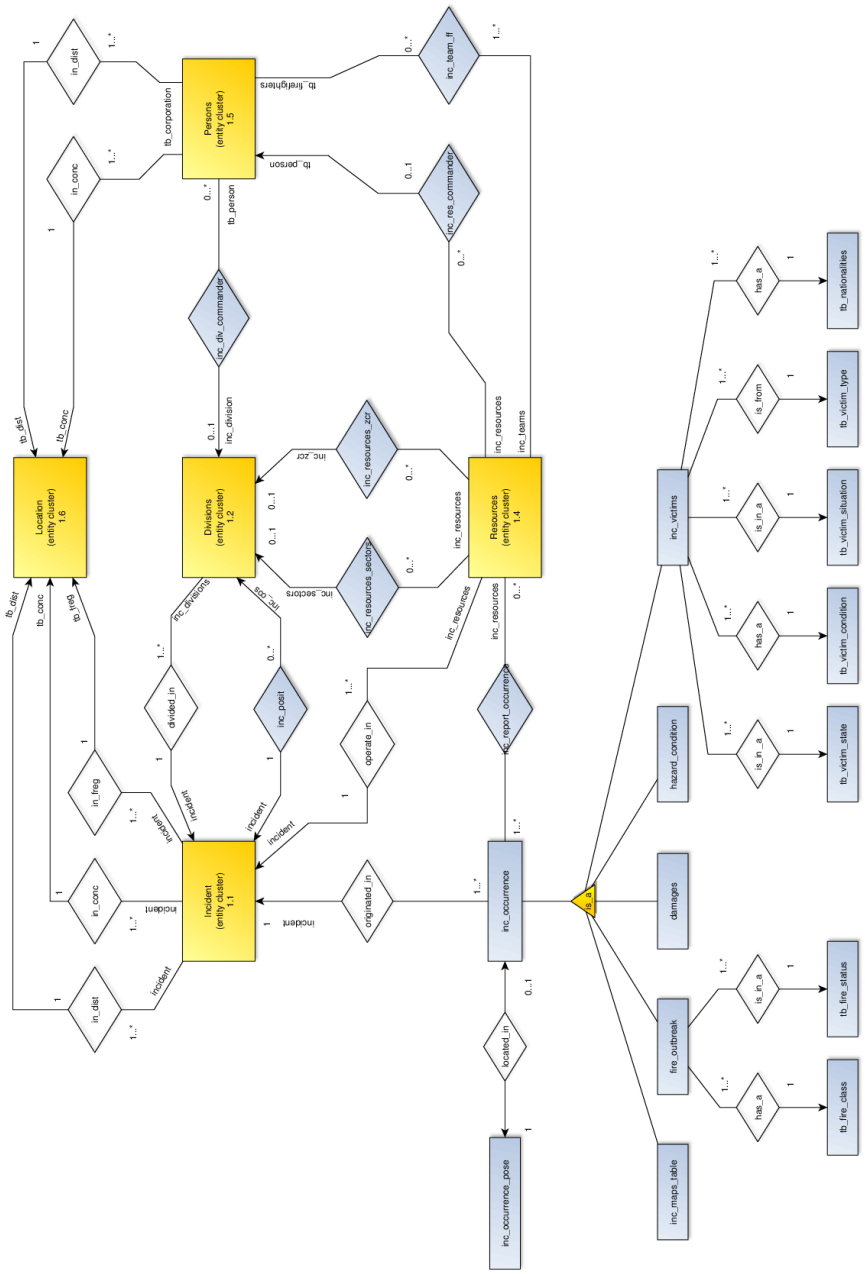


Figure C.4: Cluster 1.3 - Incident occurrences conceptual model

C.1.4 *Incident resources cluster*

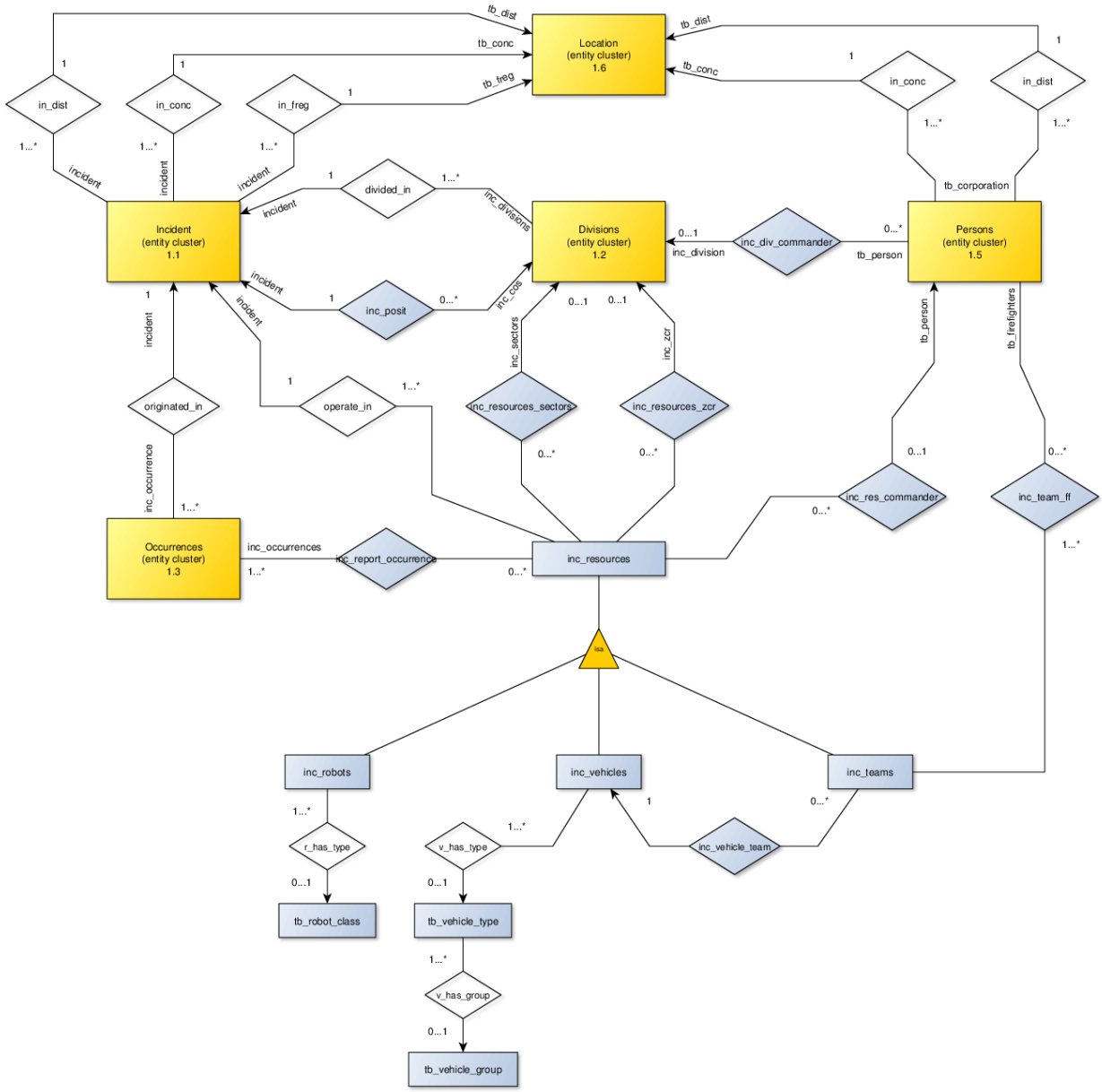


Figure C.5: Cluster 1.4 - Incident resources conceptual model

C.1.5 *Incident personnel cluster*

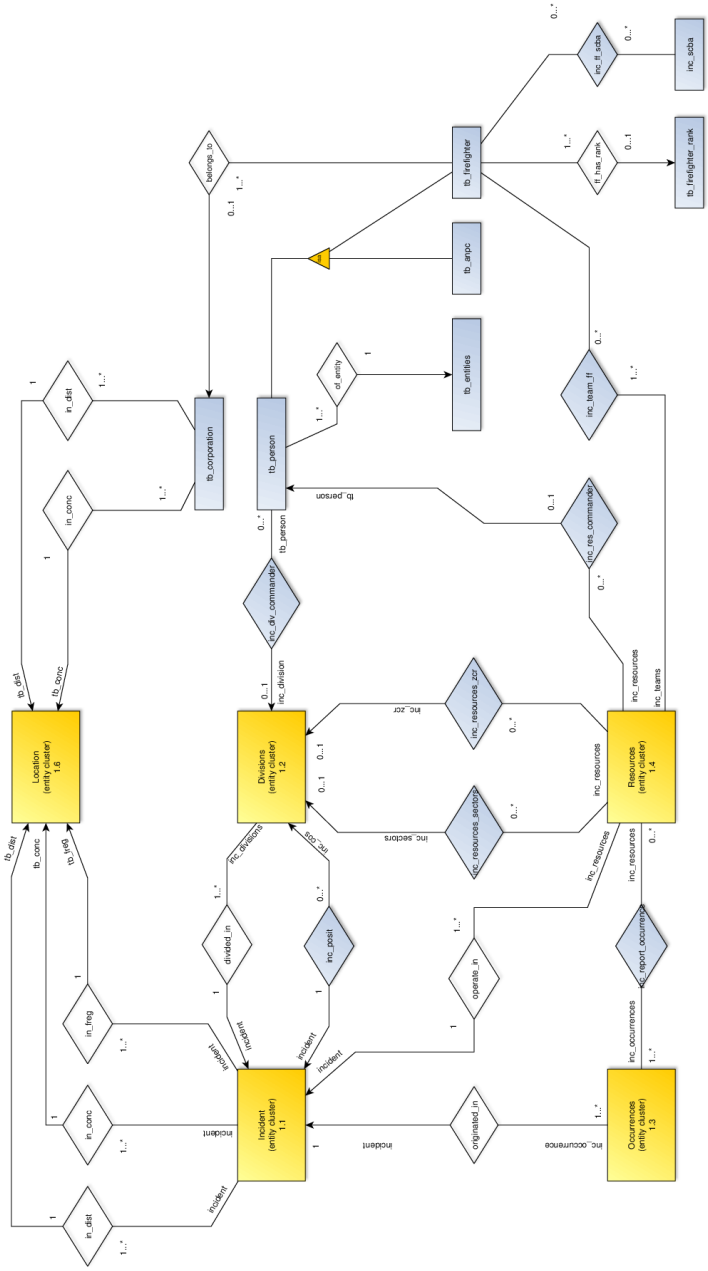


Figure C.6: Cluster 1.5 - Incident personnel conceptual model

C.1.6 *Incident locations cluster*

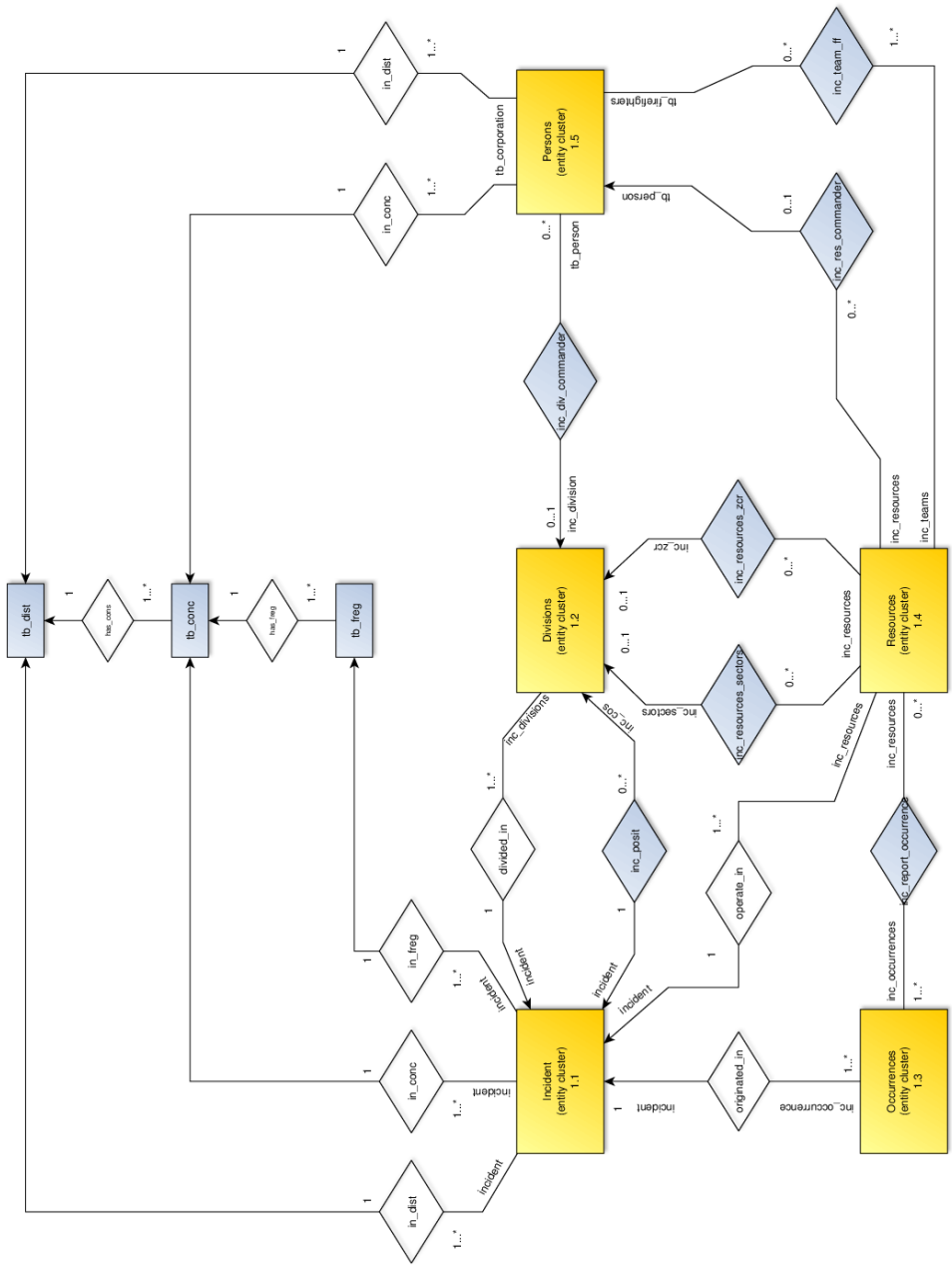


Figure C.7: Cluster 1.6 - Incident locations conceptual model

Bibliography

[1]

[2]

[AAM01] S. Alag, A. M. Agogino, and M. Morjaria. A methodology for intelligent sensor measurement, validation, fusion, and fault detection for equipment monitoring and diagnostics. *AI EDAM*, 15(4), 2001.

[AAMD11] H. Aliakbarpour, L. Almeida, P. Menezes, and J. Dias. Multi-sensor 3d volumetric reconstruction using cuda. *3D Research*, December 2011.

[ADB+99] Gregory D. Abowd, Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing*, HUC '99, pages 304–307, London, UK, UK, 1999. Springer-Verlag.

[AKP+11] A.SHARMA, R. KUMAR, P.VARADWAJ, A. AHMAD, and G. ASHRAF. A comparative study of support vector machine. In *Artificial Neural Network and Bayesian Classifierer for Mutagenicity Prediction*, 2011.

[App13] Apple. Event handling guide for ios. Technical report, Apple Inc, 2013.

[Ash07] Mark Ashdown. Asymmetric distributed collaboration in emergency response. Technical report, Marie Curie Outgoing International Fellowship, 2007.

[BCR09] Oliver Brdiczka, James L. Crowley, and Patrick Reignier. "learning situation models in a smart home". In *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 39, No. 1*, 2009.

[BDR07] M. Baldauf, S. Dustdar, , and F. Rosenberg. A survey on context-aware systems. In *International Journal of Ad Hoc and Ubiquitous Computing*, 2007.

[BLMC09] Oliver Brdiczka, Matthieu Langet, Jérôme Maisonnasse, and James L. Crowley. "detecting human behaviour models from multimodal observation in a smart home". In *IEEE Transactions on Automation Science and Engineering, Vol.6, No.4*, 2009.

[BLS08] Pierre Bessire, Christian Laugier, and Roland Siegwart. *Probabilistic Reasoning and Decision Making in Sensory-Motor Systems*. Springer Publishing Company, Incorporated, 1st edition, 2008.

[Bre05] B Brehmer. The dynamic ooda loop: Amalgamating boyd's ooda loop and the cybernetic approach to command and control. In *15 th International Command and Control Research and Technology Symposium (ICCRTS)*, 2005.

[Bur98] C. Burges. A tutorial on support vector machines for pattern recognitio. *Data Mining and Knowledge Discovery*, 2, June 1998.

[BVB08] Shishir B., G. Venayagamoorthy, and B.Paudel. Embedded neural network for fire classification using an array of gas sensors. In *SAS 2008 – IEEE Sensors Applications Symposium*, Atlanta, GA, February 2008.

[BW07] T. Born and A. Wright. Layered mode selection logic control with fuzzy sensor fusion network. In *Proc. SPIE 6561, Unmanned Systems Technology IX*, Orlando, May 2007.

[CA05] C. F. Castro and J. B. Antunes. *Combate a Incêndios Urbanos e Industriais*. , Escola Nacional de Bombeiros (Portuguese National School of Firefighters), Sintra, Portugal, 2005.

[CDB10] Francis Colas, Julien Diard, and Pierre Bessière. Common bayesian models for common cognitive issues. *Acta Biotheoretica*, 58(2-3):191–216, 2010.

[CDGL01] D. Calvanese, G. De Giacomo, and M. Lenzerini. Ontology of integration and integration of ontologies. *Proceedings of the 2001 International Description Logics Workshop (DL2001)*, 2001.

[CFA+09] Francesco Chiti, Romano Fantacci, Francesco Archetti, Enza Messina, and Daniele Toscani. "an integrated communications framework for context aware continuous monitoring with body sensor networks". In *IEEE Journal on Selected Areas in Communications*, 2009.

[CFLD12] M. S. Couceiro, C. M. Figueiredo, J. M. A. Luz, and M. J. Delorme. *Mathematical Modelling of Zombies*, chapter Zombie Infection Warning System Based on Fuzzy Decision-Making. University of Ottawa Press, 2012.

[CFM12] M. S. Couceiro, N. M. F. Ferreira, and J. A. T. Machado. Hybrid adaptive control of a dragonfly model. *Journal of Communications in Nonlinear Science and Numerical Simulation*, 17(2):893–903, 2012.

[CM03] J. Casper and R. Murphy. Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. *Systems, Man, and Cybernetics*, 33(3), 2003.

[CMRF12] M. S. Couceiro, J. A. T. Machado, R. P. Rocha, and N. M. F. Ferreira. A fuzzified systematic adjustment of the robotic darwinian pso. *Robotics and Autonomous Systems*, 60(12):1625 1639, 2012.

[CMS+09] Angelo Cangelosi, Giorgio Metta, Gerhard Sagerer, Stefano Nolfi, Chrystopher Nehaniv, Kerstin Fischer, Jun Tani, Tony Belpaeme, Giulio Sandini, Francesco Nori, Luciano Fadiga, Britta Wrede, Katharina, Rohlfing, Elio Tuci, Kerstin Dautenhahn, Joe Saunders, and Arne Zechel. "integration of action and language knowledge: A roadmap for developmental robotics". In *IEEE Transactions on Autonomous Mental Development*, 2009.

[CPR13] M. S. Couceiro, D. Portugal, and R. P. Rocha. A collective robotic architecture in search and rescue scenarios. In *Proceedings of the 28th Symposium On Applied Computing, SAC2013*, Coimbra, Portugal, March 2013.

[Cro06] Douglas Crockford. The fat-free alternative to xml, 2006.

[Dey01] Anind K Dey. Understanding and Using Context. *Personal and Ubiquitous Computing*, 5(1):4–7, 2001.

[DHYS04] Jill L. Drury, Dan Hestand, Holly A. Yanco, and Jean Scholtz. Design guidelines for improved human-robot interaction. In Elizabeth Dykstra-Erickson and Manfred Tscheligi, editors, *CHI Extended Abstracts*, page 1540. ACM, 2004.

[dREdS08] Ricardo Couto A. da Rocha, Markus Endler, and Thiago Senador de Siqueira. "middleware for ubiquitous context-awareness". In *ACM MPAC'08*, 2008.

[DV96] J. Avaritsiotis D. Vlachos. Fuzzy neural networks for gas sensing. *Sensors and Actuators*, B(33), 1996.

[DYL09] SHI Dongcheng, XING Yashu, and ZHU Long. "moving object detection based on scene understanding". In ., 2009.

[EBJ03] M.R. Endsley, B. Bolte, and D.G. Jones. *Designing for Situation Awareness: An Approach to User-Centered Design*. Taylor & Francis, 2003.

[EGR91] C. A. Ellis, S. J. Gibbs, and G.L. Rein. Groupware: Some issues and experiences. In *COMMUNICATIONS OF THE ACM*, pages 39–58, 1991.

[Elm02] Wilfried Elmenreich. An introduction to sensor fusion. Technical report, Vienna University of Technology, 2002.

[eS05] Eduardo Schnell e Schühli. Reconhecimento de gestos de maestro utilizando redes neuronais artificiais parcialmente recorrentes. Master's thesis, Universidade Federal do Paraná, 2005.

[FBMB02] J. Fernández-Breis and R. Martínez-Béjar. A cooperative framework for integrating ontologies. *International Journal of Human-Computer Studies*, 2002.

[Fer94] E. C. Ferstl. Context effects in syntactic ambiguity resolution: the location of presuppositional phrase attachment. In *16th Annual Conference of the Cognitive Science Society*, 1994.

[FLD10] J. F. Ferreira, J. Lobo, and J. Dias. Real-time perception algorithms on gpu: Real-time implementation of bayesian models for multimodal perception using cuda. In *Journal of Real-Time Image Processing, Springer Berlin/Heidelberg*, 2010.

[GFG12] Hans-Eric Grönlund, Colin Francis, and Shawn Grimes. *iOS 6 Recipes: A Problem-Solution Approach*. Apress, 2012.

[GH86] A. N. Glass and K. J. Holyoak. *Cognition*. New York: Random House, 2nd. edition, 1986.

[GHC09] Patrick Grommes, Rob Hutton, and Nick Colford. Comprehensive model of first responder operations & concept of operations. Technical report, COPE, 2009.

[Gop98] Sucharita Gopal. Artificial neural networks for spatial data analysis, ncgia core curriculum in giscience, December 1998.

[GRB09] Baptiste Gaultier, Rayene Ben Rayana, and Jean-Marie Bonnin. "context management systems applied to mobility". In *IEEE*, 2009.

[Gri09] Paul Grimwood. Tactical deployment into fire involved structures, 2009.

[GZS+09] Denis Garagic, Majid Zandipour, Frank Stolle, Matthew Antone, and Bradley J. Rhodes. "hybrid neuro- bayesian spatial contextual reasoning for scene content understanding". In *in Proceedings of the 12th International Conference on Information Fusion*, 2009.

[Har11] Bruce Hartpence. *Packet Guide to Core Network Protocols*. O'Reilly Media, Inc., 2011.

[HHH97] Frank G. Hofmann, Peter Heyer, and Günter Hommel. Velocity profile based recognition of dynamic gestures with discrete hidden markov models. *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, pages 81–95, 1997.

[HL97] D.L. Hall and J. Llinas. An introduction to multisensor data fusion. 85(1), 1997.

[HM11] Gjoreski Hristijan and Gams Matjaz. Accelerometer data preparation for activity recognition. 2011.

[Hol94] T. Holtgraves. Communication in context: effects of speaker status on the comprehension of indirect requests. *Journal of Experimental Psychology*, 20:1205–1218, 1994.

[JM09] Agnès Just and Sébastien Marcel. A comparative study of two state-of-the-art sequence processing techniques for hand gesture recognition. *Computer Vision and Image Understanding*, 113:532–543, 2009.

[Jor99] Michael I. Jordan, editor. *Learning in graphical models*. MIT Press, Cambridge, MA, USA, 1999.

[JR08] J.-M. Ahuactzin J. Rett, J. Dias. *Brain, Vision and AI*, chapter Laban Movement Analysis using a Bayesian model and perspective projections. INTECH, August 2008.

[KAQ+10] Kamrad Khoshhal, Hadi Aliakbarpour, Joao Quintas, Paulo Drews, and Jorge Dias. Probabilistic lma-based classification of human behaviour understanding using power spectrum technique. In *13th International Conference on Information Fusion*, 2010.

[KD12] Kamrad Khoshhal and Jorge Dias. Probabilistic human interaction understanding - exploring relationship between human body motion and the environmental context. *Pattern Recognition Letters*, Special Issue on Scene Understandings and Behaviours Analysis:In press, 2012.

[KK13] Shanker Keshavdas and Geert-jan M Kruijff. Functional Mapping for Human-Robot Collaborative Exploration. In *Proceedings of the 2013 IASTED Symposium on Artificial Intelligence And Applications*, 2013.

[KKM+05] Juha Kela, Panu Korpipää, Jani Mäntyrvi, Sanna Kallio, Giuseppe Savino, Luca Jozzo, and Sergio Di Marca. Accelerometer-based gesture control for a design environment. *Personal and Ubiquitous Computing*, 10:285–299, 2005.

[Kli09] Marco Klingmann. Accelerometer-based gesture recognition with the iphone. Master's thesis, Goldsmiths University of London, 2009.

[KSAD12] Kamrad Khoshhal, Luis Santos, Hadi Aliakbarpour, and Jorge Dias. Parameterizing interpersonal behaviour with laban movement analysis - a bayesian approach. In *3rd International Workshop on Socially Intelligent Surveillance and Monitoring (SISM2012) in CVPR2012*, 2012.

[KSD01] A. Kiryakov, K. Simov, and M. Dimitrov. Ontomap: portal for upper-level ontologies. *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS'01)*, 2001.

[KWM10] Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. Activity recognition using cell phone accelerometers. 2010.

[L.A65] L.A.Zadeh. Fuzzy sets*. *Information and Control*, 8, 1965.

[LCP+] Miguel A. Luz, M. Couceiro, D. Portugal, Rui Rocha, H. Araujo, and G. Dias. Comparison of classification methods for golf 3 putting performance analysis. Coimbra, Portugal.

[Lee13] David A. Lee. Fat markup: Trimming the myth one calorie at a time. In *In Proceedings of Balisage: The Markup Conference 2013*, 2013.

[LFD09] J. Lobo, J. F. Ferreira, and J. Dias. Robotic implementation of biological bayesian models towards visuo-inertial image stabilization and gaze control. In *In Proc. of IEEE International Conference on Robotics and Biomimetics (ROBIO 2008)*, 2009.

[LRT+] Ming Li, Viktor Rozgica, Gautam Thatte, Sangwon Lee, Adar Emken, Murali Annavaram, Urbashi Mitra, Donna Spruijt-Metz, and Shrikanth Narayanan.

[LW09] Michael Lewis and Huadong Wang. Using humans as sensors in robotic search, 2009.

[MB05] R. Murphy and J. Burke. Up from the rubble: Lessons learned about hri from search and rescue. In *In Proc. of the Human Factors and Ergonomics Society 49th Annual Meeting*, 2005.

[MBDH02] J. Madhavan, PA. Bernstein, P. Domingos, and A. Halevy. Representing and reasoning about mappings between domain models. *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI'02)*, 2002.

[MC96] B. A. Mellers and A. D. J. Cooke. The role of task and context in preference measurements. *Psychological Science*, 1996.

[Mil56] G. A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. pages 81–97, 1956.

[MLD11] Paulo Menezes, Frédéric Lerasle, and Jorge Dias. Towards human motion capture from a camera mounted on a mobile robot. *Image and Vision Computing*, 29(6):382–393, 2011.

[MLKK06] Aekyung Moon, Kangwoo Lee, Hyoungsun Kim, and Hyun Kim. "designing camus based context-awareness for pervasive home environments". In *IEEE International Conference on Hybrid Information Technology (ICHIT)*, 2006.

[MMSV00] A. Martinelli, F. Martinelli, S.Nicosia, and P. Valigi. Multisensor fusion for mobile robot positioning and navigatio. In *Preprints of 6th IFAC Symposium on Robot Control*, 2000.

[MP01] V. Mihajlovic and M. Petkovic. Dynamic bayesian networks: A state of the art, 2001.

[MP07] Christopher A. Miller and Raja Parasuraman. Designing for flexible interaction between humans and automation: Delegation interfaces for supervisory control. *Human Factors*, 49:57–75, 2007.

[MS00] A. Maedche and S. Staab. Semi-automatic engineering of ontologies from texts. *Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering (SEKE 2000)*, 2000.

[MWH04] Bruce A. Maxwell, Nicolas Ward, and Frederick Heckel. Game-based design of human-robot interfaces for urban search and rescue. In *In Computer-Human Interface Fringe*, 2004.

[Nah12] Vandad Nahavandipoor. *iOS 6 Programming Cookbook*. O'Reilly Media, Inc., 2012.

[NH09] Gerrit Niezen and Gerhard P. Hancke. Evaluating and optimising accelerometer-based gesture recognition techniques for mobile devices. *IEEE AFRICON 2009*, 2009.

[Nic03] S. P. Niculescu. Artificial neural networks and genetic algorithms in qsa. *Journal of Molecular Structure: THEOCHEM*, 622(1-2), March 2003.

[NOP09] Sistema de gestão de operações - simbologia, 2009.

[NR03] Peter Norvig and Stuart Russell. *Artificial Intelligence: A Modern Approach*. Prentice Hall Series In Artificial Intelligence. Prentice Hall, 2003.

[Pas98a] Jason Pascoe. Adding generic contextual capabilities to wearable computers. *Wearable Computers, 1998. Digest of Papers. . . .*, 44(0):92–99, 1998.

[Pas98b] Mr. Jason Pascoe. Adding generic contextual capabilities to wearable computers. In *Proceedings of the 2Nd IEEE International Symposium on Wearable Computers*, ISWC '98, pages 92–, Washington, DC, USA, 1998. IEEE Computer Society.

[PBHD06] A. G. Amitha Perera, Glen Brooksby, Anthony Hoogs, and Gianfranco Doretto. "moving object segmentation using scene understanding". In *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop*, 2006.

[PBL+09] Pravin Pawar, Hanga Boros, Fei Liu, Geert Heijenk, and Bert-Jan van Beijnum. "bridging context management systems in the ad hoc and mobile environments". In *IEEE*, 2009.

[PEa+06] Juha Pärkkä, Miikka Ermes, Panu Korpipã a, Jani M antyj arvi, Johannes Peltola, and Ilkka Korhonen. Activity classification using realístic data from wearable sensors. In *Cornell University*. Press, 2006.

[Pea07] Major Brent A. Peacock. Connecting the edge: Mobile ad-hoc networks (manets) for network centric warfare, 2007.

[PEK+06] Juha Pärkkä, Miika Ermes, Panu Korpipää, Jani Mäntyjärvi, Johannes Peltola, and Ilkka Korhonen. "activity classification using realistic data from wearable sensors". In *IEEE Transactions on Information Technology in Biomedicine*, 2006.

[PG94] R. Pevtzow and R. L. Goldstone. Categorization and the parsing of objects. In *16th Annual Conference of the Cognitive Science Society*, 1994.

[Ph.10] Hong Ki Min Ph.D. "context agent for smart home resident". In *IEEE*, 2010.

[PNAL11] C. Premebida, U. Nunes, R. Araujo, and O. Ludwig. Evaluation of boosting-svm and srm-svm cascade classifiers in laser and vision-based pedestrian detection. In *14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Washington DC, USA, October 2011.

[PR12] D. Portugal and R. P. Rocha. Decision methods for distributed multi-robot patrol. In *Proceedings of the 2012 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR'2012)*, College Station, Texas, USA, November 2012.

[Pre08] Z Prekopcsák. Accelerometer based real-time gesture recognition. pages 1–5, 2008.

[Rab89] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *IEEE*, 1989.

[Ras93] J. Rasmussen. Deciding and doing: Decision making in natural contexts. In G.A. Klein, J. Orasanu, and R. Calderwood, editors, *Decision Making in Action: Models and Methods*, pages 158–171. Ablex Publishing Corporation, 1993.

[Rho07] Bradley J. Rhodes. "taxonomic knowledge structure discovery from imagery-based data using the neural associative incremental learning (nail) algorithm". In *Elsevier, Information Fusion 8*, 2007.

[RNO+08] R.Araújo, U. Nunes, L. Oliveira, P. Sousa, and P. Peixoto. Support vector machines and features for environment perception in mobile robotics. Coimbra, Portugal, 2008.

[SAW94a] B Schilit, N Adams, and R Want. Context-aware computing applications. In *Workshop on Mobile Computing Systems and Applications*, number 4 in Proceedings. Workshop on Mobile Computing Systems and Applications (Cat. No.94TH06734), pages 85–90. IEEE Comput. Soc. Press, 1994.

[SAW94b] Bill N. Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *IN PROCEEDINGS OF THE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS*, pages 85–90. IEEE Computer Society, 1994.

[SLC11] Annapurna Sharma, Young-Dong Lee, and Wan-Young Chung. High accuracy human activity monitoring using neural network. *CoRR*, abs/1107.4429, 2011.

[SM05] Craig Schlenoff and Elena Messina. A Robot Ontology for Urban Search and Rescue. In *Proceedings of the 2005 ACM workshop on Research in knowledge representation for autonomous systems KRAS 05 (2005)*, 2005.

[SMC09] Sudeep Sundaram and Walterio Mayol-Cuevas. High level activity recognition using low resolution wearable vision. In *First Workshop on Egocentric*

Vision, in conjunction with the International Conference on Computer Vision and Pattern Recognition. IEEE, June 2009.

[SOG06] Extreme fire behavior, 2006.

[SPHB08] Thomas Schlömer, Benjamin Poppinga, Niels Henze, and Susanne Boll. Gesture recognition with a wii controller. *Proceedings of the Second International Conference on Tangible and Embedded Interaction (TEI'08), Bonn, Germany*, 2008.

[SR96] Mike Scaife and Yvonne Rogers. External cognition: How do graphical representations work? *Int. J. Hum.-Comput. Stud.*, 45(2):185–213, August 1996.

[S.S10] S.Sharma. Implementation of artificial neural network for odor identification using e-nose. In *NCCI 2010 -National Conference on Computational Instrumentation CSIO*, Chandigarh, INDIA, March 2010.

[SST] Steven A. Shafer, Anthony Stentz, and Charles E. Thorpe. An architecture for sensor fusion in a mobile robot. Technical Report Paper 585, CMU Robotics Institute.

[Ste] C. Stergiou. What is a neural network?

[SVS+10] P. Scerri, P. Velagapudi, K. Sycara, H. Wang, S.-Y. J. Chien, and M. Lewis. Towards an understanding of the impact of autonomous path planning on victim search in usar. In *In Proc. of 2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2010)*, 2010.

[SZL+10] Lin Sun, Daqing Zhang, Bin Li, Bin Guo, and Shijian Li. Activity recognition on an accelerometer embedded mobile phone with varying positions and orientations. 2010.

[Ten11] M. Tenorth. *Knowledge Processing for Autonomous Robots*. PhD thesis, Technische Universität München, 2011.

[Tie08] De-Kun Hu ; Hui Peng ; Ju-Hong Tie. A multi-sensor information fusion algorithm based on svm. In *Apperceiving Computing and Intelligence Analysis*, December 2008.

[TKJ+10] Moritz Tenorth, Lars Kunze, Dominik Jain, , and Michael Beetz. Knowrob-map – knowledge linked semantic object maps. *Proceedings of 2010 IEEE-RAS International Conference on Humanoid Robots*, 2010.

[Tur98] Roy M. Turner. Context-mediated behavior for intelligent agents. *International Journal of Human-Computer Studies*, 48(3):307–330, March 1998.

[vdVNHF] Bram J. J. van der Vlist, Gerrit Niezen, Jun Hu, and Loe M.G. Feijs. "semantic connections: Exploring and manipulating connections in smart spaces". In ., .

[w3ca] w3c. w3cOWL. `http://www.w3.org/TR/owl-features/`.

[w3cb] w3c. w3cRDF. `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/$\delimiter"026E30F$#section-Concepts`.

[w3cc] w3c. w3cSWRL. `http://www.w3.org/Submission/SWRL/`.

[Wei09] Yu Weihong. Research on Maritime Search and Rescue Decision-Making Ontology Model. In *. . . , 2009. ESIAT 2009. International Conference on*, pages 140–142, 2009.

[wika] wikipedia. wikiOWL. `http://en.wikipedia.org/wiki/Web$\delimiter"026E30F$_Ontology$\delimiter"026E30F$_Language`.

[wikb] wikipedia. wikiRDF. `http://en.wikipedia.org/wiki/Resource$\delimiter"026E30F$_Description$\delimiter"026E30F$_Framework`.

[wikc] wikipedia. wikiSWRL. `http://en.wikipedia.org/wiki/Semantic$\delimiter"026E30F$_Web$\delimiter"026E30F$_Rule$\delimiter"026E30F$_Language`.

[WJLSK12] P. Wang, W. Jiang, X. Li, and J.i Xin S. Kang. Research for multi-sensor information fusion algorithm of search and rescue robot based on embedded control network. *JOURNAL OF COMPUTERS*, 7(5), May 2012.

[YA99] Ming-Hsuan Yang and Narendra Ahuja. Recognizing hand gesture using motion trajectories. 1999.

[YD04] Holly A. Yanco and Jill L. Drury. Classifying human-robot interaction: an updated taxonomy. In *SMC (3)*, pages 2841–2846. IEEE, 2004.

[ZL09] Y. Ma Z. LI. A new method of multi-sensor information fusion based on svm. In *Proceedings of the Eighth International Conference on Machine Learning and Cybernetics*, Baoding, July 2009.

[ZLH08] X. Zhao, Q. Luo, and B. Han. Survey on robot multi-sensor information fusion technology. In *7th World Congress on Intelligent Control and Automation*, June 2008.

[ZRB08] Majid Zandipour, Bradley J. Rhodes, , and Neil A. Bomberger. "coalesce: A probabilistic ontology-based scene understanding approach". In *in Proceedings of the 11th International Conference on Information Fusion*, 2008.