Chapter 2
Testbeds for Cooperating Objects

2.1 Introduction

In the last years the interest in Cooperating Objects technologies has originated a growing demand of tools for testing and validating algorithms and methods. Lately, the development of testbeds for technologies such as multi-robot systems and ubiquitous and pervasive devices has been intensified both in number and variety.

This chapter briefly presents the existing testbeds for Cooperating Objects. Cooperation between heterogenous devices and technologies is a key issue in the Cooperating Objects domain. Thus, we classify testbeds attending to the level of integration between different technological fields. Other features such as heterogeneity, flexibility, modularity and ease of use are also analyzed. The selection of the software architecture and middleware is critical for the modularity, flexibility and interoperability of the testbed. This chapter also reviews the main existing middleware for integration of Cooperating Objects.

This chapter is structured as follows. Section 2.2 briefly analyzes Cooperating Objects testbeds attending to their level of integration between elements from different technological fields. Section 2.3 reviews the main existing middleware for Cooperating Objects. Existing Cooperating Objects testbeds with no integration are briefly summarized in Sect. 2.4. Testbeds with low level of integration are analyzed in Sect. 2.5. Testbeds for Cooperating Objects with high level of integration are reviewed in Sect. 2.6. Finally, Sect. 2.7 concludes the chapter.

2.2 Components of a Cooperating Objects Testbed

The testbeds analyzed in this chapter integrate elements from different Cooperating Objects technological fields ranging from robotic systems to ubiquitous systems such as Wireless Sensor Networks (WSN), camera networks and networks for smartphones, PDAs and tablets, among others.

J. R. Martinez-de Dios et al., *A Remote Integrated Testbed for Cooperating Objects*,
SpringerBriefs in Cooperating Objects,
DOI: 10.1007/978-3-319-01372-5_2, © The Author(s) 2014

Wireless Sensor Networks are comprised of a set of embedded nodes with sensing, processing and communication capabilities. WSN nodes can be integrated with a wide variety of sensors such as temperature, humidity, light intensity sensors and accelerometers, GPS receivers and even small cameras such as Cyclops and CMUcam3. WSN radio circuitries also allow to measure the strength of the radio signal of the received messages (RSSI) and other Link Quality Estimators. WSN nodes used in testbeds are frequently static, although mobile nodes may exist if carried by people or mounted on mobile robots. The design of networking and routing algorithms supporting mobility is one important open research area in WSN. Other components such as camera networks, PDAs and smartphones can also be frequently found integrated in these testbeds.

A wide variety of robotic platforms have been integrated in testbeds. Ground robots are usually able to carry heavier payloads, and tend to have larger on-board sensing and processing capabilities. On-board sensors include cameras—from regular color cameras to RGB-D cameras, laser range finders, ultrasound sensors, GPS receivers and Inertial Navigation System units, among others. Aerial robots, although limited in payload, can move in 3D. Quad-rotors, with vertical take-off and landing capabilities, are the most commonly used platforms in testbeds, although blimps, fixed-wing platforms and helicopters are also found mainly in outdoors testbeds. Underwater or surface marine vehicles are rarely used in multi-vehicle testbeds although there are some exceptions.

A rich variety of sensors can be integrated in these aforementioned platforms. They have high degree of heterogeneity. While low cost, low size and low energy constrain the characteristics of WSN sensors, robots can carry and provide mobility to sensors with higher performance. Testbed communications typically include separate wireless networks for the multi-robot system and for the WSN. They differ greatly in range, bandwidth, quality of service and energy consumption. WSN networks were designed for low-rate and low-range communications whereas Wi-Fi networks, typically used by robotic systems, can provide significantly higher rates at greater distances.

A testbed for Cooperating Objects should share design space with Cooperating Objects themselves. We have already identified several Cooperating Objects characteristics, which are depicted in more detail in [1]. More specifically, when designing Cooperating Objects testbeds we have to deal with issues such as interoperability and level of integration among heterogeneous technologies, flexibility, modularity, extensibility, scalability and usability. Refer to [1] for detailed definitions.

A high percentage of the testbed interoperability, flexibility, modularity, extensibility and scalability depends on its software architecture and middleware. Next section briefly presents a review of the main existing middleware for Cooperating Objects.

The level of integration between elements from different technological fields is one of the main characteristics of Cooperating Objects testbeds. An ideal testbed should allow high degree of interoperability. This interoperability is not easy due to the high diversity in their sensing, actuation, computational and communication capabilities. Nevertheless, heterogeneity is in many cases the origin of interesting

synergies. Thus, according to the level of integration testbeds can be classified in *testbeds with no integration*, *testbeds with partial integration* and *full Cooperating Objects testbeds*.

2.3 Middleware for Cooperating Objects

The main functions of a middleware for Cooperating Objects can be enumerated as follows:

- support communication among heterogeneous components,
- enable interoperability and integration mechanisms among objects,
- offer often-needed system services and functionalities,
- simplify the development process.

A high number of middleware for robotic and ubiquitous systems have been developed. The higher complexity of robotic technologies suggests adopting a robotic middleware extended with capabilities to integrate WSN and other ubiquitous systems. Below is a brief review of the main existing middleware for networked robotics systems.

Robot Operating System [ROS]

ROS[1] is an open-source software framework for robot software development that provides functionalities that one could expect from an operating system. ROS includes functions for hardware abstraction, low-level device control, implementation of commonly-used functionalities, message-passing between processes and package management, among others. ROS is based on a graph architecture where processing takes place in nodes that may receive, post and multiplex sensor, control, state, planning, actuator and other messages. It includes a wide range of functionalities to help development such as simultaneous localization and mapping (SLAM), planning, image processing and perception, simulation and logging, among others. ROS also includes a set of simulators, such as Stage [2] and Gazebo [3]. ROS has been recently accepted as a "*de facto*" standard by the robotics community enabling code reuse and sharing.

Player/Stage

Player/Stage[2] provides a server-client infrastructure with hardware drivers and basic algorithms for mobile robotic [4]. Player abstracts hardware heterogeneity by using standard interfaces to communicate low-level server drivers and high-level client programs. The server drivers interact with the actual sensors and actuators. Various client-side libraries exist in the form of proxy objects for different programming languages (C, C++, Java, Matlab and Python, among others) to access the

[1] http://www.ros.org

[2] http://playerstage.sourceforge.net

services provided by the Player platform. Clients can connect to Player servers to access data, send commands or request configuration changes to an existing device.

Its modular architecture makes it flexible to support new hardware. Player is independent of platforms, programming languages and transport protocols. It includes a ready-to-use set of drivers for most commercial platforms and sensors, a set of algorithms for perception and robot navigation and a number GUIs, logging and visualization tools. It is also compatible with Stage and Gazebo.

YARP

YARP is an open-source set of libraries, protocols for development of multi-robot applications [5]. It provides support to the robot control system as a collection of programs communicating in a peer-to-peer way, with a family of connection types. It includes a number of tools and functionalities aiming to keep modules and devices cleanly decoupled in a modular architecture. YARP also supports flexible interfacing with hardware devices. Written in C++, YARP is OS neutral and provides support for common devices used in robotics such as framegrabbers, digital cameras, motor control boards, among others.

Miro

Miro [6] is an object-oriented middleware developed at University of Ulm. It uses the common object request broker architecture (CORBA) standard. This allows inter-process and cross-platform interoperability for distributed robot control, using multi-platform libraries for easy portability. Examples of these libraries are the CORBA based Adaptive Communication Environment (ACE) [7] or the CORBA Notification Services [8].

Miro allows the implementation of multi-tier architectures. Its provides object-oriented interface abstractions for many sensors and actuators. It also provides a number of often-needed services such as robot mapping, self localization, behavior generation, path planning, logging and visualization facilities. The layered architecture and object-oriented approach make Miro significantly flexible and extendible [9].

Orocos

The Orocos 2.0 toolchain [10] is used to create real-time robotics applications using modular, run-time configurable software components. It provides multi-platform support, extensions to other robotics frameworks such as ROS and YARP, code generators to transfer user-defined data between distributed components, run-time and real-time configurable and scriptable components and event data logging and reporting, among others.

Orca

Orca[3] is a component-based branch of Orocos developed at KTH. It is an open-source framework for developing component-based robotic systems that provides the means to define and develop building-blocks that can be pieced together to form from single vehicles to distributed sensor networks [11]. It enables software reuse

[3] http://orca-robotics.sourceforge.net

by defining a set of commonly-used interfaces, providing libraries with a high-level application programming interface (API) and maintaining a repository of components. Orca adopts a Component-Based Software Engineering approach without applying any additional architectural constraints. It uses a commercial open-source library for communication and interface definition and provides tools to simplify component development and uses cross-platform development tools.

Microsoft Robotics Developer Studio

Microsoft RDS[4] is a Windows-based environment for academic and commercial developers to help in the development of robotic applications. It includes a lightweight asynchronous service-oriented runtime and a set of visual authoring and simulation tools, as well as templates, tutorials and sample code. It includes a development environment with a wide range of support to ease the development of robot applications. Reference Platform design defines a minimum set of components and associated software services that will allow a user to begin using RDS with relatively little effort.

Carnegie Mellon Robot Navigation Toolkit [CARMEN]

CARMEN[5] is an open-source modular software designed to provide basic navigation primitives including: base and sensor control, logging, obstacle avoidance, localization, path planning, and mapping for mobile robots [12]. CARMEN provides robot hardware support for some platforms and sensors. It also includes a 2D robot/sensor simulator, process monitoring, message logging and playback functionalities. Communications between CARMEN programs are handled using a separate package called IPC. CARMEN runs under Linux. It is written in C and also provides support for Java.

Robot Software Communication Architecture [RSCA]

RSCA is a middleware for networked robots developed by Seoul National University [13]. It provides a standard operating environment and development framework for robot applications and includes mechanisms to ensure Quality of Service. It consists of a Real-Time Operating System compliant with PSE52 in IEEE POSIX.13 and a communication middleware compliant with CORBA and RT-CORBA v.1.1 [14]. RSCA provides an abstraction layer that makes robot applications portable and reusable on different hardware.

Mobile and Autonomous Robotics Integration Environment [MARIE]

MARIE is a flexible distributed system that allows developers to share, reuse, and integrate robotic software programs for rapid application development [15]. It uses the ACE [7] communication libraries as integration infrastructure. There are four functional components: application adapters, application managers, communication adapters and communication managers. Its architecture allows interoperability among elements and re-usability of application components.

---

[4] http://www.microsoft/robotics

[5] http://carmen.sourceforge.net

Robot-Technology (RT) middleware

The RT middleware [16] uses CORBA to build robot software in a modular structure, simplifying this process by combining selected modules. A further feature of this middleware is to allow the distribution of robot resources over a network.

Sensory Data Processing Middleware

The Sensory Data Processing Middleware [17] abstracts services to access sensor information and support service mobile robots. It provides a unified model for different configurations of external sensors on a mobile robot. Developed services can be reused in applications without dealing with individual sensors. Two types of services are implemented to provide obstacle information and to localize the robot position using landmark observations from multiple external sensors.

Universal Plug and Play (UPnP) Robot middleware

The UPnP Robot middleware [18] was developed to offer peer-to-peer network connectivity among PCs, wireless pervasive devices and intelligent appliances. It uses the Universal Plug and Play—UPnP[6] architecture for dynamic software integration and for control of ubiquitous robots. UPnP mechanisms are used to configure robot components and to allow distributed robots to discover and interact with other devices like cameras and sensor networks. This approach provides a simple scheme for building intelligent robots with a wide variety of hardware and software components.

2.4 Cooperating Objects Testbeds with No Integration

A very large number and variety of testbeds have been developed in each of the individual technological fields within the Cooperating Objects domain. In these testbeds cooperation is constrained to elements belonging to the same field. However, the analysis of their approaches, capabilities and tendencies is very interesting to derive requirements for the design of full Cooperating Objects testbeds. We divide them in static and mobile testbeds. It is not the objective of this section to provide an exhaustive description but to give an overall view illustrating different schemes and approaches.

2.4.1 Static Cooperating Objects testbeds

They basically include WSN testbeds, camera networks testbeds and smarphone testbeds. We focus on the first one in which a high number and variety of approaches can be found.

A very high number of WSN testbeds have been developed. Testbeds are the most popular experimental tools in WSN community. Many of them have been designed

[6] http://www.upnp.org

as a service to the research community: they are modular, can allocate a very wide range of experiments and are open to the public. Others have been developed to be focused on particular applications or scenarios. The main tendencies in current WSN testbeds include the following:

- federations of testbeds,
- outdoor testbeds in real-world settings,
- integration with robots and other actuation devices.

TWIST [19] at Technische Universität Berlin is a good example of a publicly accessible heterogeneous WSN testbed. It is comprised of 102 TmoteSky nodes and 102 eyesIFX nodes deployed in a 1,500 m^2 scenario in three floors of an instrumented office. TWIST allows public remote access through a simple HTML interface. Its software architecture is modular, flexible and highly extensible, being able to hold a wide range of experiments. In fact, its architecture has been used in other testbeds such as WUSTL [20]. Other publicly accessible general purpose WSN testbeds are: DES-Testbed [21] at Frei Universität Berlin; SensorNet [22] in the Intel Research Center at University of California at Berkeley, comprised of 97 Crossbow Mica2 and 51 Mica2Dot nodes; NetEye [23] with 130 TelosB nodes; and the INDRIYA testbed [24] at University of Singapore, see Fig. 2.1.

Besides, many testbeds have been developed to meet particular needs or functionalities, losing generality but gaining efficiency. Networking is a commonly targeted research topic in these functionality-driven WSN testbeds. Some examples are: WiNTER testbed [25] at Acadia University, NESC testbed [26] and VineLab testbed [27]

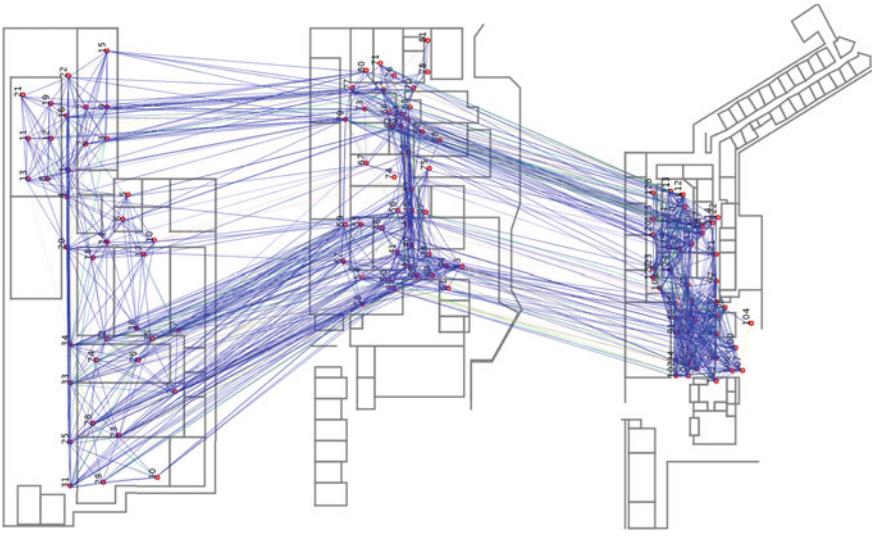

Fig. 2.1 Typical WSN node connectivity in INDRIYA testbed (courtesy National University of Singapore)

at University of Virginia. BANAID [28] targets security and worm-hole attack problems. The Imote2 testbed [29] deals with localization using RSSI from a grid of Crossbow Imote2 nodes. RadiaLE at ISEP [30] is a remotely and publicly accessible outdoors testbed that targets radio management and link quality estimators.

One of the latest tendencies is to group several testbeds under a federation. Various testbeds located at different places can be used transparently as one, benefiting from the variety of capabilities/setting each one provides. Federated testbeds share resources -virtually or physically- and can be accessed with the same API, which allows for instance testing the same experiment in different settings. The number of WSN nodes in federated testbeds ranges between around 500 in KanseiGeni [31] to 1,024 in SensLab.[7] Other examples are XSensor [32], WISEBED [33] and COFEDE.[8] It is interesting to mention the *FP7 Future Internet Research and Experimentation* (FIRE) program [34] in which a number of testbeds with a common architecture are in development. The FIRE architecture has been defined to ensure compatibility among all the FIRE testbeds.

One recent trend is the development of large outdoors testbeds for experimentation in real-world conditions. TeamTrack [35] is a characteristic example. It includes fully-instrumented tablet kits—equipped with GPS, camera and digital compass—and lightweight handheld kits (an HP handheld and Bluetooth GPS unit). TeamTrack is fairly scalable and mobility is provided by people carrying the nodes. CitySense [36] is an open testbed at Harvard University that integrates 100 Linux-based embedded PCs with dual IEEE 802.11a/b/g radios and custom sensor nodes. SmartSantander[9] is an European project in which 20,000 nodes based on XBee-PRO will be deployed in four European cities, see Fig. 2.2.

2.4.2 Mobile Cooperating Objects testbeds

They are basically mobile robot testbeds. A very high number of robot and multi-robot testbeds have been developed along the years. Experimentation in real-world conditions is critical in robotics research due to the difficulties to simulate physical interaction. Most robotic testbeds have been developed specifically to test target functionalities or applications. A number of testbeds capable of allocating a wider range of experiments. The main tendencies in current networked robots testbeds include the following:

- testbeds of aerial robots,
- integration of robots with different mobility capabilities,
- integration with ubiquitous systems.

[7] http://www.senslab.info

[8] http://www.cotefe.net

[9] http://www.smartsantander.eu

Fig. 2.2 General view of node deployment in the city of Santander, Spain (courtesy Telefonica I+D)

In robotics the need to test the methods in real conditions has motivated the development of a high number of robot testbeds designed to address specific functionalities or applications. Only some examples are to be cited. The Micro Autonomous RoverS (MARS) testbed, used in [37], at Stanford University, comprised of six custom made rovers, was developed for research in human–robot interaction, motion planning and object tracking. The CALTECH Multi-vehicle Wireless Testbed [38] includes 8 custom hovercrafts. CALTECH testbed targeted experiments in multi-robot coordination, networked control and real-time networking. Other functionality driven robot testbeds are: UTIAS testbed at University of Toronto, used to test the techniques proposed in [39]; the testbed used to validate the method in [40], at the Massachusetts Institute of Technology (MIT), which targets multi-robot exploration; the testbed used in [41] at Kansas State University, which focuses on search and rescue applications; [42] at Columbia University, which targets network connectivity and data collection; and the testbed used to validate [43] at University of California-Los Angeles (UCLA), which focuses on boundary tracking and estimation.

The need to compare and evaluate different methods in the same conditions motivated the development of general purpose testbeds. General purpose testbeds are generally designed to be used by a broader public than functionality-driven testbeds. This involved efforts to have architectures with higher modularity, flexibility and extensibility. It also involved higher interest in code reuse, openness approach and the use of common languages and interfaces. [44] describes a testbed in the GRASP Laboratory at the University of Pennsylvania based on Player/Stage [2]. Its architecture allows good scalability and easy operation even with large deployments of mobile robots. The COMET testbed [45] integrates 10 robots under a flexible and extensible architecture. The Autonomous Mobile Robotics Systems (AMRS) testbed [46] at University of Louisville also uses Player/Stage as supporting middleware.

The AMRS testbed is comprised of customized platforms while the GRASP testbed includes SCARAB robotic platforms.

The generalization of robot testbeds as tools for methods evaluation involved a higher need for remote access and used-friendly operation. The Mint-m testbed [47], developed at the Stony Brook University, comprised of 12 iRobot Roomba platforms controlled through a central processor that uses overhead cameras to compute robot poses. The testbed uses an ad-hoc software architecture with high flexibility and reconfigurability. It can be remotely operated and has tools for experiment visualization and logging. The testbed also offers tools for experiment simulation. Similarly, HoTDeC [48] at University of Illinois at Urbana-Champaign provides a GUI for remote visualization and monitoring and allows remote configuration and high level robot control.

The aforementioned testbeds integrate only ground robots. In the last years a number of testbeds using Unmanned Aerial Vehicles (UAV) and Autonomous Underwater Vehicles (AUV) have been developed. Below are some examples: [49] at MIT Space Systems and Artificial Intelligence Laboratories uses 3 custom flying crafts; [50] at University of Southern California (USC) integrates customized UAVs based on helicopters; the GRASP Multiple MAV testbed [51] includes *Ascending Technologies* Hummingbird quad-rotors[10]; and the CATEC[11] testbed is comprised of 10 quadrotors, see Fig. 2.3.

Some testbeds integrating aerial and ground robots have been developed. An example is RAVEN [52] at MIT, which integrates Draganflyer V Ti Pro quad-rotors, a foam R/C aircraft and modified DuraTrax RC trucks. MAGICC Lab. [53] developed

Fig. 2.3 General overview of CATEC testbed (courtesy FADA-CATEC)

[10] http://www.asctec.de

[11] http://www.catec.com.es

a testbed with 5 custom ground robots, one omnidirectional robot and 4 fixed-wing UAVs. The testbed used in [54] at MIT, equipped with 2 custom remotely operated underwater vehicles, is an example of an underwater multi-robot testbed.

2.5 Cooperating Objects Testbeds with Low Integration

This type of testbeds allows some interoperation between heterogenous Cooperating Objects technologies. They are often designed so that one of the technologies—primary—has control over the other. Therefore, the balance of experiments that can be held is biased either towards ubiquitous systems or towards robotics and only part of the features of the secondary technology are exploited. In some cases the testbed focus is on WSN and robots are essentially used to provide mobility to some nodes. Others are focused on robotics: they are robot testbeds enhanced with a WSN, which is essentially used as a distributed sensor or as communication backbone. Again, the objective of this section is to provide an overview of the different approaches and trends rather than an exhaustive enumeration.

The Hybrid sensor network testbed at Deakin University was developed to test localization methods in delay-tolerant sensor networks. It is comprised of static Mica2 WSN nodes and Lego robots. The robots locate the static nodes using RSSI while they move following paths. Each robot carries one Stargate board that receives RSSI readings from the static WSN and executes the localization method. There is no explicit communication between each robot and its Stargate board: robots are used to carry the boards along predefined paths.

Mobile Emulab [55] is a general purpose WSN testbed that uses robots to provide mobility to nodes. Developed at University of Utah, it is comprised of 5 *Acroname* Garcia robots and a static WSN with 25 Mica2 nodes. Each robot carries one WSN node but there is no communication between them.

Mobile Emulab includes simple robot obstacle avoidance and path-planning methods but the testbed focus and main objective are research in WSN. The testbed uses an open and modular architecture and includes a GUI for remote operation and provides high degree of interaction with the user and can show live maps and images of the experiment. It was open for public use until 2008.

The iMouse testbed [56] targets cooperative surveillance integrating a WSN and a fleet of ground robots. The WSN operates as a distributed sensor that triggers events to be monitored by the robot fleet. Coordination among systems is achieved through a central server that receives events from the WSN and assigns robots to event locations: there is no direct communication between robots and WSN. The robots are based on *Lego* Mindstorm platforms enhanced with a Stargate board and a webcam, whereas its WSN is comprised of 17 MicaZ nodes. The iMouse has a centralized architecture and uses an ad-hoc software architecture.

Sensei UU [57] targets localization experiments. It is comprised of *Lego* NXT based robots to enhance a static WSN with repeatable mobility. Each robot includes a TelosB node and one smartphone, which acts as the main robot processor. The

testbed uses two networks: one based on IEEE 802.11b/g (Smartphones) and one based on IEEE 802.15.4 (WSN nodes). Robot motion is controlled from the central site manager, which transmits commands to the robot through the smartphone. The robot platform reports the estimated traveled distance to the smartphone, which uses it to determine the robot position. The WSN node reports the received RSSI to the smartphone using simple unidirectional messages. Robot motion is constrained to fixed paths. Markers on the tracks are used for robot localization.

The Kansei testbed [58] is comprised of one static WSN with 210 nodes and *Acroname* mobile robots. Each static node includes one Extreme Scale Mote (XSM) and one Stargate board. Robots are equipped with one XSM that communicates with the static WSN. Robots and WSN are integrated by the testbed Director, a centralized software platform that enables uniform, integrated experimentation. The testbed has a modular, extensible and scalable architecture. It also includes software components for job management in complex multi-tier experiments. It can be accessed remotely and includes a hybrid hardware–software simulation engine to facilitate experimentation.

Work [59] describes a testbed for multi-robot coordination experiments comprised of heterogeneous ground robots including micro vehicles enhanced with custom sensing and wireless communication devices. Robotic platforms are connected to WSN nodes through a serial port. These nodes provide computing and communication capabilities and send direct commands to the robotic platforms. The testbed is equipped with overhead cameras to compute robots ground-truth poses. Also, an optional data monitor can be used for experiment logging and visualization. Its uses an ad-hoc software architecture with good flexibility and allows distributed and centralized experiments.

The Explorebots testbed [60] focuses on experimentation of protocols for mobile multi-hop networks. It uses robots based on customized Rogue ATV platforms, each carrying a Mica2 node as communication module. The robots are equipped with webcams, electronic compasses and range sensors. Each robot has a WSN node that provides commands to and obtains information from it using a simple protocol. The testbed uses a closed architecture with low flexibility and extensibility. A GUI allows on-site visualization of images and data.

Robomote [61] is a general purpose testbed comprised of custom micro robotic platforms—Robomote—equipped with simple compass and proximity sensors. A Mica2 WSN node acts as the master of each mobile node. The Robomote processor is in charge of the communication with the sensors and actuators and with the Mica2, which implements higher level sensing and mobility control functions. It uses a modular and open distributed software architecture based on TinyOS.

2.6 Cooperating Objects Testbeds with High Integration

They have the capacity to hold Cooperating Objects balanced experiments in which different heterogeneous devices from different technological fields work as peers. Few existing testbeds comprise the necessary hardware and software infrastructure

to belong to this category, and the majority of them are often focused on particular functionalities, with the corresponding loss of generality. Different designs of a general Cooperating Objects testbed have been proposed in a number of works. However, in few of them the testbed is implemented and made open to the community. UBIROBOT[12] comprises WSN, PDAs, smartphones and mobile robots equipped with a variety of sensors. Apart from a general design we could not find further scientific publications, details or information on its current availability. In some other cases, the approach and capabilities are promising but the testbed was developed specifically for experimentation of some functionalities, which limits its impact in the community. That is the case of the SWARM testbed [62], which focused on swarm cooperation between a fleet of robots and a static WSN, considered also as a swarm.

The testbed for networked Physically Embedded Intelligent Systems (PEIS) targets methods for the use of ubiquitous systems in every day live [63]. The PEIS-home testbed, at the AASS Mobile Robotics Laboratory (University of Örebro), is a small apartment equipped with a fleet of robots Magellan Pro robot and Roomba, among others, with communication infrastructure, ubiquitous computing devices, a camera network, automatic appliances and embedded sensors, see Fig. 2.4. It also includes WSN and RFID technologies. The PEIS architecture is centralized modular, flexible and highly extensible and scalable.

ISROBOTNET [64] was developed within the framework of the URUS project on ubiquitous robotics in urban settings. Thus, its design and architecture aim to meet the project needs. The testbed includes a camera network with ten webcams and 5 robots (4 Pioneer 3AT and one ATRV-Jr) equipped with self-localization and obstacle avoidance sensors. WSN nodes are used as distributed sensors for localization and tracking. The testbed uses a modular and flexible service-oriented middleware based on open-source software to simplify the integration of subsystems developed by different users and to enable easy transition from simulation to experimentation.

Fig. 2.4 Pictures from the PEIS testbed (courtesy University of Örebro)

[12] http://ubirobot.ucd.ie

It includes software modules with image processing algorithms for human activity detection, people tracking using static and mobile cameras, among others.

2.7 Conclusions

The interest in Cooperating Objects technologies has motivated the development of a high number and variety of experimental testbeds. Despite the number and variety of existing testbeds, most of them focus on cooperation among elements from the same technological field. Very few of them enable full integration and interoperability between elements from different Cooperating Objects fields and, those which do, are focused on specific applications or functionalities. This lack of suitable testbeds for integrated Cooperating Objects research has been pointed out as one major drawback for the development of the Cooperating Objects domain. The CONET Cooperating Objects Integrated Testbed was designed to fill in this gap.

References

1. Marrón PJ, Minder D, Karnouskos S (2012) The emerging domain of cooperating objects: definition and concepts, Springer. http://www.springer.com/engineering/signals/book/978-3-642-28468-7
2. Gerkey BP, Vaughan RT, Howard A (2003) The player/stage project: tools for multi-robot and distributed sensor systems. In: Proceedings of the international conference on advanced robotics, pp 317–323
3. Koenig N, Howard A (2004) Design and use paradigms for gazebo, an open-source multi-robot simulator. In: IEEE/RSJ international conference on intelligent robots and systems, pp 2149–2154
4. Gerkey B, Vaughan R, Stoy K, Howard A, Sukhatme G, Mataric M (2001) Most valuable player: a robot device server for distributed control. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, vol 3. pp 1226–1231
5. Metta G, Fitzpatrick P, Natale L (2006) Yarp: yet another robot platform. Int J Adv Robot Syst 3:43–48
6. Utz H, Sablatnog S, Enderle S, Kraetzschmar G (2002) Miro-middleware for mobile robot applications. IEEE Trans Robot Autom 18(4):493–497
7. Schmidt DC (1994) Ace: an object-oriented framework for developing distributed applications. In: Proceedings of the 6th USENIX C++ technical conference
8. Harrison TH, Levine DL, Schmidt DC (1997) The design and performance of a real-time corba event service. SIGPLAN Not 32:184–200
9. Mohamed N, amd Imad Jawhar JAJ (2009) A review of middleware for networked robots. Int J Comput Sci Netw Secure IJCSNS 9(5):139–148
10. Bruyninckx H (2001) Open robot control software: the orocos project. In: Proceedings of the IEEE international conference on robotics and automation, ICRA2001, vol 3. pp 2523–2528
11. Brooks A, Kaupp T, Makarenko A, Williams S, Oreback A (2005) Towards component-based robotics. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, IROS2005, pp 163–168
12. Montemerlo M, Roy N, Thrun S (2003) Perspectives on standardization in mobile robot programming: the carnegie mellon navigation (carmen) toolkit. In: Intelligent robots and systems,

2003. (IROS 2003), Proceedings of the 2003 IEEE/RSJ international conference on, vol 3. pp 2436–2441. DOI:10.1109/IROS.2003.124923
13. Yoo J, Kim S, Hong S (2006) The robot software communications architecture (rsca): Qos-aware middleware for networked service robots. In: Joint international conference SICE-ICASE, pp 330–335
14. Schmidt D, Gokhale A, Harrison T, Parulkar G (1997) A high-performance end system architecture for real-time corba. IEEE Commun Mag 35(2):72–77
15. Cote C, Brosseau Y, Letourneau D, Raievsky C, Michaud F (2006) Robotic software integration using marie. Int J Adv Robot Syst 3(1):55–60
16. Ando N, Suehiro T, Kitagaki K, Kotoku T, Yoon WK (2005) Rt-middleware: distributed component middleware for rt (robot technology). In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems IROS2005), pp 3933–3938
17. Takeuchi E, Tsubouchi T (2006) Sensory data processing middlewares for service mobile robot applications. In: Joint international conference SICE-ICASE, pp 1201–1206
18. Ahn SC, Lee JW, Lim KW, Ko H, Kwon YM, Kim HG (2006) Requirements to upnp for robot middleware. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, IROS2006, pp 4716–4721
19. Handziski V, Köpke A, Willig A, Wolisz A (2006) Twist: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In: Proceedings of the 2nd ACM international workshop on multi-hop ad hoc networks: from theory to reality, New York, pp 63–70
20. Sha M, WUSTL wireless sensor network testbed. http://wsn.cse.wustl.edu/index.php/The_WUSTL_Wireless_Sensor_Network_Testbed Accessed in June 2013
21. Günes M, Blywis B, Juraschek F (2008) Concept and design of the hybrid distributed embedded systems testbed. Technical report TR-B-08-10, Freie Universität Berlin
22. Chun BN, Buonadonna P, AuYoung A, Ng C, Parkes DC, Shneidman J, Snoeren AC, Vahdat A (2005) Mirage: a microeconomic resource allocation system for sensornet testbeds. In: Proceedings of the 2nd IEEE workshop on embedded networked sensors, pp 19–28
23. Sakamuri D, Zhang H (2009) Elements of sensornet testbed design. World Scientific Publishing Co, USA, pp 1–36 (chapter 35)
24. Doddavenkatappa M, Chan MC, Ananda A (2012) Indriya: a low-cost, 3d wireless sensor network testbed. In: Testbeds and research infrastructure. Development of networks and communities, Lecture notes of the institute for computer sciences, social informatics and telecommunications engineering, vol 60. pp 302–316
25. Murillo MJ, Slipp JA (2009) Application of winter industrial testbed to the analysis of closed-loop control systems in wireless sensor networks. In: Proceeding of the 8th ACM/IEEE International Conference on Information Processing in Sensor Networks
26. Kejie C, Xiang G, Xianying H, Jiming C, Youxian S (2008) Design and develop of the wsn testbed: Nesc-testbed. In: China science paper online
27. Whitehouse K, The vinelab wireless testbed. http://www.cs.virginia.edu/~whitehouse/research/testbed/ Accessed in June 2013
28. Alzaid H, Abanmi S (2009) A wireless sensor networks test-bed for the wormhole attack. Int J Digit Content Tech Appl 3(3):19–27
29. Poovendran R, Imote2 sensor network testbed. http://www.ee.washington.edu/research/nsl/Imote/ Accessed in June 2013
30. Baccour N, Koubâa A, Jamâa MB, do Rosário D, Youssef H, Alves M, Becker LB, (2011) Radiale: a framework for designing and assessing link quality estimators in wireless sensor networks. Ad Hoc Netw 9(7):1165–1185
31. Sridharan M, Zeng W, Leal W, Ju X, Ramnath R, Zhang H, Arora A (2010) Kanseigenie: software infrastructure for resource management and programmability of wireless sensor network fabrics. Next generation internet architectures and protocols, Cambridge University Press. http://ebooks.cambridge.org/chapter.jsf?bid=CBO9780511920950&cid=CBO9780511920950A115

32. Kanzaki A, Hara T, Ishi Y, Wakamiya N, Shimojo S (2009) X-sensor: a sensor network testbed integrating multiple networks. In: International conference on complex, intelligent and software intensive systems, pp 1082–1087
33. Chatzigiannakis I, Fischer S, Koninis C, Mylonas G, Pfisterer D (2010) Wisebed: an open large-scale wireless sensor network testbed. In: Sensor applications, experimentation, and logistics, lecture notes of the institute for computer sciences, social informatics and telecommunication Engineering, vol 29. Springer, Berlin, pp 68–87
34. EC, FP7 Future internet research and experimentation (FIRE). http://www.ict-fire.eu/home/fire-projects.html Accessed in June 2013
35. Hemmes J, Thain D, Poellabauer C, Moretti C, Snowberger P, McNutt B (2007) Lessons learned building teamtrak: An urban/outdoor mobile testbed. In: Proceedings of the international conference on wireless algorithms, systems and applications, WASA2007, pp 219–224
36. Murty R, Mainland G, Rose I, Chowdhury A, Gosain A, Bers J, Welsh M (2008) Citysense: an urban-scale wireless sensor network and testbed. In: Proceedings of the IEEE conference on technologies for homeland security, pp 583–588
37. Clark CM, Frew EW, Jones HL, Rock SM (2003) An integrated system for command and control of cooperative robotic systems. In: Proceedings of the 11th international conference on, advanced robotics, pp 459–464
38. Cremean L, Dunbar W, van Gogh D, Hickey J, Klavins E, Meltzer J, Murray R (2002) The caltech multi-vehicle wireless testbed. In: Proceedings of the 41st IEEE conference on decision and control, vol 1. pp 86–88
39. Marshall JA, Fung T, Broucke ME, DEleuterio GM, Francis BA, (2006) Experiments in multirobot coordination. Robot Auton Syst 54(3):265–275
40. Williams BC, Kim P, Hofbaur M, How J, Kennell J, Loy J, Ragno R, Stedl J, Walcott A (2001) Model-based reactive programming of cooperative vehicles for mars exploration. In: Proceedings of the international symposium on artificial intelligence, robotics and automation in space
41. Deloach SA, Matson ET, Li Y (2003) Exploiting agent oriented software engineering in cooperative robotics search and rescue. Int J Pattern Recognit Artif Intell 17(5):817–835
42. Reich J, Misra V, Rubenstein D (2008) Roomba madnet: a mobile ad-hoc delay tolerant network testbed. SIGMOBILE Mob Comput Commun Rev 12:68–70
43. Joshi A, Ashley T, Huang Y, Bertozzi A (2009) Experimental validation of cooperative environmental boundary tracking with on-board sensors. In: Proceedings of the American control conference, ACC2009, 2630–2635
44. Michael N, Fink J, Kumar V (2008) Experimental testbed for large multirobot teams. IEEE Robot Autom Mag 15(1):53–61
45. Cruz D, McClintock J, Perteet B, Orqueda O, Cao Y, Fierro R (2007) Decentralized cooperative control—a multivehicle platform for research in networked embedded systems. IEEE Control Syst 27(3):58–78
46. Riggs T, Inanc T, Zhang W (2010) An autonomous mobile robotics testbed: construction, validation, and experiments. IEEE Trans Control Syst Technol 18(3):757–766
47. De P, Raniwala A, Krishnan R, Tatavarthi K, Modi J, Syed NA, Sharma S, Chiueh Tc (2006) Mint-m: an autonomous mobile wireless experimentation platform. In: Proceedings of the 4th international conference on mobile systems, applications and services, MobiSys2006, ACM, New York, USA, pp 124–137
48. Stubbs A, Vladimerou V, Fulford A, King D, Strick J, Dullerud G (2006) Multivehicle systems control over networks: a hovercraft testbed for networked and decentralized control. IEEE Control Syst 26(3):56–69
49. Miller D, Seanz-Otero A, Wertz J, Chen A, Berkowski G, Brodel C, Carlson S, Carpenter D, Chen S (2000) Spheres-a testbed for long duration satellite formation flying in micro-gravity conditions. In: Proceedings of the AAS/AIAA space flight mechanics meeting, pp 167–179
50. Naffin D, Sukhatme G (2002) A test bed for autonomous formation flying. In: Institute for robotics and intelligent systems, Technical, Report IRIS-02-412

51. Michael N, Mellinger D, Lindsey Q, Kumar V (2010) The grasp multiple micro-uav testbed. IEEE Robot Autom Mag 17(3):56–65
52. How J, Bethke B, Frank A, Dale D, Vian J (2008) Real-time indoor autonomous vehicle test environment. IEEE Control Syst 28(2):51–64
53. McLain T, Beard R (2004) Unmanned air vehicle testbed for cooperative control experiments. In: Proceedings of the American control conference, ACC2004, vol 6. pp 5327–5331
54. Brundage H, Cooney L, Huo E, Lichter H, Oyebode O, Sinha P, Stanway M, Stefanov-Wagner T, Stiehl K, Walker D (2006) Design of an rov to compete in the 5th annual mate rov competition and beyond. In: Proceedings of the OCEANS 2006, pp 1–5
55. Johnson D, Stack T, Fish R, Flickinger DM, Stoller L, Ricci R, Lepreau J (2006) Mobile emulab: a robotic wireless and sensor network testbed. In: Proceedings of the 25th IEEE international conference on, computer communications, pp 1–12
56. Tseng YC, Wang YC, Cheng KY, Hsieh YY (2007) Imouse: an integrated mobile surveillance and wireless sensor system. Computer 40(6):60–66
57. Rensfelt O, Hermans F, Gunningberg P, Larzon L (2010) Repeatable experiments with mobile nodes in a relocatable wsn testbed. In: Proceedings of the 6th IEEE international conference on distributed computing in sensor systems, Workshop, pp 1–6
58. Arora A, Ertin E, Ramnath R, Nesterenko M, Leal W (2006) Kansei: a high-fidelity sensing testbed. IEEE Internet Comput 10(2):35–47
59. Leung K, Hsieh C, Huang Y, Joshi A, Voroninski V, Bertozzi A (2007) A second generation micro-vehicle testbed for cooperative control and sensing strategies. In: Proceedings of the American control conference, ACC2007, pp 1900–1907
60. Dahlberg TA, Nasipuri A, Taylor C (2005) Explorebots: a mobile network experimentation testbed. In: Proceedings of the ACM SIGCOMM workshop on experimental approaches to wireless network design and analysis, ACM, New York, USA, pp 76–81
61. Dantu K, Rahimi M, Shah H, Babel S, Dhariwal A, Sukhatme G (2005) Robomote: enabling mobility in sensor networks. In: Proceedings of the 4th international symposium on information processing in sensor networks, IPSN 2005, pp 404–409
62. Li W, Shen W (2011) Swarm behavior control of mobile multi-robots with wireless sensor networks. J Netw Comput Appl 34(4):1398–1407, advanced topics in cloud computing
63. Saffiotti A, Broxvall M (2005) Peis ecologies: ambient intelligence meets autonomous robotics. In: Proceedings of the joint conference on smart objects and ambient intelligence: innovative context-aware services: usages and technologies, ACM, New York, USA, pp 277–281
64. Barbosa M, Bernardino A, Figueira D, Gaspar J, Goncalves N, Lima P, Moreno P, Pahliani A, Santos-Victor J, Spaan M, Sequeira J (2009) Isrobotnet: a testbed for sensor and robot network systems. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems IROS2009, pp 2827–2833

http://www.springer.com/978-3-319-01371-8

A Remote Integrated Testbed for Cooperating Objects

Martinez-de Dios, J.R.; Jimenez-Gonzalez, A.; San Bernabe, A.; Ollero, A.

2014, XIII, 79 p. 31 illus., 28 illus. in color., Softcover
ISBN: 978-3-319-01371-8