

Fund Raising Systems using Robots (Architecture and Behavior Study)

Gaetano La Russa*
larussa@cs.joensuu.fi

Eugene N. Ageenko
ageenko@cs.joensuu.fi

Yaroslav Karulin
ykarulin@cs.joensuu.fi

Department of Computer Science, University of Joensuu
PB 111, 80101, FIN-80101, Joensuu, Finland

Abstract – In this article we present an interactive automated robot called RoboBegger that has fund raising capabilities. The system consists of a human-like multimodal robotic unit controlled by a computer. A stand with a touch screen display and a bank card reader provides access to the e-learning and e-commerce applications of the robot. The primary use for the existing robotic system is to collect donations for charity activities and to transfer charity related knowledge to the users of the robot. The robot can be placed in major public places. In the test case presented in this article, its location was a church during the Christmas festivities of 2004. The control system of the robot is a software built upon a Java framework in a way that it can be used for a variety of applications where human interaction is required (i.e. shopping, listing adviser, consultancy, etc). This article describes the architecture of the robotic system, the solution chosen for the simple management of its complex interface modules and its feasibility for providing insightful data about human interaction. In this article we also discuss possible user scenarios and future perspectives.

Index Terms – Fund Raising, Automation Management, Interactive Behavior, RoboBegger.

I. INTRODUCTION

Robotics is a research field that is actively being developed also in education and service related fields. Some of the primary interests of robotics researchers are focused in the following areas: *embedded system design* and *artificial intelligence* [1,2].

From the technical point of view the *begging robot* (named *RoboBegger*) is a complex combination of software and hardware devices. (See Fig.1 for a picture of the robot). The main difference between a traditional technical appliance meant for money collection and the RoboBegger is in RoboBegger's awareness of the situation. The robot reacts to the behavior of the eventual user and behaves according its role in the location environment, stimulating the interests of other potential users that might be in its proximity. In response to user actions, the robot software controls several peripheral devices that result in movement of robot parts, playing appropriate sounds, feeding of specific e-learning modules, etc. The reactions of the robot must be quick and adequate to the situation. It is well known that human interaction with robots can produce novel experiences and that for this to happen it is also required that human and robot can perceive each other [3]. In this article we present a possible solution for concretizing a HRI for boosting the perception and the cognitive processes.

The robots must perform some intelligent actions. These are related to a pre-defined idle condition in which the robot calls and attracts the attention of passersby. As

soon as a user starts to interact with the robot via the touch screen, the robot uses some of its modules to provide the user with the information the user wants. Meanwhile the robot takes into account the behavior of the user and gives suggestions on specific matters or encourages the user to proceed in the use of the learning modules. Elements of cognitive science have been taken into account to track, stimulate and, where possible, to foresee the learning behavior of the users [4]. This robotic *intelligence* is the feature that makes the using the robot attractive to the user.

The software controlling the robot is constructed in such a way that easily allows for addition or modification of hardware. It is also possible to modify the robot behavior scheme (its *intelligence* part) in an easy and straightforward way. This specific part of the software, which is continuously under improvement, is intended to make use of metadata and tracking information collected for each user so that the robot can recognize the specific user and, therefore, deal with the user in a personalized way.



Fig. 1. The fund raising robot (the RoboBegger)

One of the relevant aspects of the design of the management interface is that the needs of the laypeople that will probably operate the robot have been taken into account. The *operator* of the robot does not need to possess deep knowledge or understanding of the entire robot structure in order to maintain it. Indeed, with the interface

that has been developed, the operator is able to quickly change the sequence of the robot's actions and most of the other features related to the robot's behavior. The result is that the robot operator can maintain the robot in an easy, quick and intuitive way that enables the creation of new e-learning modules and, consequently, allow the operator to specify new robotic behaviors [5].

II. SYSTEM DESCRIPTION

A. *RoboBeggart interaction and behavior*

In idle mode, the robot vocally invites passersby, with randomly chosen messages, to approach and start using the system. Approaching the robot, users find the touch screen has a rotating, 3D world-globe screensaver on which an invitation to begin interaction is displayed. A touch from a user activates the robot, which responds by thanking the users for accepting the offer for interaction. A window is then presented where the user has the opportunity to register with a personal code or to continue anonymously. In the case that the user had already created a personal code in a previous session, the insertion of the same code allows the personalization of the user profile, which is stored in the robot. Next to the access-code window, the robot presents the first e-learning module (*e-module*) where basic information and links to other more detailed e-modules and resources are given. A domain-restricted internet access is also present in this first general e-module that gives the user the opportunity to explore, in more detail, the arguments for giving to charities and to collect information on charitable giving in general. Browsing is time-limited. The robot keeps talking and encouraging users through the interaction process and in users' search for information. Its limb and head move according to the situation (i.e., specific e-module, expired time from last user interaction, etc.) to simulate human-like interaction [4]. The user, when going through a detailed e-module, can choose to make a donation at anytime. A backward function, which allows the user to move back and forth between the e-modules, is available in every window. After choosing to donate, the user is presented with another window, the donation window, where pictures (*value pictures*) of 1 and 2 EURO coins and 5, 10, 20, 50 and 100 EURO banknotes are preset. The donation window reminds users of the specific donation target. After a user touches one of the value pictures, the robot brings up a confirmation window where the user is reminded of the specific donation (i.e. the amount of money and charitable destination) and asked to pass the bankcard in the bankcard reader slot. While the user passes the bankcard through the bankcard slot, the robot reads the security code and transmits money transfer information to the bank via a secured GSM channel. After this transaction, a thank-you screen appears and has a summary of the information related to the donation. A receipt of the donation is printed that indicates the time, the chronological donation number, the occasion, the purpose of the donation, and the amount of given money. A short while after the thank-you window appears, the robot switches into idle mode and the world-globe screensaver reappears.

When the robot is in interaction mode, it records the time that elapses between each response of the user. The

robot has a predefined time to act or react to users' activity or inactivity. In case too much time has elapsed in one e-module phase, the robot first warns that time will shortly expire and then, soon afterwards, switches to idle if no response from the user is received. Hence the robot talks and moves based on users' responses or lack of responses. The overall aim of the robot is to give to the user a sense of collaboration in the process of charitable giving and to provide the cognitive aura to stimulate users' actions. This cognitive aura is discussed and sustained in the article "Robo-eLC: A Robotic Adaptive Learning Appliance" presented at the International Conference on Cognition and Exploratory Learning in Digital Age, CELDA, 2004. [4]

All rules that dictate the specific actions of the robot are set by the robot manager or by the operator. The rules regulate when and in what circumstances any specific robotic action happens. These actions include the movements of the head and the right limb, the use of vocal messages, the flow of the touch screen windows, the use and breadth of Internet access, the use of the bankcard reader and the transmission of the secured data, the printing of the receipt and finally the time-counting between each possible action.

All results obtained show that the behavior of the robot produces a positive effect. In general, users feel excited and positively motivated in browsing the e-modules and acquiring information. Most adult users made a donation, which resulted more consistent when compared to similar and local collecting situations.

B. *System building approach*

One approach to building the robot is to build it as an *embedded system*. The Embedded System is a combination of computer hardware and software, with the addition of other mechanical parts, designed to perform a dedicated function [1]. In some cases, embedded systems are part of a larger system or product, as is the case of an anti-lock braking system in a car. RoboBeggart is a fully functional appliance; it has all of the needed control systems and adaptive modules already within it. The intelligent core controls and arranges the movement of the head and the right arm, the vocal functions, the interaction with the user via the touch screen, the Internet connection and its use, the bankcard reader and its secure data transmission to the banks. The intelligent core will soon allow the RoboBeggart to do its own data analysis and make autonomous decisions concerning what action to take in a given situation.

In our case we aim at constructing an easily extendable model that allows adding new functions without significant change to the existing code. This can be achieved by using an design method in which the system is modeled as a collection of cooperating agents, (i.e., objects.) The individual objects are treated as instances of definitive classes within a certain class hierarchy. There are four design stages: (1) *identify the classes and objects*, (2) *identify their semantics*, (3) *identify their relationships and specify class and object interfaces* and, finally, (4) *implement the design* [6]. To extend the system some external modules should be added or otherwise replace some of the existing ones.

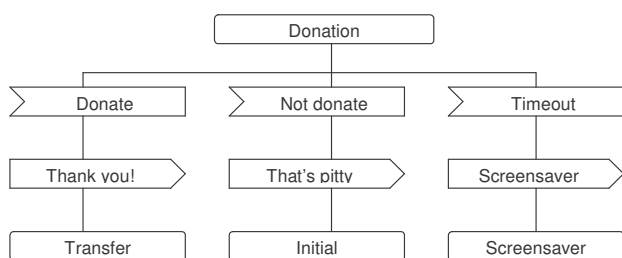


Fig. 2. The rules definition for the donation choice state.

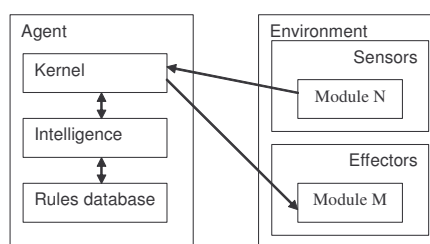


Fig. 3. RoboBegger system architecture.

Another requirement for the system is that several actions should happen simultaneously. Thus each module must have its own execution thread [9].

C. Self-management of the robot

The robot has an intelligent behavior, which means that a set of rules define the robot's behavior. The robot is capable of interacting with the users in a way that encourages appropriate learning behaviors and stimulates their cognitive activities [4]. The robot manages itself according to its set of rules, which constitute its "brain". For flexibility reasons, the rules are stored in an XML file, which is loaded during execution time. The robot also uses frame-based logic. It has several states, and for each state there is a set of rules. When some event happens, the robot chooses the response action according to the specific state, event and rule. An example of rule definition used in the robot is shown in Fig. 2.

D. System architecture

The system consists of several main components. The system class-diagram is outlined in Fig. 4. Each system component is an object, which runs on its own execution thread. The main objects are instances of the classes that inherit the *RoboThread* class [8], which in turn describes common thread behavior. Objects can interact by sending each other messages. The central object of the system is a *Kernel*, which is an instance of the *RoboKernel* class. It dispatches all messages in the system.

For managing the system, there is an *intelligence module*. After receiving message requests, the *Kernel* checks with the intelligence module for the proper order, the intelligence module in turn looks for the available rule and after retrieving it, transmits the data to the kernel that sends the command to the appropriate component of the robot (e.g., to the learning module component, voice component, movement component, etc.) (See Fig.3 and Fig.7). This system avoids many problems related to the handling of multiple commands that caused hang-over situations in previous prototypes. In those situations the robot used to stall the audio components remaining mute or stopped to move for short periods.

This architecture brings flexibility into the system. If anyone wants to change the behavior of the system, the only module to change is the intelligence module. If anyone wants to add any hardware to the system that person should develop a module responsible for the new hardware and

then add some commands or events to the intelligence module.

The system implementation language is Java. It has been chosen because its portability and because it is appropriate for rapid development requirements. The system could be easily ported to various platforms, such as Win32, Unix [7], etc.

E. Message passing

The system is driven by messages. Each module can initiate some set of actions by sending a message to other modules. Usually message sending is caused by some events (e.g., the user has pressed a certain button or the card reading has succeeded.) Messages are sent to some certain module, and the sender specifies the receiver of the message before sending. Receivers are main modules. Main modules are addressed by string identifiers. The string identifiers simplify the use of the system for beginners.

Each module has its own message queue. The selected modules do not process all the messages they have received; they process only the last one. Otherwise, the robot would pronounce, for example, phrases at the wrong time. For this reason, there is a member-variable controlling this behavior.

Message handling is done in a separate thread. This allows the robot to speak, move and change the display picture simultaneously.

F. The Kernel module

The main module of the system is the kernel module. In spite of its importance, its implementation is very simple. The task of the kernel module is to receive event from peripheral devices modules, to request information about actions and to transmit these commands from the intelligence module to the peripheral devices modules. See Fig 5 and Fig 6.

The behavior scheme of other objects is quite similar. The only difference is that they do not request any information from the intelligence module. They interpret commands and handle them by themselves.

G. Intelligence

The current behavior of the robot can be described as a reflex agent [2], where the user action (as environment main source) causes a robotic response based on its rules and metadata. This behavior is graphically described in Fig 7.

The current implementation of the intelligence module uses a XML rules database file. This file is loaded during

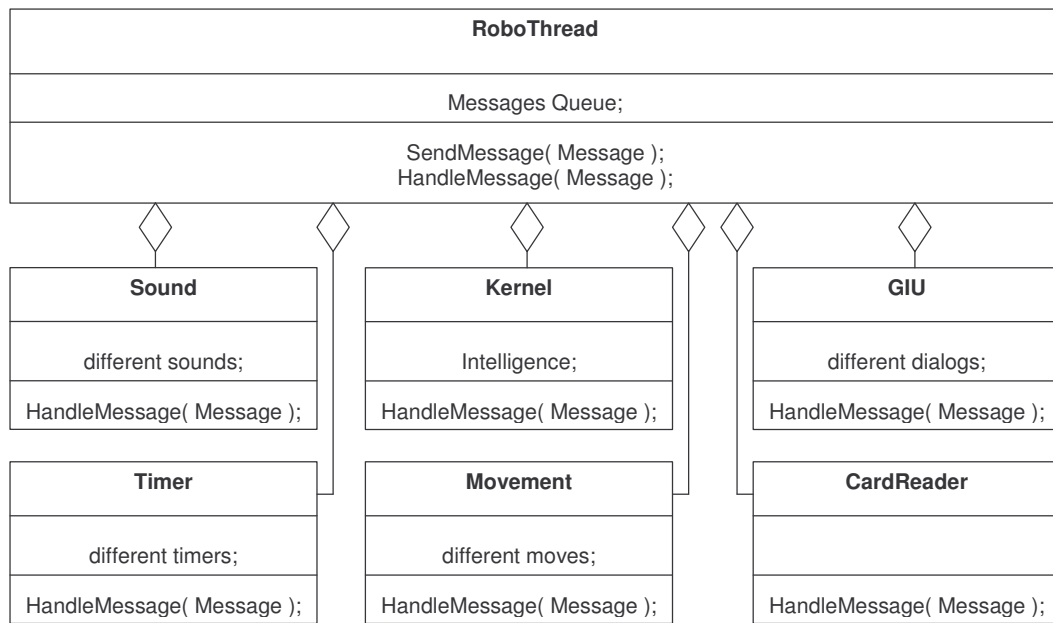


Fig. 4. The fragment of the system class diagram.

system startup, so the operator can change the robot's behavior according to the operator's remarks and wishes. However, to apply changes the operator needs to restart the system.

H. Actions tracking

The robot also contains the possibility for actions tracking. This feature can be used for software debugging, for intelligence improvement and for the statistical analysis of users' behavior.

Every time kernel receives a message or sends a command, it sends a message about this event to the logging module. The logging module constructs a record and sends this record to a database. The record contains a timestamp, a message originator, a message receiver and all message parameters.

A special tool was developed to comfortably exam the database log. Operators can specify which time, which message originators, which message receivers and which sessions they are interested in examining.

III. TESTING AND RESULTS

A. About the Users

The working system of the RoboBeggart has been recently tested in the entrance hall of a church (Church of Noljaakka, Joensuu, Finland) where people could easily see and hear the robotic appliance. The robot was used during the festivities of Christmas 2004, guaranteeing enough potential users. At four seasonal events, the designers of the system supervised and kept written records of the individual (user of the robot) and group (observers of robotic actions) behavior in relation to the robot. The amount of people

assembling during the four events was between 80 and 150. Consequent analysis of the behaviors of the people and their actions when approaching the robot were conducted. Forms containing various multi-choice questions and qualitative answers were submitted to the users of the robot. Such feedbacks were collected to conduct a qualitative-quantitative analysis. About 20 forms were collected.

```

void RoboKernel.handleMessage( Message msg )
{
    Command response[];
    Response = intelligence.getResponse(msg);
    for (int i in response)
    {
        send(response[i].receiver,
            response[i].command);
    }
}

```

Fig. 5 The scheme of the *handleMessage* method of the Kernel class

From the feedbacks and the observations it is clear that some level of technophobia affected many people when dealing with new and sophisticated appliances like the RoboBeggart. Nevertheless some people that originally were perplexed by the robot got quickly acquainted to it after starting to use it. In some cases in this environment (and in previous evaluations too), the users seemed to experience some sort of enthusiasm and joy after having successfully used the appliance. It is interesting to note that teenagers and youngsters have shown greater interest in exploring the features of RoboBeggart than adults.

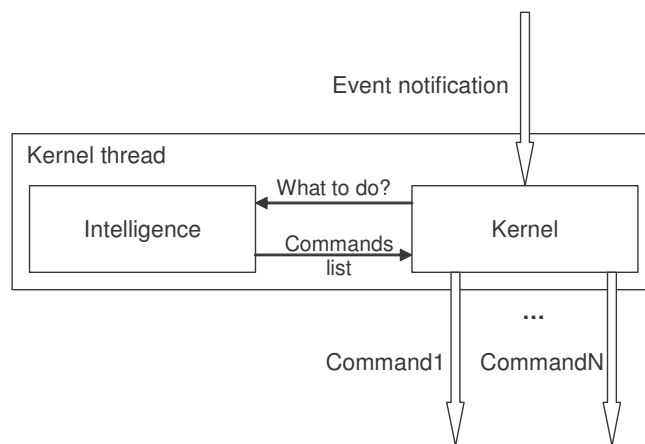


Fig. 6. The kernel behavior scheme

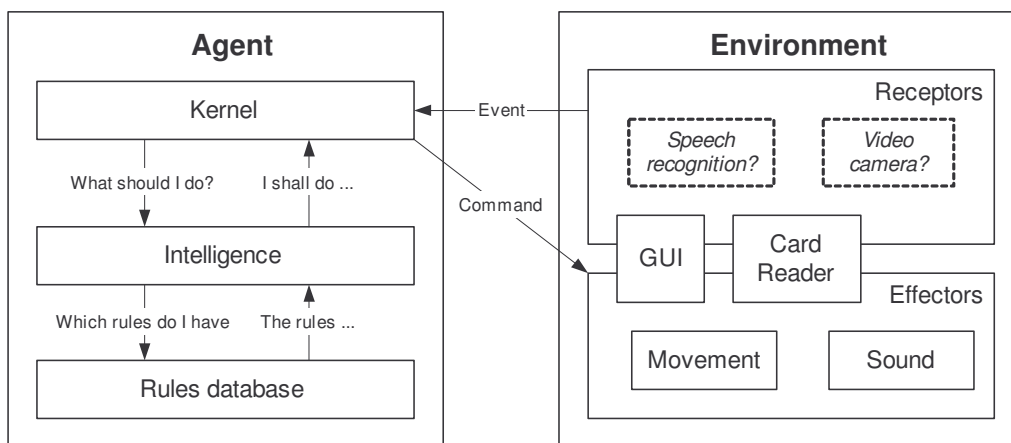


Fig. 7. The robot as an agent

Users have confirmed that the clear instructions found in the robot and the incentives given by the system quickly led to a donation. The quality of the e-modules is fundamental for satisfactory interaction and users have always expressed wishes that the robot would be improved. The robot voice interaction has proven to be relevant for some, but not all, of the users. Many users would like to use the robot in a private or semi-private environment, probably because of the nature of charity fund raising.

In our overall assessment, we can say that the RoboBegger is a functional fundraising appliance and its capacities do constitute a sound platform that, nevertheless, needs to be improved based on users' feedback.

B. About the Operator

On this test occasion, the RoboBegger modules had been organized and prepared by a non-expert operator. One of the partners accepted to design the e-modules and provide the sounds needed to give voice to the robot. After some initial hesitation and some corrections to the robot

management interface, the work started to gain momentum. Only few of the provided elements had to be reworked by the designer of the system and fit back into the robot's system.

The results of the RoboBegger interface were not completely satisfactory, due to the fact that the chosen designer had low graphic and artistic skills. This, obviously, was reflected in the perception of the users. Our conclusions about the requirements for a non expert to be eligible as *perfect* RoboBegger operator, and hence to design the e-modules, are that the operator should (a) have some artistic skills (i.e., have a basic artistic sense for color distribution, spatial picture positioning, choice of image dimensions, etc.), (b) should have some basic understanding of image editing (i.e., how to change file formats and dimensions, how to use filtering, etc.), (c) should have some concept of html editing (i.e., how to create links in the main e-module for the Internet domain and create some html pages on Internet, etc.) and (d) should be capable of synthesizing large amount of data in a few sentences or figures. Given all

of this and enough time to get acquainted with the tutoring interface, the operator would be capable of creating or adapting the e-modules to any need or circumstances. If considering work situations where natural skills are scarce, it is clear that the combination of the above mentioned requirements should be searched and found in two or three individuals instead than in a single candidate. In addition, it seems that the *artistic skills* are more dominant than the technical ones in an operator that would create a successful interfaces.

C. Overall results

According to the tests conducted, we can say that the architecture of the robot is quite successful – it was quite easy to modify the system for the current tasks and needs. There was also some kind of stress-testing performed by children, which the robot passed successfully.

On the other hand, there are some areas for improvement in the current system:

- It was unstable because of intensive memory operation and the usage of Windows 98 operational system. For various reasons it was not possible to use a newer OS, even though the hardware and software could cope and benefit with W2K or higher. To manage and avoid this trouble in the future it will use the Linux operational system.
- The touch-screen of the user interface should be improved. The buttons should be larger and the distance between buttons should also be increased. Tutors will have to take into account this information.
- The current intelligence works satisfactory but it would be advantageous to try other non-static types of intelligence, which could allow, for example, supervised learning.

IV. CONCLUSION

In this phase of improvement of the RoboBeggar, an intelligent system has been used in a XML rules database file, which provided the rules needed for robotic management of components. During the test, it has been proved that this choice has practically eliminated all functional bugs related to the handling of commands. In addition, the robot could be managed by a non expert and its behavior was easily redefined at will. The administrator of the robot could manage, have access to the modules of the robot, modify the visual and aural interface and the physical behavior of the robot with no difficulties.

We expect to have the complete and full management of the robotic parts (except for the bankcard reader) plus the external printer in an easy user interface. Intelligence will also be implemented in terms of the capacity of the robot to “remember” individual users. The problem related to the loading of any new set of rules, which happens during the system start-up, will be modified and separate from it.

Further on we expect to implement both a voice recognition system and a motion recognition system so that the robot will be able to “look” at the user and follow the movements of the user. Intelligent tissues will be used to add expressivity to the face and provide a more realist interaction between the user and the robot.

The next innovation feature for the robot is the self-management that the robot will have. Obviously, as described in the article, the rules will always set the limits for the robotic behavior. However, future self-management will include the ability to choose among a variety of behaviors and keep the best one according a specific user’s behavior. The combination of mastering the vocal modules, the movements and the interface of e-learning modules will grant the robot a wide spectrum of possible interaction tools that will positively affect the quality of human-robot interaction.

REFERENCES

- [1] P. Marwedel, *Embedded System Design*. Springer; 1st edition, October, 2003.
- [2] S.J. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall; 2nd edition, December 2002. pp 46-49
- [3] B. A. MacDonald, T. H. J. Collett, “Novelty Processing in Human Robot Interaction”, in “Novelty Processing”, A One-Day Transdisciplinary Intermedia Symposium, Elam School of Fine Arts, University of Auckland, 7 April 2004. pp. 1-12, 2004.
- [4] La Russa, G. & Marjomaa, E.: "Robo-eLC: A Robotic Adaptive Learning Appliance". CELDA, 2004
- [5] La Russa G., Faggiano E., "Robo-eLC: Enhancing Learning Hypermedia with Robotics". ICALT, 2004
- [6] "Object-oriented analysis and design with applications", Grady Booch, 2nd ed., pub. Benjamin/Cummings, Redwood CA, pp 43-54, 1994.
- [7] "The Java™ Programming Language" Ken Arnold, James Gosling, David Holmes, pub. Addison-Wesley, pp 54-56 2000.
- [8] "Java™ 2: The Complete Reference, Third Edition" Patrick Naughton, Herbert Schildt, pub Osborne/McGraw-Hill, pp 23, 263-299, 1999
- [9] "Java™ 2 for Professional Developers" Mike Morgan, pub SAMS Publishing, pp 159-161, 1999.