Proceedings of the 2003 IEEE
International Conference on Robotics & Automation
Taipei, Taiwan, September 14-19, 2003

A Service-Based Network Architecture for Wearable Robots

Ka Keung Lee, Ping Zhang and Yangsheng Xu

Department of Automation and Computer-Aided Engineering
The Chinese University of Hong Kong
{kklee,ysxu}@acae.cuhk.edu.hk *

Abstract

We are developing a network architecture for our novel robot concept of wearable robot. Wearable robots are mobile information devices capable of supporting remote communication and intelligent interaction between networked entities. In this paper, a service-based wearable robot network architecture that involves extensions to the Jini network model is presented. We will discuss three extensions to the original Jini model. The first extension involves the incorporation of a task coordinator service such that the execution of the services can be managed using a priority queue. The second extension enables the system to automatically push the required service proxy to the client intelligently based on certain system-related conditions. In the third extension, we allow the system to automatically deliver the services based on contextual information. Using a fuzzy-logic-based decision making system, the matching service can determine whether the service should be automatically delivered utilizing the information provided by the service, client, lookup service and context sensors. An application scenario has been implemented to demonstrate the feasibility of this distributed service-based robot architecture.

1 Introduction

Robots can now be found performing tasks in many different application areas. In order to further explore the potential impact of robotic technologies to our quality of life, we are exploring a new robotic niche where robots are wearable, in close proximity to humans, and are able to accompany us wherever we go. The wearable robot is an interface through which its user can interact with other people and other networked machines. It helps to extend the users' sensing and actuating capabilities through the networking environment, as well as getting access to useful information.

We are on our way to establish the methodologies for developing distributed wearable robot systems, which can be used to support these mobile devices for communication and interaction [10] (Figure 1). Our research will provide a framework for constructing distributed robot infrastructure in which each robot is a portable communication interface through which remote interactions between humans, robots and machines in the networked environment can be achieved. In this paper, we will concentrate on the issues related to the design and development of a network architecture that is capable of supporting the interaction and communication between the wearable robots and other networked entities, while taking the environmental context information into account.

Figure 1: *Prototype of wearable robot*

The rest of the paper is organized as follows. In section 2, we will present an overview of the service-based wearable robot architecture and the Jini networking model. In section 3, we propose three types of extensions to the Jini networking model. In our architecture, a special component called the matching service is responsible for the activation of networked services, and it will be explained in section 4. The wearable robot architecture is applied in an application scenario and it will be presented in section 5. Section 6 contains the conclusion of this work.

2 Distributed Service-based Architecture

We have developed a service-based architecture to support the communication within the distributed wearable robot system. Service-based architecture breaks the tight coupling between components by

*This work was partially supported by the Hong Kong Grant Council under grants RGC 3/98C and CUHK 4317/02E, and Innovation and Technology Fund ITS/140/01.

providing a higher abstraction layer called service and this architecture model has started to attract much attention in recent years [4] [1] [8]. Using a service-based architecture, our distributed robot system can enjoy a number of advantages offered by object technology [9]. Firstly, it permits the creation of service in such a way that its user may be another service and not necessarily people. The resources used by a service can also be provided by another service. This makes complex interactions between the robots and other networked machines possible. Secondly, it allows service reusability by permitting the specification of new services relying on existing specifications. The facilities provided by existing services can be used to create new services. Thirdly, integrated design of management and control can be supported by encapsulating the operations and management functions into distinct interfaces of the same object.

In a service-based architecture, typical interaction usually occurs between the service and the client. The service basically publishes itself by providing a general and high level description of its location, category and functions it provides, etc. Also it will reveal the technical details of its network location through which it can be connected. On the client's side, it needs to determine the service it would like to use and enquire the network about it. After obtaining a reference to the service implementation, the client can invoke the service.

In our implementation, we apply a Jini-based architecture to support the distributed communication. Jini [3] is a middleware whose purpose is to federate groups of devices and software components into a single, dynamic distributed system to provide simplicity of access, ease of administration, and support for sharing. The service-based architecture utilizing Jini can provide a number of advantages to the wearable robot system. First, we want to develop a network system in which the services, including robot, machines and other functional devices, can find each other automatically. Jini helps by providing a lookup service that can let any networked member to identify and get access to a service by specifying the desired functions. Second, it is very desirable that the components can be added into the network and removed from the network without affecting other components. During the operation, the communication relationships between the services and components may change. The components in the service-based architecture are loosely coupled and network stability can be improved by Jini by ensuring such change to be carried out seamlessly. Third, the wearable robots are portable and their locations may change as the users move around. The wearable robots can be service providers or clients. As service receivers, the services that are delivered to the wearable robots may need to change the destinations as the robots move. As service providers, the positions from where the robots provide their services may also change as the they move. By utilizing service stub mobility, the Jini system allows mobile services to be accessed by the clients in the network.

3 Extension to the Jini Model

The original service delivery model is basically a pull model in which the clients request the services and the service proxies are delivered to them if the services are available. In order to handle the issues related to efficient resources allocation and context-aware applications in our wearable robot system, we develop three types of extension to the original Jini network architecture. The extensions involve the introduction of two new services - the matching service and the task coordinator service. The first extension is an enhancement of the pull model and the second and third extensions are based on the service push model.

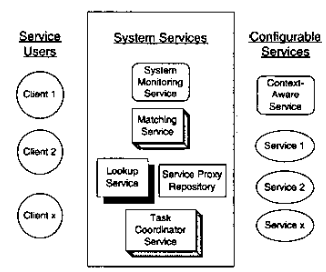

Figure 2: *Components in the service-based architecture*

The matching service decides whether a service should be provided to a particular client, based on a fuzzy decision making process influenced by the conditions of the service provider, the client, the network system and the environmental context information. When the system starts up, the services will form a federation by registering their presence by transmitting their proxies to the lookup service. Any new service can join the federation dynamically after the lookup service has started up. The clients who wish to utilize the services in the federation will need to contact the lookup service.

There are three types of networked entities in our design of the service-based architecture. They are the system services, the configurable services and the service users (clients). The system services include the lookup service, the system monitoring service, the matching service and the task coordinator service. The task coordinator service contains a queuing system that manages the execution of the service tasks. Using fuzzy logic, the matching service determines the priority levels of the service tasks to be executed. The system monitoring service mainly keeps track of the number of running services in the system and the number of tasks in the queues. For context-aware applications, context-aware services are used to manage the context related sensor data. We are going to explain the extensions below.

3.1 Extension 1: The Addition of Task Coordinator Service

The limited resources in distributed systems affect the provision of services to clients from time to time. The task coordinator service provides coordination between all tasks by considering their content and priority. In order to meet the requirement of the quality of service in the system, a queuing mechanism is implemented in the task coordinator such that the resources in the system can be controlled. After a client has downloaded the proxy of a service provider from the service proxy repository, it has to notify the task coordinator before the service can be delivered. The steps are shown in Figure 3 and Table 1.

The task coordinator service contains a first-in-first-out queuing system which allows task reordering by priority and it coordinates the execution order of the services using priority queues. For services having equal priority levels, they are executed in a first-in-first-out manner. The task having the highest priority in the sequence is executed first (descending order) and there is no assumption on how elements enter the priority queue, but only a criterion for their exit. There is one queue for each service provided by each server and the queues are implemented using a heap data structure. The services will notify the task coordinator service when they are available for delivering new services and the priority levels of the services are determined by the matching service using fuzzy logic.

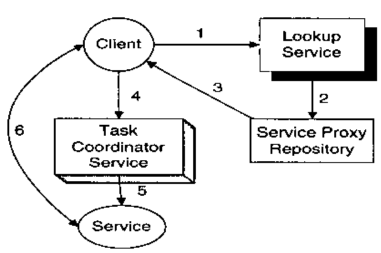

Figure 3: Incorporation of the task coordinator service to the original Jini model

3.2 Extension 2 : Automatic Service Proxy Download

If the system is to deliver services efficiently in applications, not only the users can pull the required service for utilization, but more importantly, the system should be able to intelligently push the services to the users when certain conditions are met. In this extension, the system is equipped with a decision making service, called the matching service, which monitors the relationship between the service provider, the client, the network system and the environmental context. The aim is that when the matching service has reached the conclusion that the collective behavior of all theses parties in the system have come to a certain state, the proxy of the suitable service will be pushed to the clients who need them. The steps are shown in Figure 4 and Table 2.

Step	*From*	*To*	*Action*
1	Client	Lookup Service	Client discovers and joins the lookup service
2	Lookup Service	Service Proxy Repository	Request Service from SPR
3	Service Proxy Repository	Client	Proxy downloaded from Web Server
4	Client	Task Coordinator Service	Send task object to task coordinator, joins the queue and waits for service task execution
5	Task Coordinator	Service	When the service is ready for use, the task of the client will be sent to the service
6	Client	Service	Service is delivered to client

Table 1: Steps involving the task coordinator service

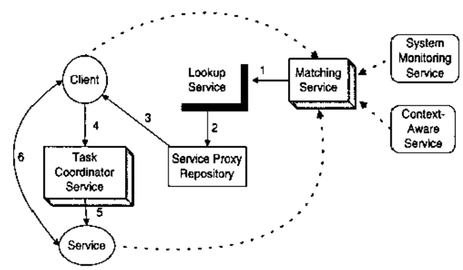

Figure 4: Pushing the service proxy automatically to the client based on processing result of the matching service

3.3 Extension 3: Automatic Service Delivery

Our third extension to the Jini architecture is also based on a push service model. For certain services, there may not be particular clients that the proxies are required to be delivered to. Instead, the services can be directly performed without establishing links between the client components and the service providers. This happens when the client of the service is a human, but not a network component. The automatic delivery of the services is decided upon the condition of the service provider, the system status and the contextual data obtained from sensors. In this case, the matching service does not require input from client component.

Step	From	To	Action
1	Matching Service	Lookup Service	Lookup service receives information from the matching service
2	Lookup Service	Service Proxy Repository	Request Service from SPR
3	Service Proxy Repository	Client	Proxy pushed to client
4	Client	Task Coordinator	Send task object to task coordinator, joins the queue and waits for service task execution
5	Task Coordinator Service	Service	When the service is ready for use, the task of the client will be sent to the service
6	Client	Service	Service is delivered to client

Table 2: *Steps for automatic service proxy push*

The steps are shown in Figure 5 and Table 3.

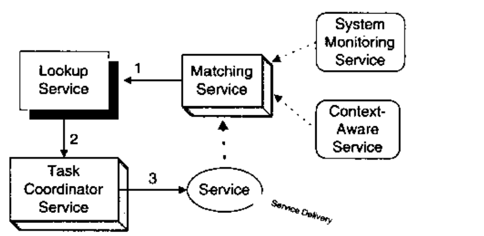

Figure 5: *Automatic delivery of service based on processing result of the matching service*

Step	From	To	Action
1	Matching Service	Lookup Service	Lookup service receives information from the matching service
2	Lookup Service	Task Coordinator Service	Service task is pushed to task coordinator service
3	Task Coordinator Service	Service	When the service is ready for use, the service will be automatically delivered

Table 3: *Steps for automatic service delivery*

4 The Matching Service

We apply fuzzy logic to analyze the collective behavior of the services provided by the wearable robots and the components in the network system. Fuzzy logic is a suitable technique to be applied in the matching service for making decision relating to service delivery because of a number of reasons. First, an accurate state of the global network is very difficult to update and maintain. In a dynamic network, the global state will always be imprecise and this imprecision will only increase as more interacting components are added into the federation. Second, good and realistic quantization can be provided by the fuzzy states, since the system and human related features only have limited resolution. Unrealistic result may occur if computation is made using single values or sharp intervals. Third, our network takes context information into account when making the decisions related to automatic service delivery. Context information is usually described using linguistic terms. For example, when we express the context of relative position between two objects, we often describe them as being 'close', 'very close' or 'not too close', etc, which are rather fuzzy.

The purpose of the matching service is that after observing the interaction between the clients and the service provider, it can abstract the relationship between the system context and the delivery of services, such that when the appropriate conditions are met, the suitable services will be automatically delivered to the clients.

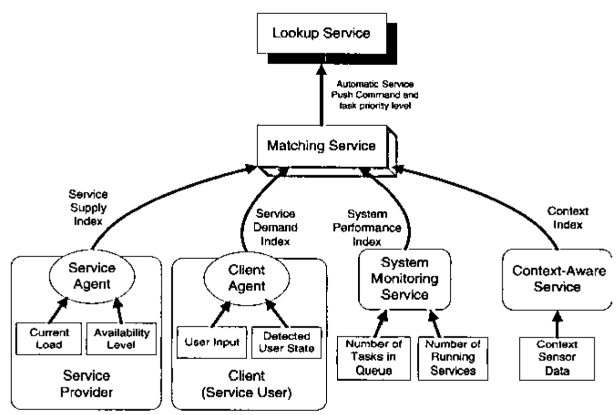

Figure 6: *The relationship between the matching service and other components in the federation*

Four different indices are fed into the matching service for making decisions related to service push, which include the service supply index, the service demand index, the system performance index and the context index. The matching service is a decision making engine based on fuzzy logic. The context index is calculated by the context-aware service based on the context sensor data. The context-aware service provides the functions of data abstraction and separation of concern. For simple cases, the context-aware service can perform a straight forward scaling operation on the sensor data for normalization. For more complex context types, more sophisticated mechanisms such as those provided in the Context Toolkit [2] can be used to help handling the process. The service supply index, the service demand index and the system performance index are generated by fuzzy engines based on two inputs each. The system monitoring service observes the number of services running in the system and monitors the number of tasks in the queue inside the task coordinator service. The service demand index is generated by a client agent residing in the client. The client agent keeps track of the state of the user (by using intelligent user interfaces) and receives direct input from the user indicating the level of desire for a particular service. The service supply index is also generated by a service agent residing within the service provider. By detecting the relationship between the current load of the service provider and how much the service would like to make itself available to the clients, the service agent determines the service supply index of this service. When the matching service receives a new non-low index from the client agent or the context-aware service, the matching service will request the indices from the service agent and the system monitoring agent. The decision making process will start after the three required indices are all obtained and two of which must be the system index and the service supply index.

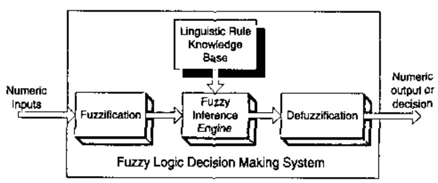

Figure 7: Fuzzy decision process

The basic rationale behind the fuzzy rule base is as follows:

1. The service supply index will be higher if the service has a higher availability or a lower load.

2. The service demand index will be higher if the client has a higher detected need or a higher input desire to use the service. The need of the user can be detected by an intelligent user interface.

3. The system index will be higher if the number of items in the queue is lower or the number of running services is lower.

4. If enough number of input indices are sufficiently high, the service proxy will be automatically pushed to the client or the service will be automatically delivered. If the indices are too low, no action will be taken by the lookup service. If the indices are close to middle value, the user will be prompted for making the decision.

5 Application Scenario

For the purpose of demonstrating the usefulness and feasibility of the service-based architecture for networked wearable robots, we have implemented an application system in our research lab. Let's consider a scenario situated in an office room and a lab which utilizes the wearable robot and the service-based architecture for context-aware and communication purposes (Figure 8). In this application, we use a visual person tracking system to detect the presence of visitor in the lab and utilize this location context information to determine which service to activate. Apart from the visual person tracking system, the lab is equipped with two voice communication service providers (A and B), and a humanoid robot (Figure 8). If the visitor approaches the humanoid, it will present a greeting message to the visitor. If the visitor leaves the lab, the humanoid will present a farewell message. The master of the wearable robot, who is working in another office room, will be notified by the wearable robot if the visitor moves close to one of the voice communication service providers. A voice communication link can then be established between the wearable robot's master and the visitor in the other room through microphones. If abnormal behavior is displayed by the visitor, such as quick running and moving in the lab or being stationary in the same position for a long time, the wearable robot user in the office room will also be notified and he/she can request to take control over the motion of the humanoid robot for a period of time using the interface on the wearable robot. The flowchart of this application scenarios is shown in Figure 9.

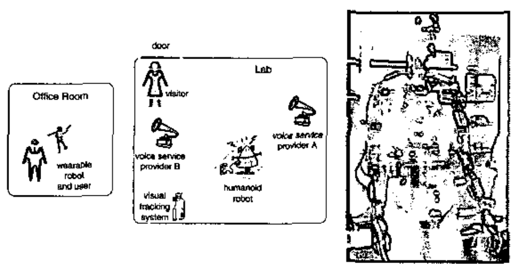

Figure 8: Setup of the two rooms in the application and the humanoid used

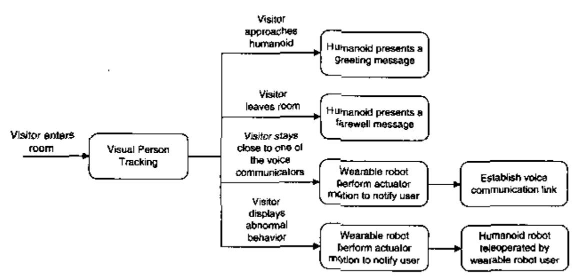

Figure 9: Flow chart of the sample application

The visual person tracking system, the voice communication gateway, the humanoid action controller and the wearable robot action controller are implemented as Jini services. The person tracking system can determine the position of the visitor in the scene using background subtraction technique [5] and the motion speed can be calculated. The visual system can also detect if the visitor is displaying a high degree of motion by calculating the rate of change of the rectangular blob shape and size. The context index is directly related to the person's position. There is one context index related to each specified position in the lab, including the areas close to the door, the humanoid robot and the two voice communication service providers. For example, the position context index related to the humanoid robot would give a higher value if the person is closer to the humanoid robot.

There is one entry in the matching service for each service. The input configuration for each matching service entry is designed before service registration. The input of the matching service must include the service supply index and the system performance index. The service demand index and the context index are optional, but at least one of the two must be present. Depending on the design of each individual service, it can be triggered by the client or by contextual information change in the environment.

The context information service provided by the visual tracking system is a continuous sensing service and is always on. The humanoid teleoperation service allows its client to take control over the motion of the humanoid robot for a certain period of time. These two services are provided to any capable clients that request for it and is not dependent on any other context information for activation. The humanoid farewell service and the humanoid greeting service will be automatically delivered if the matching service considers that the required conditions are met and these two services do not require client information. The wearable robot user notification service allows the wearable robot to be notified by the person tracking system. The context index of its matching service will be high if the visual tracking system finds that the visitor stays in front of the humanoid or abnormal human behavior is displayed by the visitor. The voice communication service allows us to experiment with the effect of the automatic service proxy push mechanism. Depending on which voice communication service provider (A or B) the visitor is staying close to, the corresponding service proxy will be pushed to the wearable robot to allow both users to establish a communication link.

6 Conclusion

In this paper, we have introduced the architecture for the novel concept of wearable robot. A serviced-based architecture using Jini can enable the flexible and reconfigurable connection between the interacting components in the distributed network. We have extended the original Jini network model in three different ways which allow the system to prioritize the tasks using priority queues in the task coordinator service, automatically download proxies to the relevant client based on the conditions of related parties, and automatically deliver service based on contextual information. The usefulness of the architecture has been demonstrated in a sample application scenario.

References

[1] R. Agrawal, R. Bayardo, D. Gruhl, and S. Paparimitriou, "Vinci: A service-oriented architecture for rapid development of web applications", in *Proc. 10th Int. World-Wide Web Conf.*, Hong Kong, May 2001

[2] A. Dey, "Providing architectural support for building context-aware applications", PhD Thesis, Georgia Institute of Technology, 2000

[3] W. K. Edwards, *Core Jini*, The Sun Microsystems Press, 2001

[4] D. W. Forslund, et al, "The importance of distributed, component-based healthcare information systems: the role of a service-based architecture", in *Proc. 14th IEEE Symp. on Computer-Based Medical Systems*, pp. 79-82, 2001

[5] D. M. Gavrila, "The visual analysis of human movement: a survey", *Computer Vision and Image Understanding*, vol. 73, no. 1, pp. 82-98, Jan. 1999

[6] S. Ghosh and Q. Razouqi,"A survey of recent advances in fuzzy logic in telecommunication networks and new challenges", IEEE Trans. on Fuzzy Systems, vol. 6, no. 3, pp. 433-447, Aug. 1998

[7] K. Hinckley, et al, "Sensing techniques for mobile interaction", in *Proc. of ACM User Interface Software and Technology, CHI Letters*, vol. 2, pp. 91-100, 2000

[8] R. D. Schimakat, W. Blochinger and C. Sinz,"A Service-based agent framework for distributed symbolic computation", in *Proc. 8th Int. Conf. on High Performance Computing and Networking, (HPCN Europe 2000)*, Amsterdam, The Netherlands, pp. 644-656, 2000

[9] A. Trigila, K. Raaiikainen, B. Wind and P. Reynolds, "Mobility in long-term service architectures and distributed platforms", *IEEE Personal Communications*, vol. 5 Issue: 4 , pp. 44-55, Aug. 1998

[10] Y. Xu, K. K. Lee and P. Zhang, "Wearable Robots as Mobile Devices for Intelligent Communication and Interaction", Submitted to the IEEE Int. Conf. on Robotics and Automation, Taipei, ICRA 2003