# Implementation of the Distributed System for Data Acquisition, Processing and Analysis

Vladimir V. Kryukov
Computer Technology and System Department,
Vladivostok State University, Russia
kryukov@vvsu.ru

Carina J.Shakheldyan
Computer Technology and System Department,
Vladivostok State University, Russia
carinash@vvsu.ru

## Abstract

*This article describes the design and some results of implementation of a distributed computing data acquisition and processing system being developed by the Computer technology and system department of Vladivostok State University (Russian Federation). The system is designed to provide data acquisition, processing, and storage in distributed heterogeneous environments. Some of the system's components meet demands of soft real time mode.*

## Keywords

Distributed system, data acquisition, digital signal processing.

## 1. Introduction

An important characteristic of modern computer systems is that they are heterogeneous [1]. Various operating systems (Unix, Windows) and hardware (PC, Sun) are used to design systems for scientific research and education.

The long-term goal of the developed distributed data acquisition and processing system (DDAPS) is to provide large-scale, reliable, high-speed, multi-user environment for data acquisition, processing, analysis, visualization and storage. The system must be capable of handling large number of simultaneously connected clients and must use resources of heterogeneous environment. Besides, clients should be able to join the system through Internet.

To achieve this goal a number of tasks are to be solved:

1. to acquire data in real time mode using a dedicated data acquisition node;
2. to preprocess data in soft real time with digital signal processors and without them;
3. to process and analyze data by distributed computing;
4. to support centralized data storage and multi-user control across local net and Internet;
5. to control load balancing;
6. to provide co-operation between Windows, Solaris, Linux and OS/400 applications;
7. to provide generating reports.

DDAPS is intended to carry out data acquisition, data processing, data storage, and data visualization regardless of computer architecture or operating system used. The tasks may be executed using different operating systems and architectures. Three aspects are important:

1. to provide soft real-time mode for time critical applications;
2. to provide high performance for post-processing and analysis;
3. to provide reliable data storage.

The system will be used for educational and scientific purposes, therefore frequent updates of system architecture and algorithms are assumed. Several approaches to design the distributed system were considered. The component models both DCOM [2] and CORBA are used for the design of the DDAPS.

So far we have known the only middleware providing real-time specifications for distributed systems. It is TAO [3]. TAO is a high-performance, real-time CORBA ORB endsystem.

Thus DDAPS is developed using DCOM and TAO CORBA.

## 2. DDAPS specification

### 2.1 DDAPS architecture

Though the system operates in the heterogeneous environment, clients are PCs running Windows 95/98/NT. PC Linux and Sun Solaris will be considered as the clients in future. The data acquisition node runs Windows NT. Windows NT is to be used as the workstation operating system for time-critical digital signal preprocessing.

Several multi- or single-processor computers are intended for data postprocessing employing multiple algorithms of digital signal processing: spectral Fourier analysis, spectral parameter estimations, filtering, wavelet analysis, zero-crossing, denoise, signal compression, restoration, and simulation. These computers run Windows NT, Linux or Solaris. At present two processor Sun Ultra Enterprise 450, two processor PC workstation, Sun Ultra Sparc 10 and several PC desktops are involved.

Some nodes of the system are application servers operating as the middle tiers for transaction control and storing/fetching data to/from the database. The application servers provide interface between the clients and the database. Five Sun Ultra Sparc 5 are used as the application servers.

Oracle database server resided on Sun Ultra Enterprise 450, being reliable and multi-user system for data storage, is used in DDAPS. It operates under Sun Solaris 2.7. We consider possibility to use MS SQL Server in future.

There is one node for storing research reports; it is AS/400 with Lotus Notes. AS/400 DB2 database is considered for possible use in future along with Oracle and MS SQL Server.

The network infrastructure of DDAPS is based on Fast Ethernet.

### 2.1. The component approach

DDAPS may be considered consisting of dedicated (specialized) components. A component is a software code implemented as an object that meets demands of DCOM or CORBA specifications. The components have standard interface that allows component substitution without re-coding.

We will call a software component an agent if it is attached to a network computer node. One component may be picked up by several nodes.

There are three component groups: manager group, administration group, and work group.

The manager group consists of the manager components including task manager, data acquisition manager, processing manager, data storage manager, data analysis manager, data visualization manager, and report manager.

The administration group provides user access to DDAPS resource, DDAPS administration including user permissions and component performance evaluation, and search for a computer to run a component.

The work group comprises components carrying out the base work of DDAPS. In this group one component solves one problem. For example, there are components for data acquisition, signal storage, mean removing, fast Fourier transform (FFT), autoregressive parameters calculating with Burg algorithm, and so on. These are simple components. A combined component aggregates a number of simple components. There is, for example, a combined component for data filtering comprising data acquisition and storage, FFT, inverse FFT, multiplication, and addition signal components. A combined component for spectral analysis with periodogram method consists of data acquisition and result storage, mean removing, FFT, windowing, accumulating spectrum, and scaling components.

The output from one component of the work group is the input to another components of the same group. The output from data acquisition components is the input to the FFT, mean removing or data storage components. The same output from a component can be fed to several other components’ inputs.

However the implementation of that is not direct. There are data transmission components, very important for reliability and assuring soft real-time mode. These components are in the work group. Their purpose is to supply or consume data and events as soon as data block is ready to be transmitted.

### 2.2. The work group components

The work group components operate some data and results of its processing. They are divided into six base parts:

1. the components producing data - data acquisition and simulation;
2. the components processing data - algorithms and methods for digital signal and image processing and other;
3. the components analyzing results of data processing;
4. the components reporting research activities;
5. the components consuming data - data storage and data visualization;
6. the components transmitting data.

All these components either produce or consume or transmit data. Usually there are one data source (sender) and several data receivers. The data senders are data producing and processing components. The data receivers are the data storage components, the data visualization components, and the processing components.

The CORBA Event Service provides a flexible model for asynchronous communication among objects. However, the standard CORBA Event Service specification lacks important features required by real-time applications. The Real - Time (RT) Event Service [3] supports asynchronous message delivery and allows one or more suppliers to send messages to one or more consumers. The consumers and suppliers may specify their execution requirements using QoS parameters. Suppliers push events to the Event Channel, which in turn pushes the events to consumers. RT Event service supports periodic event processing, event filtering and correlation, and prioritized dispatching.

We use RT Event Service to develop data transmission components (consumer and supplier) for time critical applications. TAO supports both Solaris and Windows therefore developed components are compatible on the source code level.

Consumer components specify type and some details of each event it is interested in receiving. Consumer provides the worst case and expected execution times for receiving data. For example, components willing to receive data from the data acquisition board must specify the worst case and expected execution times depending on the sampling frequency and the buffer size of the acquired signal.

In order to prevent data loss consumer components should specify high criticality of data reception. Data processing, data storage, and data visualization components use consumer components.

Supplier components specify types of each event and maximum rate at which it will generate events of each type. Supplier component that sends data from signal acquisition board specifies maximum rate depending on the sampling frequency and the buffer size of the acquired signal. Data acquisition and simulation components use supplier components.

The consumer and supplier roles may be combined. The most of the processing components use combined consumer/supplier components. These components generate event only after they have received certain event and processed the data. These components specify dependencies between events. For example, ‘data ready’ event requires ‘data received’ event having occurred and subsequent data processing.

Applications that do not need real-time mode may enable another transmission components not using RT Event Service. Such components support standard CORBA Event Service.

## 2.3. The manager components

The system has a number of the management components. The main one is the task manager. The task manager is a control component of DDAPS. It determines what components are needed to do a required task; it determines where the components must be run.

All tasks of the system can be divided into a few types:

1. data acquisition;
2. data processing;
3. data storage;
4. data analysis;
5. data visualization;
6. reports;
7. administration.

The dedicated management components control the task of a certain type. The data acquisition manager component manages data acquisition from different sources including signal acquisition board. The processing manager component manages data processing including different algorithms and methods of signal and image processing.

The data storage manager operates data storage to different destinations including a database, file system, or data warehouse.

The data analysis manager carries out analysis of data processing results and makes conclusions about different data characteristics.

The data visualization manager operates data presentation on the client side.

The report manager provides reporting of research activities.

The task manager decides what manager components should be used to carry out a task put by user. The every of selected manager components make resolve what components should be used to carry out problem put by task manager.

## 2.4. The administration components

The administration group includes several components: security, system administration, and scheduler components.

The security component allows checking access permissions for a user. The security manager checks access permissions for the user. The security manager issues a database query to identify information resources available for this user. Different users have different access permissions to the components, data, results of processing, and the computers. The access component controls access to the system services.

The system administrator is a management component for DDAPS administration. The responsibility of the component is to match between a component and computers to convert the component into the agent and to collect the statistics of execution time of the different components on the different computers. These statistics are saved into DDAPS database.

The scheduler analyzes the statistics collected by the system administrator and the computer resources to notify the task manager and another managers what component will be executed on what computer. In order to identify the computer scheduler checks whether the considered component attached to a computer satisfies real-time demands. If a computer is suitable, then scheduler returns the computer identifier otherwise it checks another computers from available computers for the component. If there is no available computer then either it real-time restriction should be loosed (for example, by reducing the sampling frequency) or the component should be attached to a computer manually. In the latter case one have to cope with data loss or use some load balancing procedures.

Consider how the scheduler selects computer to process data in detail. Signal processing in real-time mode has a number of features.

1. Data block is produced periodically in a priori known time interval.
2. Data processing is carried out in frames. We assume processing be sequential within a frame. Of course, processing is not always sequential. However we use sequential processing within a frame in the existing version of DDAPS.
3. Frames may and must be processed in parallel.
4. We assume data transmit time be negligible.

The basic task of the scheduler is to load the most productive computers as much as possible. The scheduler selects the most powerful computer and tries to attach the longest processing component to it. If it is impossible, the scheduler assumes that real-time restrictions are not met. If the scheduler has attached the component to the computer, then it selects the longest component from the remaining and tries to attach it to the most productive computer. If it is impossible, the scheduler does the following steps.

1. If no component is attached to the computer, the scheduler assumes real-time demands are not met.
2. If some components are attached to the computer, the scheduler selects next powerful computer and tries to attach the component to it.

Thus the scheduler provide static distribution of components between computers of DDAPS.

The main components of DDAPS are mapped on fig. 1.

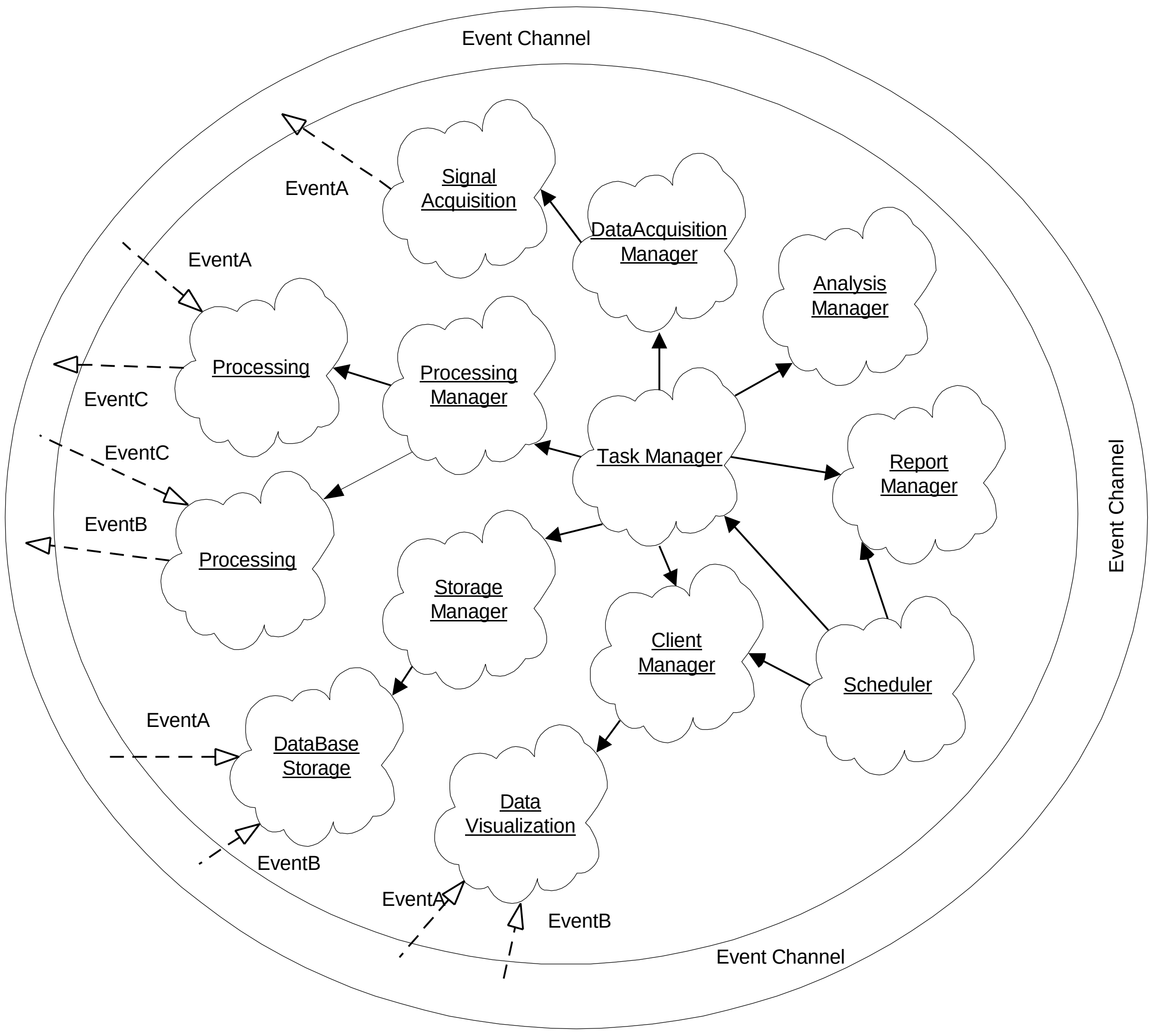

Figure 1. The main DDAPS components

The data acquisition component plays role of a pure supplier. It generates events and passes them into the event channel. The data acquisition component specifies the type of event it will generate (EventA). The component specifies maximum rate at which it will generate the event, criticality, and importance as well.

Signal processing may be implemented with several components. The processing components are a combined consumer/supplier. They produce EventC event as a certain part of processing has been carried out.

The data storage and the data visualization components are consumers and specify the events they are interested in receiving (EventA and EventB).

In a real-time system events must be processed so that consumers can meet their QoS deadlines. All that receive events produced by the data acquisition component must process the signal block before next event is produced.

It means that the processing, storage, and visualization components must receive signal and results of its processing at the same rate that signal is acquired.

The event, which type is EventA, is produced by the data acquisition component as soon as the data acquisition board acquires a signal block.

The event, which type is EventB, is produced by the processing component as soon as processing has been completed.

## 2.5. Logical model of DDAPS

The main logical unit of DDAPS is a *task*. A task's goal is to solve a need specified by a client. For example, there is a need to acquire data, filter it out, and store it into database. The filtered data is to be displayed on the client's computer.

The task intended to solve this need is defined on the client side. A researcher (client) composes the *task* from several work group components (combined and simple) using dedicated software. For the above example, there are four components:

1. data acquisition (simple component);
2. data filtering (combined component);
3. data storage (simple component);
4. data display (simple component).

The client sets acquisition, processing and display parameters such as sampling frequency, a number of channels, filter impulse response. The client transfers the defined task to the task manager. The task manager examines the task and determines manager components to carry out the task. In this case these are data acquisition manager, processing manager, data storage manager, and data visualization manager.

Each of the managers looks for work group components to carry out its part of the whole task. The data acquisition manager has found signal acquisition component and converts it to a data acquisition agent using the scheduler component. The processing manager has found filtering component and converts it to a filtering agent using the scheduler. The data visualization manager has found signal visualization component to display filtered signal. In this case to convert component to agent the scheduler is not needed. Data storage manager has found database storage component and converts to an agent using the scheduler.

The task manager details the *task* by adding the information about the basic subtasks (acquisition, processing, etc.).

The subtask managers detail their subtasks by adding the information about agents, that render services needed for carrying out the task.

As soon as the task detalization is complete, the subtask managers create and initialize necessary agents.

A client is not a part of the *task* but it can join an existing *task* or create a new *task*. The task consists of agents. The *task* can be saved into database. The signals, results of signal processing, user information, component information, computer information, statistical information are stored into the database too.

The task manager will check all the available addresses and will try to find necessary agents.

Those components that use hardware (data acquisition or analogous filter components) have the static location only. Other components have dynamic addresses except the security manager. The security manager has static IP address and every client of the system must know this address.

The task is launched as soon as the agents have been connected and initialized. While the task is running, other clients may connect to it.

## 3. Implementation

So far several parts of DDAPS have been developed:

- data acquisition component;
- analogous filter and gain programming component;
- signal processing component employing Short-Time Fourier Transform (STFT), FFT, mean removing, windowing, autoregressive parametric algorithms: Burg, covariance, minimum dispersion, multichannel autoregressive estimation including covariance, Nattoll method and minimum dispersion;
- CORBA component for querying Oracle database residing on Sun (Solaris);
- data transmission CORBA component based on RT Event Service;
- signal and spectrum visualization components;
- Win32 client;
- the simplest implementation of the scheduler component.

The architecture of the developed DDAPS is shown on fig. 2.

## 4. Conclusion

We are currently deploying the system to provide gap-free data acquisition on the centralized node, to process signals, and to store data into database. We are developing various parts of the system. The system described in the article uses RT Event Service to achieve high performance and scalability, it uses CORBA and DCOM to provide the flexibility needed for distributed data acquisition and processing system.

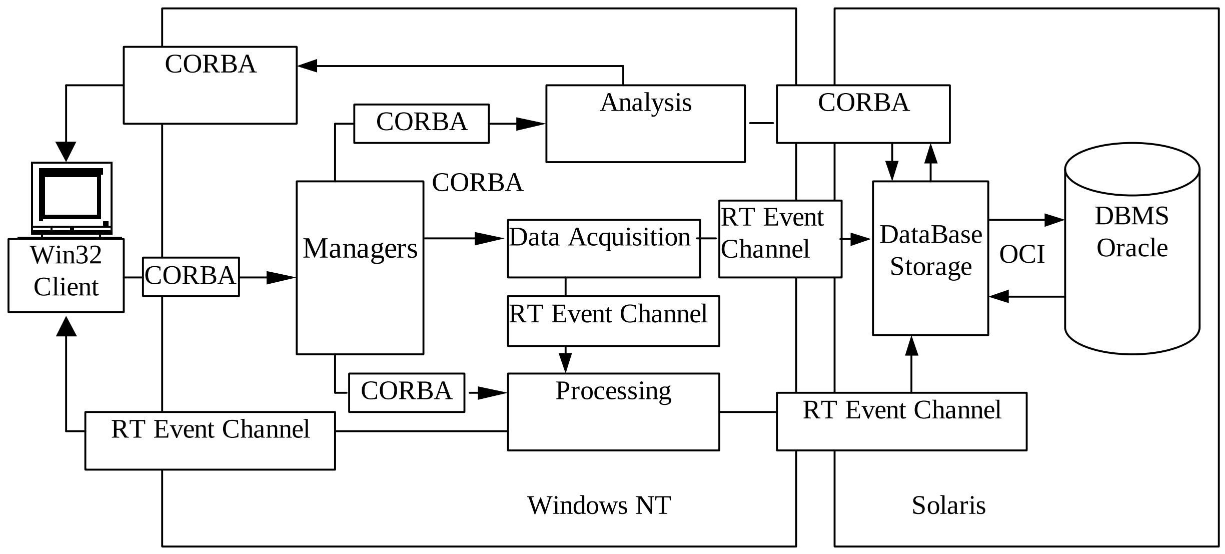

Figure 2. Architecture of the developed DDASP

The forthcoming steps will be aimed at

- design of the manager components;
- design of the scalable high-performance system to provide digital signal processing using digital signal processors in soft real time mode;
- load balancing control;
- test of the system components.

## References

[1] S. Vinoski. CORBA: Integrating diverse applications within distributed heterogeneous environments, *IEEE Communications*, vol. 14, no. 2, Feb. 1997

[2] D.Rogerson. Inside COM. Microsoft Press. 1997.

[3] TAO ORB and CORBA Service Tutorials

http://www.cs.wustl.edu/~schmidt/TAO