DISTRIBUTED SYSTEM FOR COOKING OIL BOTTLE PRODUCTION FACTORY

Lwin Mar Oo, Khin Myat Myat Moe
University of Computer Studies (Magway)
pphyo437@gmail.com

Abstract

Recent advances in distributed system architectures may provide the solution to a number of challenges in industrial systems. Middleware and grid technologies make it possible to utilize the processing power in subsystems, so that greater reliability through fault-tolerance can be achieved, as well as efficient use of resources through load-sharing. By using principles from distributed computing it is possible to achieve fault-tolerance, and thus increase the reliability of applications. For industrial systems, such as in a chemical plant or an oil rig, there may be from tens to tens of thousands of different processing units in operation, and these are becoming more powerful and multi-purpose. In this paper, Machine A processes the filling bottles. After filled one bottle, Machine A sends message filled one bottle to Machine B that processes the packing bottles. Machine B receives the message and then Machine B processes the packing bottle. Machine B messages the packed bottle after packing bottle. This paper is implemented by JAVA programming language.

1. Introduction

A distributed system is an application that executes a collection of protocols to coordinate the actions of multiple processes on a network, such that all components cooperate together to perform a single or small set of related tasks. A distributed system is one in which hardware or software components located at networked computers communicated and coordinate only by passing message. The characteristics of distributed systems are concurrency of components, lock of a global clock and independent failures of components.

The sharing of resources is a main motivation for constructing distributed systems. Resources may be managed by servers and accessed by clients or they may be encapsulated as objects and accessed by other client objects [1].

This paper presents the distributed system for cooking oil bottle production factory. In this paper, there are two interfaces: filling machine and packing machine according to distributed system. Processes in a distributed system communicate by sending message from filling machine to packing machine and receiving messages at packing machine.

2. Distributed System

A distributed system is one in which components located at networked computers communicate and coordinate their actions only by passing messages. Networks of computers are everywhere. The internet is one, as are the many networks of which it is composed. Mobile phone networks, corporate networks, factory networks, campus networks, home networks, in-car networks, all of these, both separately and in combination, share the essential characteristics that make them relevant subjects for study under the heading distributed systems [1].

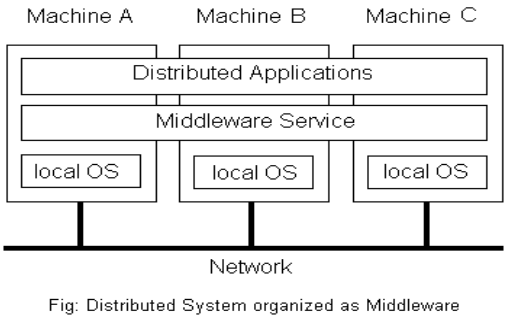

Fig: Distributed System organized as Middleware

Figure 1. Distributed System Organized as Middleware

Middleware is presented by processes or objects in a set of computers that interact with each other to implement communication and resource-sharing support for distributed applications. Middleware is concerned with providing useful building blocks for construction of software components that can work with one another in a distributed system. Applications and services use the middleware layer for their interactions. Below the middleware layer is the operating system layer. Middleware runs on a variety of OS-hardware combinations at the nodes of a distributed system.

The architecture of OS for a distributed system:

- Run only that system software at each Computer
- Allow the software implementing any particular service to be changed independently of other facilities.
- Allow for alternatives of the same service to be provided, when this is required to suit different users or applications.
- Introduce new service the integrity of existing ones [1].

3. Examples of Distributed System

Examples are based on familiar and widely used computer networks: the Internet, intranets and the emerging technology of networks based on mobile devices. They are designed to exemplify the wide range of services and applications that are supported by computer networks [1].

3.1 The Internet

The internet is a vast interconnected collection of compute networks of many different types. Programs running on the computers connected to it interact by passing messages, employing a common means of communication. The design and construction of the Internet communication mechanisms is a major technical achievement, enabling a program running anywhere to address messages to programs anywhere else.

The internet is also a very large distributed system. It enables users. Wherever they are, to make use of services such as the World Wide Web, email and file transfer [1].

3.2 The Intranet

An intranet is a portion of the Internet that is separately administered and has a boundary that can be configured to enforce local security polices. An intranet is connected to the Internet via a router, which allows the users inside the intranet to make use of services elsewhere such as the Web or email Many organizations need to protect their own services from unauthorized use by possibly malicious users elsewhere [1].

3.3 Mobile and Ubiquitous Computing

Ubiquitous computing is the harnessing of many small , cheap computational devices that are present in users physical environments, including the home, office and elsewhere, The term ubiquitous is intended to suggest that small computing devices will eventually become so pervasive in everyday objects that they are scarcely noticed [1].

3.4 Resource Sharing and the Web

Users are so accustomed to the benefits of resource sharing that they may easily overlook their significance. We routinely share hardware resources such as printers, data resources such as files, and resources with more specific functionality such as search engines.

Look at from the point of view of hardware provision; share equipment such as printers and disks to reduce costs. But of far greater significance to users is their sharing of the higher-level resources that play a part in their applications and in their everyday work and social activities [1].

4. Challenges on Distributed System

The construction of distributed systems produces many challenges:

4.1 Heterogeneity

They must be constructed from a variety of different networks, operating systems, computer hardware and programming languages. The Internet communication protocols mask the difference in networks, and middleware can deal with the other differences [1].

4.2 Openness

Distributed systems should be extensible-the first step is to publish the interfaces of the components, but the integration of components written by different programmers is a real challenge [1].

4.3 Security

Encryption can be used to provide adequate protection of shared resources and to keep sensitive information secret when is transmitted in message over a network. Denial of service attacks are still a problem [1].

4.4 Scalability

A distributed system is scalable if the cost of adding a user is a constant amount in terms of the resources that must be added. The algorithms used to access shared data should avoid performance bottlenecks and data should data should be structured hierarchically to get the best access times. Frequently accessed data can be replicated.

4.5 Failure handling

Any process, computer or network may fail independently of the others. Therefore each component needs to be aware of the possible ways in which the components it depends on may fail ad be designed to deal with each of those failures appropriately.

4.6 Concurrency

The presence of multiple users in a distributed system is a source of concurrent requests to its resources. Each resource must be designed to be safe tin a concurrent environment.

4.7 Transparency

The aim is to make certain aspects of distribution invisible to the application programmer so that they need only be concerned with the design of their particular application. For example, they need not be concerned with its location or the details of how this operation is accessed by other components, or whether it will be replicated or migrated. Even failures of networks and processes can be presented to application programmers in the form of exceptions- but they must be handled [1].

5. Characteristics of Distributed System

A distributed system can be much larger and more powerful given the combined capabilities of the distributed components, than combinations of stand-alone systems. But it's not easy - for a distributed system to be useful, it must be reliable. This is a difficult goal to achieve because of the complexity of the interactions between simultaneously running components [1].

To be truly reliable, a distributed system must have the following characteristics:

- Fault-Tolerant: It can recover from component failures without performing incorrect actions.
- Highly Available: It can restore operations, permitting it to resume providing services even when some components have failed.
- Recoverable: Failed components can restart themselves and rejoin the system, after the cause of failure has been repaired.
- Consistent: The system can coordinate actions by multiple components often in the presence of concurrency and failure. This underlies the ability of a distributed system to act like a non-distributed system.
- Scalable: It can operate correctly even as some aspect of the system is scaled to a larger size. For example, we might increase the size of the network on which the system is running. This increases the frequency of network outages and could degrade a "non-scalable" system. Similarly, we might increase the number of users or servers, or overall load on the system. In a scalable system, this should not have a significant effect.
- Predictable Performance: The ability to provide desired responsiveness in a timely manner.
- Secure: The system authenticates access to data and services [2].

6. Advantages of Distributed System

We can achieve many advantages by using distributed system for cooking oil bottle production factory. There are as follows:

- Easier to integrate different applications running on different computers into a single system
- Scale well with respect to the sized of underlying network
- Data and hardware resource sharing
- Reliability if carefully engineered
- Incremental growth
- Parallel Processing
- Time saving and aggregate computing power
- Faster time than human labor's time

7. System Design

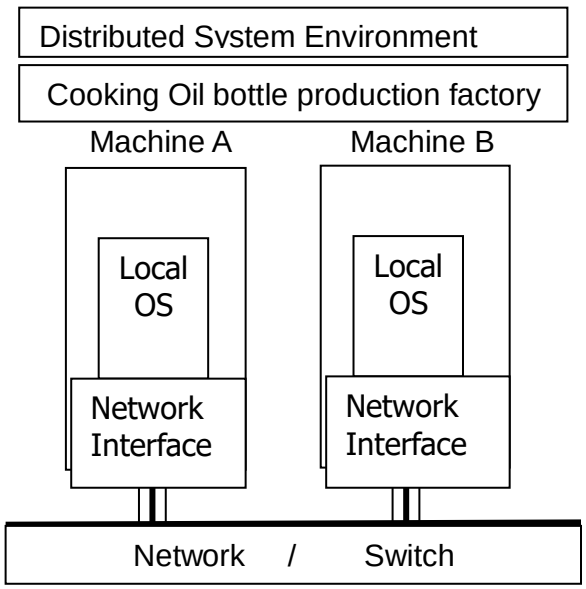

Figure 2. System Design of Cooking Oil Bottle Production Factory based on Distributed Systems

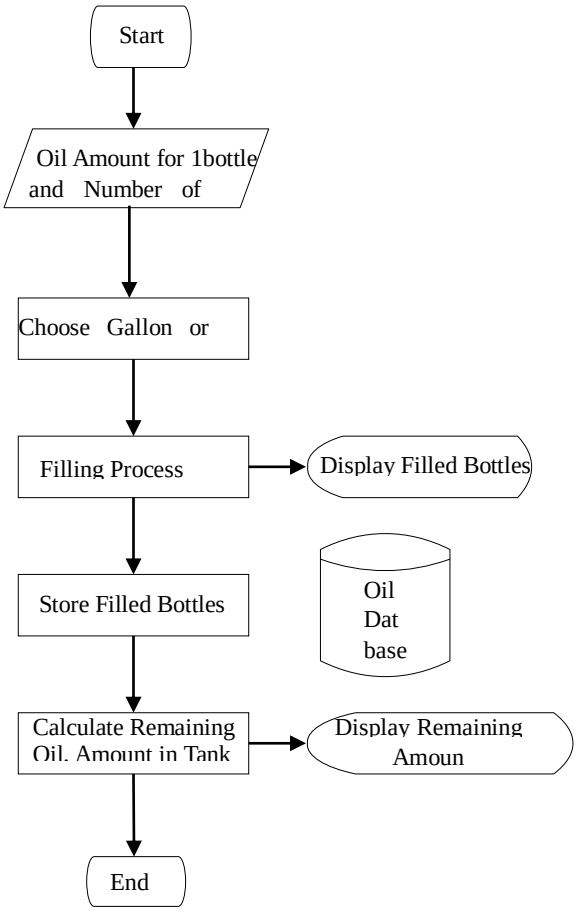

Figure 3. System Flow Diagram of Filling Process

In the Filling Process, the user inputs the amount of oil in a tank. Then, the user works out oil unit. In the process, two functions can be done at the sane time. One system fills the bottles with oil. Another system packs the bottles that have been filled with oil.

The user can view the number of the packing bottles and that of the finished bottles and then filled bottles taken to the packing process one by one. When the number of filled bottles is equal to the number of input bottles, the filling process has finished.

The Packing Process packs the bottles filled with oil one by one. The packing process and filling process can be done at the same time. The packing process produces packed bottles one by one. When the packing process has finished, it shows the finished signal.

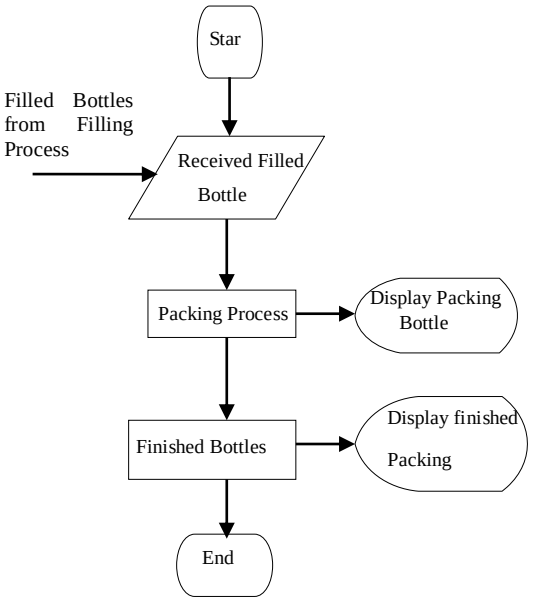

Figure 4. System Flow Diagram of Packing Process

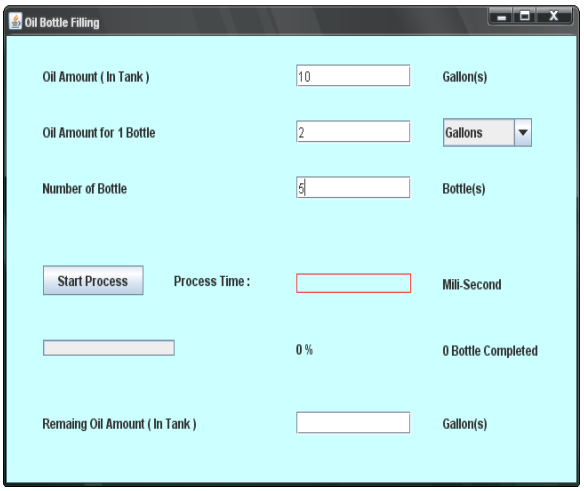

Figure 5. Insert Oil Amount, Oil Amount for 1 Bottle and Number of Bottle for Machine A

In Machine A, the user types the amount of oil in tank. After defining the amount of oil in tank, the user identifies oil amount for one bottle and then the user identifies the number of bottle in Figure 5.

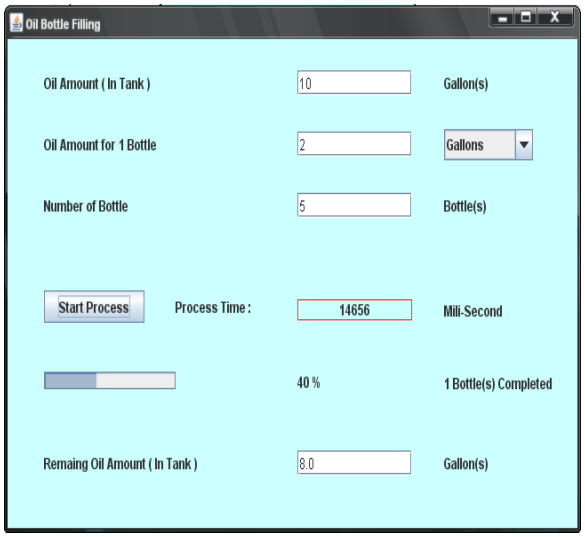

Figure 6. Oil Filling Process

In Oil Filling Process, the user clicks "Start Process" button. The system fills the oil to the bottles. The user can view process time (Millis Second), completed filling bottles and remaining of Oil Amount in Figure 6.

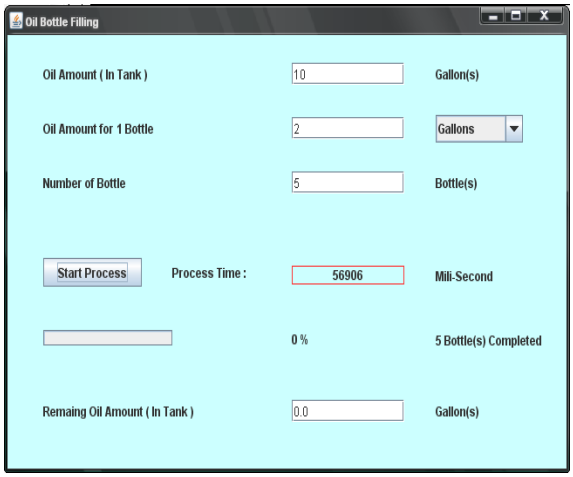

Figure 7. Finished Oil Process

In finished oil process, the user can view the total process time of the filled bottles and all number of the filled bottles in Figure7.

After filling one bottle in Machine A, Machine A sends the filled bottle to Machine B based on the distributed system. Machine B receives the filled bottles and then Machine B processes the packing bottles.

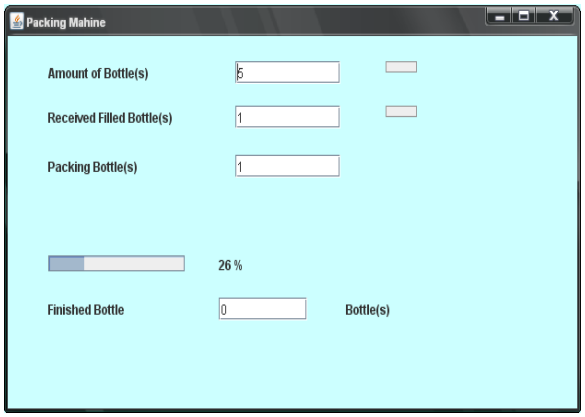

Figure 8. Packing Process for Machine B

In this process, Machine B receives the filled bottles after filling bottles in Machine A. If Machine A finishes one filled bottle, Machine A sends one filled bottle to Machine B. If Machine A finishes two filled bottles, Machine A sends two filled bottles to Machine B and so on. If Machine B receives the filled bottle, Machine B starts the packing process. Machine B results the finished packing bottles in Figure 8.

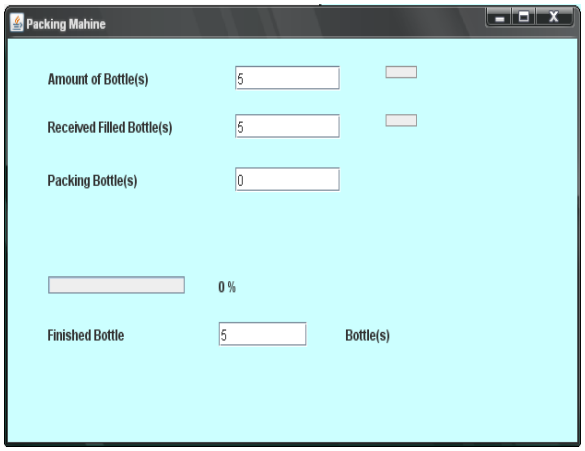

Figure 9. Finished Packing Process

In finished packing process, the user can view all finished packing bottles in Figure 9.

8. Conclusion

The system has been implemented based on the distributed client-server model. Traditionally, distributed systems have been developed for massively parallel computing, usually for large, homogenous systems, relying on some common, specialized hardware or software mechanism, to support for distributed shared memory in modern industrial system. So, this system should work on any condition. The system can run more accurately than human labor. It can give the knowledge of distributed automation system.

REFERENCES

[1] G. Coulouries, J. Dollimore, and T. Kindberg, "Distributed Systems Concepts and Design," Third Edition, ISBN-0-201-61918-0.

[2] C.D. Jones, A.B. Smith, and E.F. Roberts, *Book Title*, Publisher, Location, 2000.

[3] A.B. Smith, C.D. Jones, and E.F. Roberts, "Article Title," in *Proc. IEEE Int. Symp. Circuits and Systems*, Monterey, CA, pp. 11-14, June 1998.

[4] G. Coulouries, J. Dollimore, and T. Kindberg, "Distributed Systems Concepts and Design," Third Edition, ISBN-0-201-61918-0.

[5] C.D. Jones, A.B. Smith, and E.F. Roberts, *Book Title*, Publisher, Location, 2000.

[6] A.B. Smith, C.D. Jones, and E.F. Roberts, "Article Title," in *Proc. IEEE Int. Symp. Circuits and Systems*, Monterey, CA, pp. 11-14, June 1998.