Architecture for Building Web-Based Communication in Reconfigurable Manufacturing Systems (RMS)

A. H. Tseumogne Noumodje, K. Mpofu, N. S. Tlale

1Department of Industrial Engineering, Tshwane University of Technology, Pretoria, South Africa
2Department of Industrial Engineering, Tshwane University of Technology, Pretoria, South Africa
3Anglo America, South Africa

Abstract
Reconfigurable manufacturing system (RMS) is a solution to the customer's satisfaction in the context of mass customization, huge variable product or service demand. Given that RMS is adjustable on production capacity and functionality, its configuration needs to be rapid and manageable within the enterprise despite location. An enterprise having various branches globally has a need for good communication when changes need to take place. Therefore an appropriate software architectural technology platform is needed so reconfiguration can happen quickly. This paper introduces the main characteristics of software architecture requirements for web-based communication in RMS, and presents a comparison between languages used for architecture design.

Keywords
Reconfigurable Manufacturing, Architecture, Middleware

1 INTRODUCTION
Manufacturing companies are the centre of operation to respond to the markets' varying demands. A new paradigm has been developed to ease the responsiveness: Reconfigurable Manufacturing System (RMS). It is a new manufacturing system that can be modify in structure (hardware and software) depending on the production. In spite of its advantages, its implementation is cumbersome due to unavailability of its components. Effectiveness and efficiency of RMSs could be improved by implementing a system of communication between end users of RMS when changes in configurations take place. The idea of developing a system to search for modules on the web arises. As the size and complexity of software systems increase, the design of the software goes beyond the algorithms and computation currently available. Designing and specifying the overall system structure emerges as a problem.

In this paper, the term architecture refers to software architecture. It is defined as a description of how the application functionality is distributed over a number of logical components and how these components are distributed across processors [1]. Architecture is the art of organizing modules into a whole to achieve a function, in the presence of constraints. In this instance, researchers seek to enable communication in RMSs through the web.

The principle developed by Zhu [2] to build an architecture for large-scale web-based virtual communities is used to achieve this goal. Researchers first define RMS systems and their enabling technology. The next step is presenting the software architecture requirement for web-based communication in RMS. It will be followed by the different types of software architecture technology. Comparison of available technology is made and selection of the right architecture for building web-based communication in RMS is discussed before the future work is expressed.

2 RMS AND COMPONENTS
Human needs evolved dramatically in the past centuries and very much so in the 20th century, the 21st century has also seen significant technological innovations before even a quarter of the century has passed. Responses to those unpredictable needs and change in the market, intensifies the competition between producers. To be able to satisfy the demands, manufacturers have to adapt their systems and still be able to manage their costs by using the right technology. Therefore, manufacturing systems have evolved from dedicated manufacturing systems (DMS) to reconfigurable manufacturing systems (RMS) via flexible manufacturing systems (FMS). These manufacturing systems are defined and compared in terms of capacity, cost and flexibility; The advantages of RMS and its components are shown in figure 1 [3]. DMS and FMS are fixed systems. The first one is static in terms of automation and produces parts at high volume; therefore the production cost is low. For the FMS, it can produce a wide variety of products with changeable volume on the same system but then the costs for
The combination of the advantages of the two technologies creates RMS. It is defined as follows: “A Reconfigurable Manufacturing System (RMS) is designed at the outset for rapid change in structure, as well as in hardware and software components, in order to quickly adjust production capacity and functionality within a part family in response to sudden changes in market or in regulatory requirements”. So RMSs are dynamic systems which are made possible by their components: Reconfigurable Manufacturing Tools (RMT), reconfigurable software and reconfigurable controllers. These components are cost–effective because they are custom–designed for a given range of operational requirements, and can be economically converted to meet new requirements [4, 5]. RMSs enable rapid adaptation to production of new products through rapid reconfiguration. by Using Commercial-Off-The-Shelf (COTS) module concepts such a system has been developed [6]. It will reduce the manufacturing system development time in a manufacturing enterprise.

However, when changes and configurations occur in one location of the manufacturing company, there is no rapid means of communicating those changes and configurations so that the RMS of the remote location can be immediately reconfigured. Therefore the need to develop a platform component through the web has been indicated. In the manufacturing environment, some solutions to the communication issues have already been developed. E-Manufacturing [7] and production in network [8], these solutions emphasize on the sharing of information from shop floor to management over different locations so that all localities are harmonised with one another. It is noted by the authors that, in the RMSs environment, these solutions have been applied [9-12]. However, we would like to synchronise the production. More we think of sharing in real time parts from the concept development, design, simulation and the analysis on a real time basis. The premise is to use web communication as a solution. The third section of the article focuses on architectures that could be used to realise this goal.

3 SOFTWARE ARCHITECTURE REQUIREMENTS

Software architecture denotes the high-level structure of software system; therefore it is the foundation of implementation of any software. The architecture design phase is critical because most of the technical software issues are derived from there. The greater complexity in this instance is that users can access the application using a web browser. So the type of hardware and software the user will use is unknown. It is therefore important to define the basic software architecture requirements for a web-based communication in RMS in an open manner.

- Platform independence

The application system may run on different types of Operating Systems (OS) or platforms, such as Unix/Linux and Windows. To integrate heterogeneous platforms, the architecture technology should offer a neutral standard. It will allow interoperability between different types of OS.

- Language independence

Some parts of the system can be programmed in different languages. As the middleware can lay between software with its complexity, it is advised to have different modules to build the software[13]. Those modules are independently developed elements that are not combined until the program is linked, which are needed to resolve specific tasks; they must not be confused with the mechanical modules of RMSs to build an RMT. The architecture design technology should allow interoperability between the different programming languages.

- Security

Security is an essential element in the web-based communication in RMS. In distributed manufacturing systems, security is crucial with the exchanges that occur for a safe communication [13].

- Multimedia Support

Multimedia is included in the application to convey information through the Internet. It provides an effective way to communicate through the Internet [14].
Reusable resources
Reuse refers to using the component of one product to facilitate the development of another product with different functionality. In order to go faster in the development of the application, reduce the cost and improve the reliability; it is essential to acquire reusable resources and use them profitably [15].

Compatibility for legacy applications
Legacy application refers to all of the large applications that companies use to do business. Most organizations today have investment in applications and Information Systems (IS). To be able to constantly upgrade and change due to rapid development, the application should integrate legacy systems.

Programming technology support
Programming is the next phase following the software architecture design for the web-communication in RMS development life cycle. Programming is the process of coding and implementing the software architecture. The programming approach must provide the software coding technology an opportunity for system owner to realize the designed software architecture.

4 DIFFERENT TYPES OF ARCHITECTURES
Since the software is going to lie in the operating systems, it is middleware. Middleware objects mediate platform-specific and network translation issues so that programs developed by different vendors can communicate in the network. For this structure, researchers look at the mainstream distributed architecture technologies. Distributed object architecture facilitates communication between different programs in a network.

4.1 CORBA
The Object Management Group defines and manages the Common Object Request Broker Architecture middleware specification. CORBA, a core part of the Object Management Architecture, provides a method of invocation mechanism with location and implementation transparency. CORBA and the Object Management Architecture (OMA) specify three components as basic building blocks for distributed applications: the Object Request Broker (ORB), client interfaces (stubs), and server interfaces (skeletons). Communication protocols have been specified for interactions among ORBs (General Inter-Orb Protocol: GIOP, Internet Inter-Orb Protocol: IIOP) [3]. See figure 2, it illustrates the basic CORBA architecture.

4.2 DCOM (Distributed Component Object Model)
Distributed COM is COM extended over the network. Clients use DCOM to interact with components on other hosts. DCOM also extends COM’s location transparency to remote processes. This means that clients don’t need recoding to access objects on other systems—“it just works.” Further, DCOM supports multiple security mechanisms and transport protocols.

4.3 Java RMI (Remote Method Invocation)
Java remote method invocation is built into the Java framework. This framework can distribute any Java object that implements the java RMI remote interface. See the representation in Figure 4.
5 CORBA/JAVA FOR ARCHITECTURE

CORBA is an industry–level standard and support for distributed computing. Java is a widely used platform, component model and programming for building distributed objects[16]. For building web-based communication in RMS, the combination of CORBA and Java (CORBA/Java) will be the most suitable architecture technology. Based on the software architecture requirements, architecture such as DCOM and Microsoft.Net are not suitable. It is due to the fact that they do not provide most important requirements for the software system such as platform independence and multimedia support,

- Platform independence
In CORBA/Java, the platform is supported by both Java and CORBA. In Java, Java Virtual Machine (JVM) is especially designed to provide portability to different platforms. So Java objects can run on any platform that supports JVM. In addition, CORBA provides the functionality that objects implemented using a different Object Request Broker (ORB) can inter-operate through the General Inter-ORB Protocol (GIOP) and the Internet Inter-ORB Protocol (IIOP). The CORBA specification with Java allows writing applications that are portable across operating systems, platforms and Java ORB implementations.

- Language independence
In his research, Zhu affirmed that CORBA is the appropriate technology to use when building systems that are written in a mix of programming languages and operating systems[16]

- Security
CORBA provides security mechanisms through IIOP over Secure Socket Layer (SSL) and CORBA security service. SSL is a security protocol that sits on top of the TCP/IP (Transmission Control Protocol / Internet Protocol) transport protocol, which has become popular in the context of the Web as the underlying protocol for secure HTTP (Hypertext Transfer Protocol) With CORBA security service, a framework is defined for the use of different underlying security technology to secure CORBA applications.

- Multimedia support
CORBA provides the Audio/Visual Streams Standard, which supports delivering multimedia elements, with the language-independent and platform-independent facility. Java's component (Core Java 2 platform and Java Media Framework (JMF)) libraries support images audio and video. The CORBA/Java architecture is available to convey and render multimedia elements for the heterogeneous web-based distributed environment as shown in figure 5.

- Reusable resources
CORBA-services are collections of system-level services packages with IDL (Interactive Data Language) specific interfaces. In addition to simple distributed object capabilities offered by ORB, CORBA-compliant ORBs provides a number of optional services defined by the OMG, called CORBA-facilities. CORBA-services are used to create components, main components, and introduce component to environments. They also provide fundamental services for building objects. So application developers can reuse these services via interface specification to find and manage the application objects, and co-ordinate complex operations.

CORBA-facilities are end-user-orientated interfaces that provide facilities across application domains. They fill in the architecture between the basic CORBA-services and the visible application objects. CORBA-facilities are useful for almost every application project. All these aspects will simplify the architecture design and programming process. Hence, with OMG (Object Management Group) IDL, CORBA-services and CORBA-facilities provide an international standard for building and managing complex web objects based on heterogeneous platforms and languages.

- Compatibility for legacy application
Java is the most suitable programming language for internet-based applications. Under CORBA environment, Java can be used for developing Internet-based objects, the legacy application can be wrapped by OMG IDL, and other languages can combine with Java to obtain the most appropriate Internet accessibility.

- Programming technology
Java provides a clearer approach to object-orientated programming than C++, with fewer memory management responsibilities, no pointers, a less confusing syntax, and simpler method resolution rules. Additionally, Java provides features not available in C or C++ such as automatic
garbage collection, exception handling, and integrated thread support. These features are desirable and are particularly useful for distributed systems programming. Java also has a component which can help the programmer to combine functionality provided by a number of Java classes into a single component, called JavaBeans. Therefore, components can be easily put together to achieve new functionality.

6 COMPARISON OF ARCHITECTURES
Based on the requirements to build the architecture, we compared the different architectures concluding that the chosen combination is adequate. This comparison is established on the key architecture requirements among the above stated reasons for building the web-based application. These criteria are the platform independence, language independence, legacy application and multimedia support.

In Table 1, we have the summary of the comparison between the architectures which prove that based on the criteria stated to argue the case, the complementation of CORBA and Java provide a solution to the software architecture needed in an RMS environment.

<table>
<thead>
<tr>
<th></th>
<th>Platform Independence</th>
<th>Language Independence</th>
<th>Legacy Application</th>
<th>Multimedia Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java</td>
<td>Enable (with JVM and JRE)</td>
<td>Through CORBA</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>CORBA</td>
<td>-</td>
<td>YES</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DCOM</td>
<td>ONLY windows-based OS</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Microsoft .NET</td>
<td>Not all OS</td>
<td>Yes</td>
<td>NOT all system</td>
<td>Limited</td>
</tr>
</tbody>
</table>

Table1 - Comparison table

7 CONCLUSIONS AND FURTHER WORK
This paper looks at the architecture in a broad way, the next milestone in this work will focus on the detailed aspect on how the middleware elements will be put together to have the application running.

The architecture solution given here is from a theoretical analysis, actual practical implementation is planned for the future. This means that, from the architecture solution, the development (programming or coding) of the software will be executed.

CORBA/Java presents a good solution to the architecture challenge, nonetheless on the contrary it also has flaws such as software reuse. The following task is envisaged to be finding a methodology of having a very useful application with less complexity.

The main goal is to have a running middleware RMS software, therefore after the concept is confirmed the process of coding will be executed.

Communication in RMSs can be improved in many aspects such as interoperability and interchangeability between different systems. To enable that development, research has been undertaken. This paper focused on the basic elements needed to design a web-based architecture in RMS. The previous section proves the suitability of CORBA/Java for software architecture in web-based communication to RMS application. To prove the availability of CORBA/Java is to prove that the CORBA/Java architecture technology is compliant with the architecture requirements discussed in section 2.

8 ACKNOWLEDGEMENT
Sincere acknowledgements go to Tshwane University of Technology TUT, Technology Innovation Agency TIA and my supervisors, in particular Dr Khumbulani Mpofu and the whole research team under his supervision.

9 REFERENCES


10 BIOGRAPHY

Aymar Harold Tseumogne Noumodje student in Master's degree of technology at Tshwane University of Technology. His focus area is the application of Information technology in Reconfigurable Manufacturing Systems.

Dr Khumbulani Mpofu from the Tshwane University of Technology. He is currently postgraduate coordinator in the Department of Industrial Engineering. His specific focus and area of research interest is in Advanced Manufacturing and Robotics.

Dr Nkgatho Tlale from Anglo America, a research leader in advanced manufacturing and robotics. With a wealth of experience in the academia, research, business and leadership.