Software Development and Testing for Shipyard Robotic Welding

Evanthia Antoniou¹, Nikitas Nikitakos²

*¹ Algosystems S.A., Applied Research Department
Mirtidiotissis 9 & Fidiou 2 185-33 Piraeus, Greece
tel +30-1-4112802, fax +30-1-4118736
email :evanthia@rtd.algo.com.gr*

*² Hellenic Naval Academy
PO Box 80318, Piraeus 18510, Greece
tel +30-1-4625993, fax +30-1-4181768
email :nikitas@naxos.esd.ntua.gr*

Abstract

Software Development and Testing in a shipyard environment is a challenging task, due to many difficulties arising. An intelligent robotic welding system installed in a shipyard and consisted of several sub-systems running on different platforms and using separate software, was the object of software development methodology presented in the paper. The several quality factors, the sequential development steps and the testing reviews are presented. A validation of project at the end, gives useful information for future consideration of the proposed methodology, in case of similar development and testing.

Keywords

Robotic welding system, Software Quality Factors, Software Development, Software Testing.

1. INTRODUCTION

Shipyard welding is a job that, when done by humans requires considerable training skill and concentration because there are often too many variables to be considered in

the shipbuilding industry for a number of advantages such as faster production, product quality, economy, flexibility, etc.

An intelligent robotic welding system was installed last year in a Greek shipyard. The system is consisted of different hardware units connected by specific software modules and is capable to work efficient in a heavy industrial environment. In this paper, we present a methodology for software development and testing applied during implementation phase of this particular robotic welding system.

The paper is organized as follows : In section 2 a brief technical description of robotic system is presented. Next, in section 3, the quality factors development are reviewed followed, in section 4, by detailed description of methodology for whole system, the different subsystems including the reviews for software testing concurrently with the procedures. Finally, in section 5, the methodology is validated and, in section 6, our conclusions are presented.

2. SYSTEM DESCRIPTION

The robotic system, is concerned with the repair, by welding of accident damaged ships. The objective of the system is to fully automate the welding of large, steel structural modules used in shipbuilding and repair - a task which is done so far as a manual process.

The system is designed to be fully automatic and incorporates many state-of-the art technologies including knowledge-based (KBS) , vision survey systems, advanced laser and ultrasonic range sensors, advanced robotic simulation and planning and advanced sensor control of robot motion. Figure 1 shows the control architecture of the system.

Figure 1. Welding Robot System

The components to be welded together to form the modules are typically large steel plates laid horizontally, with various stiffeners arranged across the surface to form structures. These are set up manually according to well-defined procedures, but the

individual parts can only be placed with limited accuracy, not sufficient for conventional robotic welding.

The modules can, however, be classified into a limited number of categories and described parametrically. Typical module categories are : bulkhead sections, deck sections, etc. The parameters include the component dimensions, the number and type of stiffeners, webs, struts, and other component parts. Also, welding information such as : joint type (fillet, butt), joint attitude (flat, horizontal-vertical, vertical, etc.), leg length, root gap, number of passes are required. By using these parameters in conjunction with a knowledge-based system, which has been programmed with all the process information to carry out the welding task, a "worst case" representation of the module can be produced, together with instruction on how to weld it.

Key features, of the system, are:

- adaptive robot control
- multi-sensor control,
- welding of large structures, and
- open platforms

The vision survey system measures and determines the position and orientation of the component to be welded when dealing with unknown workpieces and passes the information to the KBS, which adds the process information to a dimensional and positional module to form the complete model. The resultant output is used to drive a graphical simulation system with automatic collision avoidance algorithms built in. This produces collision-free nominal trajectories for a nine degree of freedom, gantry-mounted robot. The trajectory is then combined with the process information and transcribed into a nominal trajectory program for the robot. The nominal trajectory program is then passed to the robot controller, which uses "through arc" and laser sensors techniques to modify the path of the robot in real-time, and to control the welding system as the robot is actually welding the component. As the robot executes its task, the workpiece model is updated according to the sensor data. Ultrasonic sensors are used to determine the precise location of the start of the weld path, and for collision avoidance (detection of the exact location of struts, stiffeners, etc.).

3. QUALITY FACTORS IN SOFTWARE DEVELOPMENT

A quality factor is an aspect of a software product which is important to the customer or the developer. Sometimes it is important to both. Those factors were the guidelines in software development for the intelligent robotic welding system. The factors are briefly reviewed since the robotic welding system is primitive in shipbuilding production and has many differences with the already known other production systems.

The first quality factor is *correctness*, that is , that a software system actually conforms to its requirements specification, in our case to several welding scenarios. The next is *maintainability* or *modifiability*, which describes the ease with which a software system can be changed. This factor, is very important during whole the development procedure, due to very imperfect shipyard conditions. There are three categories of modifications. The first category of modifications, due to error fixing, are known as corrective changes. The second category , due to the developer responding

to changes in requirements, are known as adaptive changes and the third category, which improve a system, are known as perfective changes.

Testability describes the ease with a system or part of a system can be tested. A robotic system, which has a very poor requirements specification that contains a large amount of ambiguities, is very difficult to test. Another important factor, is *usability*. This is the effort required to learn, operate and interrupt a functioning system. Usability is a major problem with the system since the human interface has to be in the simplest possible operation level for workers and automatic production is an innovation in a shipyard.

Reliability describes the ability of a software system to carry on executing with little interruption to its functioning. Since robot is a safety-critical system it can be specified in terms of metrics involving factors such as mean time between failure.

Efficiency, is another important quality factor. It is used to describe the degree to which computing resources - file space, memory and processor time - are used in an application, since our application is novel. This is a difficult factor to categorize and to explicitly specify.

Integrity [Gri-93], is used to describe the extent to which the system and its data are immune to access by unauthorized users. In our system, there are many sensitive welding data interchanged during robotic welding operation and any modification of them cause severe errors.

The final quality factor, is *interoperability*. This is the ability of a system to operate in conjunction with another software system, for example a spread-sheet. This factor is of direct interest for future development in automatic production for the shipyard.

4. SOFTWARE DEVELOPMENT AND TESTING ACTIVITY

4.1 General view of system implementation

The general steps which were followed in order to build the system are represented in Fig 2. In this figure, several sequential steps are presented related with several testing reviews during the whole system development procedure

A detailed description of each step is followed.

- Enterprise Modeling and Domain Study

It was very important for the implementation team to study the general enterprise model, particularly for shipbuilding production, to understand the intention of the enterprise to supply the system and to find out if the user was able to provide all necessary resources (money, personnel, working area and of course patience) until the end of the project [Pre-92].

- System Requirements Analysis

This phase was one of the most critical development phases, because it defined and documented the common understanding of the functional and performance requirements baseline [Ala-90]. In this phase, the problem was bounded. The idea of the project was confused in users mind who knew the working field of manual welding

but have not knowledge what exactly was possible to be implemented by an information technology system in robotic welding.

Figure 2. General view of an Robotic Welding System implementation

It was decided that the robotic system will be installed in the ship hull workshop and will be integrated in the production line of the yard. All the basic techniques which are state of the art in the automated robot welding technologies such as laser sensors, ultrasonic sensors and vision systems were agreed to be used by the intelligent robotic welding system.

- User Needs and Constraints Analysis

In this step, the working shell was initially defined as follows : “The robot will be installed in the particular place, and the operator will stay in an office environment, in the vicinity of the robot, from where he will provide the commands to the system”. Based on this outlines a detailed analysis was specified. As an example the workpieces to be welded were classified in general categories according their geometry and their characteristics, i.e. chemical composition of steel, plate thickness etc.

- System Design

This activity was divided in two parts : the preliminary and the detailed system design [Pre-92]. The sub-systems involved in the process and their operation were defined. The following systems were used : a multi -sensor system composed of the laser sensor responsible for the seam finding and seam tracking and an ultrasonic sensor system for the obstacle detection and recognition, a simulation system in off - line programming to provide the collision free parametric trajectories and in on - line to help the operator to monitor the system operation, a vision system to provide

geometrical data concerning the workpiece, and a Knowledge Based System which is the supervisor of the whole process [Kar-89].

- **Software Requirements Analysis**

This was the task in which the type of Real Time Operating Systems inside the supervisory machine, the robot controller and the peripheral machines in which reside the software for each sub - system were selected. The principles of the intercommunications between the tasks were also, carefully, designed.

- **Software Logical Design**

The flow chart of the software applications for each task of the system was designed to help the implementation team agreement on the logical system operation. Particularly, a special care was given in priority of several task to be performed by robot.

- **Software Implementation Design**

It was decided the different developers team work to prepare the software applications. The result was the implementation schedule of the software modules and the plan for their integration.

As an example, for the ultrasonic sensor system the schedule was :

- Develop the algorithms for the obstacles recognition and test their correctness and their effectiveness in the laboratory using mock ups of the real objects.
- Develop the robot program open for the input from the ultrasonic system.
- Develop the dispatcher part (software application which resides in the supervisory machine and it is responsible for the communications inside the system) which exchanges data between the robot program and the ultrasonic software module.
- Integration of the two modules.
- Operation testing.

- **Development Tools Selection**

In this state of the process, the basic development and testing tools were selected, such as the NEXPERT for creating Knowledge based applications, representing domain knowledge, and completing tasks in the domain using its inference engine, the MOTIF X Windows manager for the KBS user interface development, the operating system platforms, the programming language C++ for the peripheral applications, the static code analyzers, the profilers and the tools for the automated testing techniques.

- **Coding / Programming**

Skilled programmers and coders made the programs which implement the algorithms for each task of the process. The result of this phase was not any list of specifications or agreements but it was real products, i.e. the parts of the final system. They followed the schedule of the implementation from the previous phases and they worked together, in parallel or using the results from other tasks if it was necessary, to produce the different software modules. All the personnel had in mind the compromise that they had to accept for diverse tasks.

- **Software Integration and Testing**

The software integration and testing phase was designed to integrate the software components. All problems which were arisen in the code, were analyzed and changes were documented to incorporate fixes and patches into the code. The developers team, the coders, the testers, the software quality assurance team, and the user collaborated on the gradual integration of the software modules, already prepared from the previous step, to produce the final software product. The software system was evaluated under testing conditions in order to test each individual sub-system right operation and the correctness of data exchanges between them.

- **System Integration and Testing**

The System Integration and Testing was conducted to ensure that different software codes integrated together meet all the functional and performance requirements of the project, such had been defined in the System Requirements Analysis step. The software product was integrated in the hardware of the system and the final product worked many hours under real welding conditions in a heavy and polluted shipyard condition. Many modifications were applied on the software modules and the operation of the system, due to the specific demands of the production environment. The importance of this stage of the process was determinative because it was the last step before the robotic system integration in the production line of the factory. For the final acceptance of the system operation, the user was responsible focused primarily in the welding quality and then in the usability of the system.

4.2 Milestones for software testing

During the development procedures milestones for software testing were defined in appropriate sequential tests as shown in Figure2. In more details those reviews were:

- **Specification Review**

It was the first milestone of the project. During the initial phases a very careful study was accomplished and a draft specifications overview of the goals of the project was generated.

- **Preliminary Design Review**

Before beginning of the actual programming, preliminary review was necessary. The general system design and the software design had to be proceeded before the developers team started to study the implementation process.

- **Critical Design Review**

This was the review with the results of the actual implementation phase. The output of this milestone was a Information System product. The objective of the review was the answer in the questions “ What system we have already implemented ? “ , “ Is it the right system ? ” , “ What kind of modifications or changes are necessary to be implemented in order to build the system that makes user happy ? “. This review had the first output which consists a pilot of the final system or a part of the delivered system.

- **Test Readiness Review**

As it is mentioned before, the system have been implemented modularly. This means that there were numerous system components each of one was a stand-alone system which solved a different problem of the whole system. For example, there was a unit which addressed the obstacles detection during the welding by the system and also an individual unit which computed the right welding parameters for the joint of a specific workpiece. After the implementation and the individual testing of the units above, as well as their integration a review in which the concurrent software modules operation was examined

- **Functional Audit**

This was the last review of the project. After the software and hardware integration, the whole system was constructed, an intelligent robotic welding system. Next the system evaluated under real conditions for a long time. The objective of this milestone was to demonstrate the operation of the system working in production, and to establish that the system was reliable, safe and economic in use.

Three important comments were extracted from project implementation that were dominant for the whole software development .

- * The testing procedure was integral part of the whole Information Technology development activity [Bow-88]. This means that this phase begun early, from the start of the process, proceeded in parallel with the other phases (which follow the Waterfall model) and closed at the end of the process producing the final product. The software testing was not an afterthought, done after a development activity was finished. Due to the complexity of the software development in the robotic system, there were the need of methodical software testing integration into the whole development process.
- * The most familiar phase of software development, the actual programming phase, i.e. the code writing, was a small part of the process, which was implemented at the latter stages of the Software System Development.
- * Each milestone was very important because it composed the feedback from the user to all the other teams involved in the process. In other words, was the approval of the user to the implementation team.

4.3 Sub-systems implementation process

The most important part of the development was the procedure for the in parallel design, of the several sub-systems [Mau-91]. A general strategy was adopted for the sub-units implementation. The basic concepts of this strategy are described in the following paragraphs. The approach is that, the output of the implementation process is the result of a cyclic repeatable procedure until the extraction of the final product.

For each sub-system, the whole process consists of four distinct phases as it is shown in the drawing above. The different processes during the implementation of the

product are noted between the arrows in the cycle. Inside the ellipses it is noted the output of each step which feed the next phase. The people who are involved in each phase are enclosed between curly brackets. The rectangle represents the limits of the process and the polygons outside are the main input and output of the process i.e. User Domain and Product respectively.

Figure 3. Sub-systems implementation process

The implementation process begins from the requirements step [Ala-90]. A complete list of specifications is the input of the next phase, the actual development phase. In this activity the teams which are involved are the developers and the testers and the output is the “program”.

This specific code is integrated with the other parts of the system to produce a complete Information Technology System for the application. The last phase from which the process passes through, is the reviews state. All the teams involved in the activity, participate in this most important process. The integrated system is evaluated and the result is a *prototype* or a more determined list of specifications.

There are two categories of prototypes [Mye-79] , [Gil-1993]:

The *design prototypes* which verify that it can provide the result required by the user. These prototypes also help the developers to touch the psychology of the user and the operators and enhanced the confidential of the user to the developers team. The Graphical User Interfaces (GUI) belong in this category.

In the system, a graphical environment had been created with all the masks which the user must complete with the appropriate information in order to inform the system about the type of the workpiece he wanted to weld. The process running underneath these masks was not prepared in this phase of the project. The user had the opportunity to check if these “communication” masks had the desirable

were helped to understand what was exactly the way required by the user to communicate with the system.

The *functional prototypes* which are parts of the final system. Using this prototype the user confirms that the prototype generates the expected result. The Simulation Systems is a typical example of this category. The simple way to operate the prototypes and simulation system (through a user friendly graphical environment and a menu driven environment) contributes towards the acceptance and the understanding of the advanced information technology systems by the operators and the user.

In the robotic system, many functional simulation processes were implemented in order to achieve the output that the user expected to be produced by an operation. For example, there was a sequence requested from the welding process to weld the workpiece components in order to achieve a good quality welding.

There were also, processes in the simulation system which simulated the robot configurations and the sequence of the robot movements during the welding process before its interpretation in robot program commands. The user informed the team of developers and testers about the correctness or not of the result of the simulation process. The process was opened in any modification and any change. The process continues many times, until the output of the review to be the final product.

5. PROJECT VALIDATION

The software development in the case of robot run concurrently with the rest production of the shipyard, the novel development, the major factor of human interface plus the technology transfer from a traditional production system were the main points. The reliability of whole system, plus the quality of welded structures were the main factors for integrating the robot inside shipyard production. We should notify that the time of software integration, was very restricted in order to have an economic product.

There are several dominants very important in the procedure, those are :

Systematic Testing : Using automatic testing techniques the repeatable testing were achieved.

Active user participation : Because of the role of the user, his verification through participation during the construction of the system was the feedback to the development team.

Directed benefits : the priority of parts delivery, depends on the key-system operations defined by the user. From the beginning of the implementation design, the key-system concentrated the main attention of the developer.

Right Solution / Right System : A very important aspect of the system right construction is that the developed system had met the real requirements.

6. CONCLUSIONS

A software development and testing methodology, has been presented in this paper. The proposed methodology, is based on both environment characteristics and user active participation. It supports cases when different sub-systems with different characteristics had to be integrated for operation as whole system, with time and requirements restrictions. The consideration of software quality factors and several testing reviews are key aspects of this methodology. The implementation of this methodology to an intelligent robotic system has proven the usefulness of the proposals as well as their effectiveness without severely restricting the production system capabilities of a shipyard.

REFERENCES

- Pressman, R. (1992) "Software Engineering - A Practitioner's Approach " McGraw-Hill, Inc., New York .
- Gritzalis D., Gritzalis S., (1993) "Operating Systems Security", 2nd ed., MITT Press (in Greek).
- Myers G. (1979), "The Art of Software Testing", John Wiley & Sons, Inc.
- Gilb T., Graham D. (1993) "Software Inspection", Addison-Wesley.
- Karen L McGraw, Karan Harbinson-Briggs (1989), "Knowledge Acquisition: Principles and Guidelines", Prentice -Hall International.
- Alan M.Davis, (1990) "Software Requirements : Analysis and Specification", Prentice -Hall International.
- Maus Rex, Jessica Keyes, (1991) "Handbook of Expert Systems in Manufacturing", McGraw Hill.
- Bowerman R. Glover D. (1988) "Putting Expert Systems into Practice", Van Nostrad Reinhold Book .

Evanthia Antoniou is a research associate with the Department of Applied Research ALGOSYSTEMS S.A. She holds a University Diploma in Computer Science (Software Engineering) from University of Patras (1991). She participated as principal scientist, in several European Community Projects such as ESPRIT, BRITE etc. Her research interest includes CAD for manufacturing, Software Development for large fabrication and Systems Integration. She is a member of the Technical Chamber of Greece.

Nikitas Nikitakos is an active Commander in Hellenic Navy and an Adjunct Professor in Electrical Engineering of the Hellenic Naval Academy. He holds a B.Sc. degree (Naval Engineering) from HNA (1980), a MSc degree (Electrical Engineering) and a MSc degree (Applied Mathematics) both from Naval Postgraduate School, USA (1988) and a Ph.D. (Electrical and Computer Engineering) from National Technical University of Athens, Greece (1996). His published work includes several papers on applications of signal processing and on computer integration manufacturing. He is a member of IEEE, the Technical Chamber of Greece and the Greek Computer Society.