Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. V (2010), No. 5, pp. 953-960

Using QSPS in Developing and Realization of a Production Line in Automotive Industry

N. Tudor, V.C. Kifor, C. Oprean

Nicolae Tudor
Continental Automotive Systems
Test Department
E-mail: nicolae.tudor@continental-corporation.com

Vasile Claudiu Kifor
"Lucian Blaga" University of Sibiu,
Director of the research department

Constantin Oprean
"Lucian Blaga" University of Sibiu
Rector

Abstract: Using the QSPS (Quality System for the production software) for industrial projects and not only therefore, has led to accurate running of the production line from beginning of the SOP (Start of Production). This paper presents the application way of the QSPS at one of the strongest European automotive company. By using of this system several significant costs savings and quality improvement can be observed.
The content of this paper will show step by step how to use QSPS for the integration of a production line in the traceability system from a big company in automotive industry.
The production line involved contains 56 production equipments, which have to be passed trough by the product before being packet and deliver to the customer.
The control of the line is done by this traceability system, so the impact of this system with the quality of the product is very high.
The structure of this system contains 7 steps. All of these steps are followed and executed in each System (test, pilot and production environment).
Keywords: quality improvement, control, savings, efficiency, capability.

1 Introduction

The traceability software for the production lines becomes more and more important and is a very significant process while running complex production lines with a high difficulty degree. The entire customer claims issues, statistics, the control of the process, CPK (process capability index) studies, FPY (First Pass Yield) and PPM (defective Parts Per Million) reports are done very simple, based on a strong traceability system.

Therefore a lot of techniques for implementing the traceability software were tested and checked, some of them successful and some less successful depending on the kind of the project. The paper below describes the practical usage of the QSPS system in real situation for the implementation of the traceability software of a complex line which contains 56 stations. The QSPS system, should improve the quality of work, cost and time saving at one side, but on the other side the system should be very simple and easy to be used in order to simplify the work activities and not to make it more complicated.

Copyright © 2006-2010 by CCC Publications

Also the system should cover and prevent all the risk which can appear in all 3 QSPS Phases (test, pilot and production) because of non conformity of improper software.

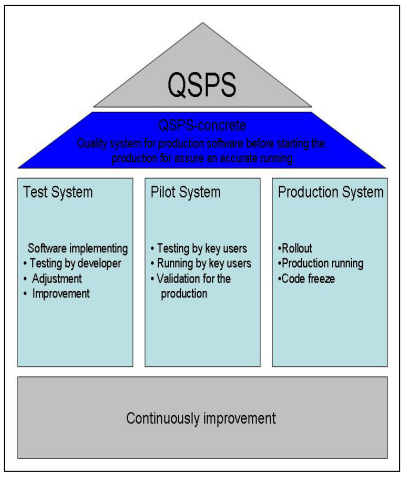

Figure 1: QSPS Framework

The next chapters describe step by step the implementation strategy of the traceability software, based on QSPS, of the already specified production line. All of the steps are described in the test (model) system, which have to be implemented and build up much closed (or even identical) to the characteristics of the production system. Those steps should be followed in all of the 3 environments systems of the QSPS (test, pilot and production).

2 Step I: Time and cost evaluation

This phase of the project should help to understand the deepness of the project, the difficulty level, and cost of the project, to make very clear and transparent the resources needed (time resources and headcounts) and to define a due date very closed to the reality.

Therefore the project should be split out in work packages, which can be better followed up, controlled and evaluated. The **Table 1** shows how the project was divided in work packages and subprojects.

Subproject	**Work packages**	**Difficulty level**
Process definition		2
Process description		2
Process review		2
Technical requirements specification		3
Target specification		3
Software implementation concept		3
Programming		3
Documentation of the software		2
Installation and configuration of the 56 modules		2
Testing the 56 Modules		2
Advising and user support		2
Project evaluation(the time for evaluation itself)		2
Internal software release	compliance to technical requirements	3
	compliance to the software ISO norms	2
	calibration and adjustments(bug solving)	3
	capability measurements	2
	usage work instruction	2
	user manual	2
	ergonomics and design	2
Customer software release	efficiency analyze	3
	defects analyze	3
	corrective actions	2
	analyze after corrective actions	2
	customer release reports	2
Validation and verification of the implemented software package	lifetime test	3
	test procedure for modification issues	2

Continued on next page

Table 1 – **continued from previous page**

Subproject	Work packages	Difficulty level
	approval procedure	2
	final releases	2
Functional follow up of the software during production	software audit	2
	statistical rate of the occurred errors	2
Customer claims	Problem recording	1
	Analyze and solving procedure	2
	Analyze report	1

Table 1 - Project segmentation

With this information the time evaluation can be done more accurate using the PERT algorithm. The results of the time evaluation can be observed in **Figure 2**.

Resource, time and costs evaluation

Project Name Traceability project of production line

Version Date 24.11.2009

Work Package (WP)	Number Employees	Rep.	Time[h] VO	VN	VP	VA	Total	Deviation	Dificulty Level	Costs (Euro)
Process definition	1,0		30	30	32	30,3	30,3	0,1	2	2821
Process description	1,0		16	20	20	19,3	19,3	0,4	2	1798
Process review	1,0		8	10	10	9,7	9,7	0,1	2	899
Technical requirements specification	1,0		80	90	90	88,3	88,3	2,8	3	14486,66667
Target specification	1,0		40	50	50	48,3	48,3	2,8	3	7926,666667
Software implementation concept	1,0		80	85	85	84,2	84,2	0,7	3	13803,33333
Programming	1,0		160	160	170	161,7	161,7	2,8	3	26513,33333
Documentation of the software	1,0		40	50	50	48,3	48,3	2,8	2	4495
Installation and configuration of the 56 modules	1,0		40	50	50	48,3	48,3	2,8	2	4495
Testing the 56 Modules	1,0		40	50	50	48,3	48,3	2,8	2	4495
Advising and user support	1,0		8	10	10	9,7	9,7	0,1	2	899
Project evaluation(the time for evaluation itself)	1,0		4	5	5	4,8	4,8	0,0	2	449,5
Internal software release	1,0					0,0	0,0	0,0		0
compliance to technical requirements	1,0		40	50	50	48,3	48,3	2,8	3	7926,666667
compliance to the software ISO norms	1,0		8	9	9	8,8	8,8	0,0	2	821,5
calibration and adjustments(bug solving)	1,0		16	18	18	17,7	17,7	0,1	3	2897,333333
capability measurements	1,0		8	10	10	9,7	9,7	0,1	2	899
usage work instruction	1,0		8	8	8	8,0	8,0	0,0	2	744
user manual	1,0		8	8	8	8,0	8,0	0,0	2	744
ergonomics and design	1,0		8	8	8	8,0	8,0	0,0	2	744
Customer software release	1,0					0,0	0,0	0,0		0
efficiency analyze	1,0		4	4	4	4,0	4,0	0,0	3	656
defects analyze	1,0		4	4	4	4,0	4,0	0,0	3	656
corrective actions	1,0		4	4	4	4,0	4,0	0,0	2	372
analyze after corrective actions	1,0		4	4	4	4,0	4,0	0,0	2	372
customer release reports	1,0		2	3	3	2,8	2,8	0,0	2	263,5
Validation and verification of the implemented software package	1,0					0,0	0,0	0,0		0
lifetime test	1,0		8	9	9	8,8	8,8	0,0	3	1448,666667
test procedure for modification issues	1,0		4	5	6	5,0	5,0	0,1	2	465
approval procedure	1,0		8	8	8	8,0	8,0	0,0	2	744
final releases	1,0		8	8	8	8,0	8,0	0,0	2	744
Functional follow up of the software during production	1,0					0,0	0,0	0,0		0
software audit	1,0		4	5	6	5,0	5,0	0,1	2	465
statistical rate of the occurred errors	1,0		4	6	6	5,7	5,7	0,1	2	527
Customer claims	1,0					0,0	0,0	0,0		0
Problem recording	1,0		0,5	0,5	0,5	0,5	0,5	0,0	1	30
Analyze and solving procedure	1,0		3	3	3	3,0	3,0	0,0	2	279
Analyze report	1,0		0,5	0,5	0,5	0,5	0,5	0,0	1	30
TOTAL			626	705	717	693,8	693,8	21,2		93549

Figure 2: Project time and cost evaluation

3 Step II: Internal software release

This step is the first evaluation of the software package from quality point of view. At the same time it is the first control tool used to signalize if the software was done in the right way, accordingly to the requirements, to the quality norms and not at least to assure the production of qualitative products. Therefore each of the steps below has to be verified.

3.1 Compliance to customer specification

This check has to be done based on the target specification. Each step from this chapter must be checked. For help a check table can be used and the characteristics from the target specification should be proved. For the line in our example following characteristics were verified:

- Program flow;
- Special cases in real life (like emergency stop, current interruption);
- Fail /pass /scrap situations;
- Handshake protocol between the industrial equipment and traceability software;
- Communication syntax;
- Repair scenarios;
- Limitation of the nr. of repair process;
- Check in/Check out process;
- Process parameter;
- Process results, CPK Measurements;
- Statistics of the process;
- Statistic of the failure;
- Product logistic on the line (process flow);
- Line flow control;
- Back flushing to the main database, in order to administrate material stocks;
- Label scanning;
- Label syntax;
- Packing. Check if the product pass all the processes in the line before being packed;
- Quality alert;
- Alert notification.

3.2 Check the implemented software package for software techniques and standards

For this point it was used the ISO/IEC 9899, for programming language C. Based on technical corrigenda 1:2001 the source code was reviewed. Also the next criteria were inspected at the very early beginning:

- Is the right software?
- The target is reached?
- Interface to the users is clear and unique while using it?
- At least one diagnostic message?
- The programming editor was used correctly? (Tabulator, empty spaces etc.)
- Requirements resulted from coded character set.
- Requirements resulted from binding techniques.
- Comments in the source code.
- Documentation.
- Usage documentation.

3.3 Software calibration and adjustments (bugs solving)

After the two chapters above were effectuated and all of the situations where verified, so all the bugs could be resolved, bugs which can occurred if the programmer doesn‘t consider a state from the program logic.

3.4 Capability measurements (using Pareto distribution, CPK, ISO 15504)

This measurement was done even in the test system. Production units can be created virtual for a good simulation. Knowing the handshake protocol between the traceability system and the

industrial equipment, a simulation can be done very closed to the reality situation. With these results the real running of the production can be compared. The upper, respectively the lower limits are in this case the cycle time of the process admitted by the customer while processing a production unit. CPK was done as described in **Table 2**, respectively the Pareto distribution for the analyses of the software package in case of rejected software, **Table 3** respectively **Figure 3**.

Unit Id	Measured time	Lower Limit 31	Upper Limit 36
1	33,5	30	36
2	33,2	30	36
3	34,1	30	36
4	33,9	30	36
5	33	30	36
6	34,3	30	36
7	33,2	30	36
8	32,9	30	36
9	33,8	30	36
10	33,6	30	36
11	33,7	30	36
12	33,6	30	36
13	32,9	30	36
14	33,1	30	36
15	33,8	30	36
16	34,3	30	36
17	33,9	30	36
18	33,6	30	36
19	33,2	30	36
20	33,3	30	36
21	32,7	30	36
22	33,5	30	36
23	33,8	30	36
24	34	30	36
25	33,6	30	36
26	33	30	36
27	33,8	30	36
28	32,9	30	36
29	33,4	30	36
30	33,9	30	36
Average	33,51666667		
Deviation	0,435560149		
CP	1,913245127		
CPK	1,90049016		

Table 2: CPK Calculation

3.5 Software ergonomic and designs

Therefore we based on the ISO 9241. This ISO Norm is a standard of Ergonomics of Human System Interaction. Following rules were proved:

- suitability for the task (the software interface should be suitable for the user‘s task and skill level);
- self-descriptiveness (the interface should make it clear what the user should do next);
- controllability (the user should be able to control the pace and sequence of the interaction);
- conformity with user expectations (it should be consistent);
- error tolerance (the software should be forgiving);
- suitability for individualization (the software should be able to be customized to suit the user) and
- suitability for learning (the software should support learning).

4 Step III: Customer software release (Run@Rate)

For this step we run the line with 100 production units (in the test system we simulate the running of the line with virtual units) and observe the behavior of the software, of the users

Software return reasons			
Problem	Count	Percent of total	Cumulative Percent
Not compatible	21	30,67	30,67
Does not perform as expected	18	18	50,33
Missing hardware resources	15	16	54
Not suitable for self learning	11	14	65,4
Missing user manual	10	7	71
Bad user interface	8	5,33	78
Too complicated	7	4	88,9
Bad cycle time	4	3	98,3
Bad Backup of recorded data	3	2	100
TOTAL	**97**	**100**	**100**

Table 3: Pareto analyze

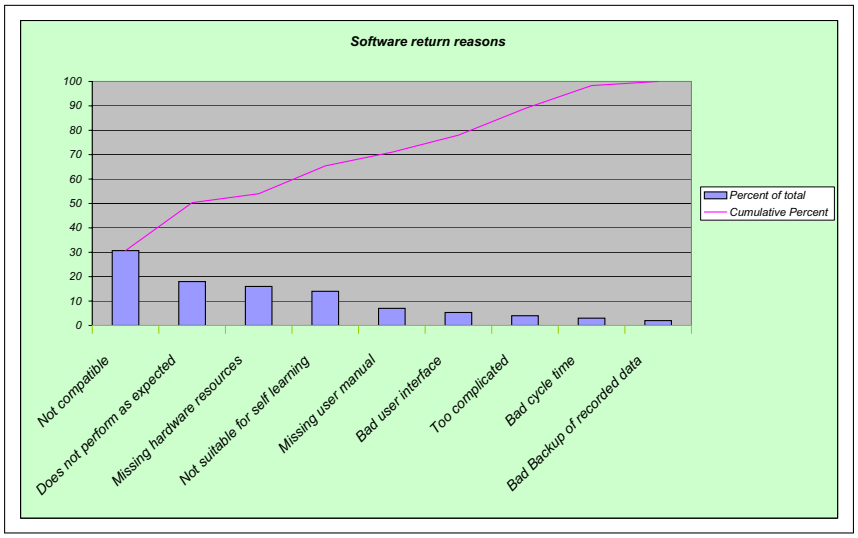

Figure 3: Pareto distribution

(operators) while using it in different situation and intentionally we cause extreme situation (like emergency stop of the equipment, current interruption... etc.) for a better software rating. A statistic is made for each equipment, inclusively the packing station. This statistic contains the number of the parts being processed on each station during one hour.

Also a pareto was realized similar to the **Figure 3**. The following issues were also effectuated:

- Efficiency analyze (how many parts per hour per equipment);
- Defects (bugs) analyze;
- Causes and corrective actions;
- Reevaluation procedure after corrective actions;
- Creating reports (like 8D report).

5 Step IV: Validation and verification of the implemented software package

After release and after the customer agreed the implemented software, the final release is made by the project manager. That means that the software is officially registered as a production software. At this level the project manager analyze the robustness of the software package by making some research about the following properties:

- Lifetime (how long the software will run without any modifications or maintenance), ISO 12207

  - Acquisition analyze (how was the software required);
  - Supply analyze (how was the software provided);
  - How it was develop;
  - How is the operating mode;
  - How much maintenance is needed.
- Modifications handling (how will be the modification make and tested);
- Documents like PPAP (product part approval process) and PSW (part submission warranty) are made and signed from the project manager.

6 Step V: Functional follow up of the software during the production

After the software was accepted and installed in the production line, periodical (every 2 weeks) software audits were made to assure that the software is running in conformity with the requirements and the production criteria are fulfilled.

In this software audit the main criteria (see chap. 3.1) are checked again.

Statistics are made for the cycle time, and errors occurred, in order to make the difference between software errors and process error.

The main aim of this step is to find improvements for the software in order to make it more robust and efficient.

7 Step VI: Customer claims

Customer claim escalation plan is also created like in the Figure 4 in order to obtain the best reaction time in case of software error which could appear, and disturb the normal running of the production. This will drive to very small solving and reaction times for eventually downtimes on the line because of software problems.

8 Conclusions

Using the QSPS system, the downtime of the line (containing the 56 production equipments) because of production software is decreased to almost 0%. The target of 0% is only possible when the technical requirements from the customer are done accurate and to 100% complete. Even with incomplete requirements, using the QSPS system the possibility to reach the target of 0% is possible. Therefore the implementation and running of the software should have a much longer period of time running in the pilot environment, to assure a safe lunch in the production. In most all of the cases this is not possible because the production should always start even if some problems are unsolved, because of time pressure. QSPS should eliminate that inconveniences, provoked by time pressure or other factors, due to a much easier structure, better and faster to be used, compared to other systems met till know in other studies.

The advantage of using the QSPS is that during the 3 environments (test, pilot and production), using the rules and ISO norms for each of them, the almost or all troubles which occurred are filtered before running the real production. That means that after the SOP (Start of Production) the interruption of the production line because of the software is almost inexistent.

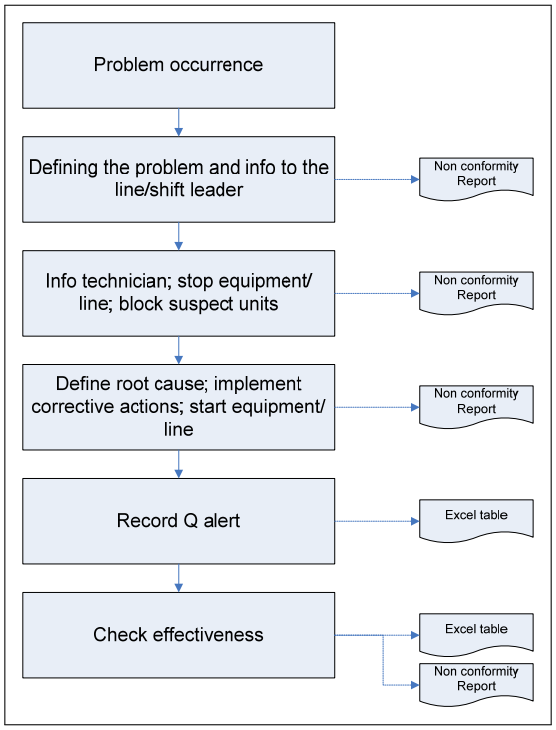

Figure 4: Escalation Plan

Bibliography

[1] Kai-Yuan Cai, Bey-Bey Yin, *Software execution processes as an evolving complex network*, Information Science, April 2008.

[2] Georg Kühner, Torsten Bluhm, *Employing industrial standards in software engineering for W7X*, Fusion Engineering and Design, 2009.

[3] J.-W. Li, *Modeling a quality assurance information system for product development projects with the UML approach*, Vol. 20, Nr.4 of International Journal of Computer Integrated Manufacturing, June 2007.

[4] N. Tudor, C. Kifor and C. Oprean, *Quality System for Production Software - QSPS*.

[5] Oprean, C., Kifor C. V., Suciu, O., *Managementul integrat al calităţii*, Sibiu, Editura Universităţii Lucian Blaga din Sibiu, 2005, ISBN 973-739-034-2.

[6] N. Tudor, D. Dumitrascu, *The Benefits of Project Structuring in Sub-projects and Work Packages*, http://imtuoradea.ro/auo.fmte/files-2008/MIE_files/TUDOR%20NICOLAE%202.pdf.

[7] N. Tudor, D. Dumitrascu, *Advance Estimate Expenses for Project Execution Time*, http://imtuoradea.ro/auo.fmte/files-2008/MIE_files/TUDOR%20NICOLAE%201.pdf.