ABSTRACT

In recent years software for cars adds to size and complexity, and the development is timely, and it is wished that reliability is high simultaneously.

It is wished that even development about Tele-matic system for cars is similar, and we think that spiritual awakening of the experience that I got in the development is very useful for item reliability security of future car use software development.

Software Quality assurance team carried out original assessment about acquisition of software and We were timely and were able to give Top Manager correct information and measures about project management.

1 Introduction

In recent years, software embedded in a car has been increasing its size and complexity dramatically. A total quantity of software developed for TOYOTA auto models expands year by year. The trends are especially prominent since 1997 through 2000 as shown in Fig.1 while a lot of electrical systems have been equipped onboard in standard packages.

That simultaneously means that new features such as ITS and usefulness of driving are brought by electronics systems, and software has been promoted to a main role in vehicle design from a former secondary player who supports safety and environment protection behind scene.

On the other hand from a view of business, software is required to be accomplished in a short term.

In this paper, our empirical approach with our practical knowledge of software design process is introduced to achieve software very high reliability nevertheless under the technical and business circumstances. A lot of software with various aspects from a very high reliable control system to a very complex network communication system, are now developed based on our approach, and the approach also supports to make individual software systems integrated in a coming development.

2.1 Analysis of Current Development Processes Issues

The most important process for improvement is to analyze and understand what are going on and to measure it. It is also remarkable to take actions to improve software quality based on numbers such as program mounts equipped into a vehicle and trouble cases both in development processes and in market.

Estimation of program mounts reveals a target number of necessary human-resources in coming development and makes it feasible.
The number of trouble cases contributes to build up a targeted quality level. Cause analysis of trouble cases makes us possible to deploy practical actions to maintain software development process in a high quality level.

2.2 Constraints and Goal of Software Design Process Improvement

Program size by application domains has been increasing caused by multimedia applications which are outstandingly increasing among others. We supposed the increasing trend of program size and launched process improvement actions in 1997 to deal of explosion of programs. At that staring moment, our strategic goal of software quality, especially focusing on reliability was settled that; by 2001,

- Keeping market malfunctions caused by software than that in 1997 (Quality),
- Saving human resources increasing quantity of development (Cost),
- Shortening software design and development term 10% shorter than that in 1997, synchronized with vehicle development schedule (Delivery).

We also regard a balance of the above three factors to improve total QCD whose formula is as bellow;

\[
QCD(\text{Target}) = Q^a + C^b + D^c
\]

\[
(a, b, c: \text{given parameters according to projects, application, total size, kind and importance})
\]

3. Empirical Methods to Enforce Software Quality

Measuring software is to be started from lower development by our bottom up approach. Electrical components ordered by OEM are tested at the timing according to milestones defined in development processes. Analyzing malfunction cases clarifies their causes and is regarded as a starting point for future improvement steps.

3.1 Acceptance Inspection in Software Acquisition

Inspection sheets (questionnaire) for vendors are used in software acceptance inspection, which inquire development processes, project management, program skills and quality. The questionnaire is consisted of about hundreds items and each item has a weighted score to evaluate development processes, project management, program skills and quality comprehensively.

The date obtained from the inspection sheets is analyzed to find causes of malfunctions. More than a thousand malfunction cases in last three years have been analyzed, where error factors and causes are identified. It is statistically shown that 1 serious-level bug is found with three minor bugs and 10 light-level coding-rule violations imply 1 serious-level bug in our current design processes (see in Fig. 2). A software with many coding-rule violations statistically is predicted to have a serious-level bug. By this estimation formula, code level improvement is justified to contribute final product quality assurance.

![Software Quality Index](image-url)

**Fig2. Software Quality Index**

3.2 Development Process Audit
To analyze the past malfunction for improving reliability of software, and it is important factor analysis technique to locate what process malfunction occurred by. Some effective measures were examined in each development process. This example was a problem about a C coding rule and was able to get quality by devising the standard coding rule as these measures.

3.3 Project Management

Risk analysis of a project is provided when we analyze malfunction information during software development more in detail timely. In particular this technique was effective for large-scale development, and we inflected in G-book (Toyota Telematic System).

4.Reviewing/Improving Development Processes

CMM and CMMI are well-known methods to improve quality level of software development processes. According to our practical activities in TOYOTA in-house development, a particular support team who oversees development teams is very effective to improve process management, project management and engineering process defined as continuant models in CMMI. The team is called as the SQA team and is also charged to take responsibility of information on quality. They analyze malfunctions cases caused by software, then they deploy countermeasures against them. Before establishing the team, process improvement did not contribute quality assurance immediately because quality information was not managed by anybody. The actions by the QSA team make control of a QCD value possible. Hence, it is concluded that manageable of QCD value is equal to manageable of projects.

5.Toyota Activity of Process Improvement for High Reliability Software Development

Former software design methods such as structured design and object-oriented design insist to address high reliability software design. However, from view of design processes, they are not enough to handle with whole of design processes and process management. Design and development processes are closely related with architecture, management process and engineering process. All of them are to be understood as a element of total process system. Target system architecture is directly mapped from design/development organization structure. Each component composing an architecture is regarded as a targeted object of project management. Design deliverables are also to be organized following the same structure as a system architecture. Progress of deliverables is to be strongly associated with management process from view of engineering process.

5.1.Process Management

Design of electronic parts of auto domain is accomplished with limited partners and venders in limited use context. A certain single purposed development without customization is very common. On the other hand, CMM and CMMI are well-defined process models. Although they are very informative for us to establish our own process of an auto domain, they are not enough alone in such circumstances. We introduced a strict process management additional to traditional process models. That is, processes are selected to be specially cared under the strict management by prioritizing processes according to a degree of iteration, business priority and an engineering risk. That makes development team’s performance maximize and it also makes
QCD possible to set under control. Our process management method is proved so effective in our experiments.

5.2 Engineering Process

A technical countermeasure is introduced to improve engineering processes in this section. Leadership of a development manager is a key in verification process because design review led by the development manager is extremely effective. The development manager takes responsibility of a project by approving result of the final design review. In the verification process, statistical coding check is also important. 10% malfunction cases in design phases are caused by violating coding rules. It is shown that particular teams or designers are likely to infract coding rules and their codes have high provability to contain malfunctions.

6. Other Actions to improve software quality
6.1 system architecture

Electronics system in a vehicle is expected to collaborated with each other and social infrastructure via vehicle LAN and access network, so that an auto will provide advanced and sophisticated features hiding complicated implementation. The necessity for a new design method to assure high software quality should be emphasized more than before. System architecture as a deliverable of upper design process becomes important to build up a high quality system. Analysis of a system architecture is a starting point. Standardization is also a key to achieve an efficient and high quality design process. The standardized architecture contains ITS tier which bridges a vehicle to social infrastructure, vehicle manager tier which controls vehicle dynamics from view of a driver and domain manager tier which are independently composed of sub-domains of autos, device tier as program components. Multi-tiered structure enables a system architecture to have high re-usability and to achieve reliability.

6.2 Software Platform

An actual application has to be implemented mixed with higher and lower re-usability components. Software platform is considered as a solution to enhance re-usability of application programs. Components re-usable for many applications are to be located in the platform. Hardware dependency is to encapsulated into inside of the platform, then an application becomes fully device/hardware independent and its re-usability is increased. The software platform will be equipped into all systems so that high reliability is strongly required. Very strict design rules should be applied. High portability is required because it should be tolerance for hardware modification including CPU. Readability is also important to update or port it.

![Hierarchy Structure of Engine Control](image)

7. SUMMARY

High reliability has been achieved in our design and development process by solving problems and by applying countermeasure derived from fundamental causes. Without one of two
approaches, it is impossible to satisfy a quality level for new coming large scaled applications or to deploy any of such products either. It is confident that our practical and empirical approach is the best and the shortest way to provide attractive services for auto users.

REFERENCE
1. CMM, CMMI: Carnegie Mellon University/Software Engineering Institute