Software Reliability Compliance Model for Requirements Faults

CHANDASEKARAN SUBRAMANIAM\textsuperscript{1}, KANCHANA NATARAJAN\textsuperscript{2}
Department of Computer Science & Engineering\textsuperscript{1}, Department of Information Technology\textsuperscript{2}
Kumaraguru College of Technology\textsuperscript{1}, Bharathiar University\textsuperscript{2}
Coimbatore, INDIA
chandrasekaran_s@msn.com\textsuperscript{1}, kanch.bu@gmail.com\textsuperscript{2}

Abstract: - The objective of the paper is to propose a reliability compliance model focusing on the requirement faults or non-compliance of the requirements during software process and product. The reliability compliance is referred with respect to the specification based on the existing standards. In the proposed work, the model of compliance checking along with the parameters of the requirements for reliability enhancement is addressed. The reliability of any software based system can have a direct function on the degree of compliance with the existing standards stipulated by the standards committee related to the domain of operation. The non-compliance that may be intentional or non-intentional with respect to expected usage and needed development requirements affects the processes and overall product reliability. The architectural level compliance solution in meeting the existing standards of the software executable components or services is proposed with the help of Correct Timed Current Function (CTCF) Logic for the compliance model checking towards reliability.

Key-Words: - uncertainty, reliability compliance, standards, specification, intentional, product requirements, model checking;

1 Introduction
The software quality management incorporates quality control and assurance activities which is the core area where the customers, developers, management and the government authorities meet. Each and every actor playing different role in the quality management seek satisfaction and conformance to the requirements of the developed software. Since the arrival of many application software in different domains, their compliances with the existing standards need to be analyzed. Different forms of analyses have been carried out in the compliance requirement engineering. The initial work focused on these compliance checks and grouped them into two categories namely, error identification and prevention through process management techniques [1]. The compliance model is having a direct impact on the reliability of the software application in eliciting the exact requirements. The compliance requirements are emerging not only from the customer side but also from all the stakeholders’ side in the design and deployment of these applications. Each and every stakeholder has their own view regarding the standards on the specifications of the requirements which is culminated by uncertainty and vagueness. The other important sources are the legal, statutory, regulatory and contractual requirements to which an organization, its trading partners, contractors and service providers have to comply [2]. Within the software development organization, based on the business process domain, there are many forms of compliances to satisfy the regulations or recommendations of the concerned authorities. Regulatory compliance complicates the collaborative development environment since the obligations are to be refined by many personnel across an organization. Given a non-compliance associated with a specific decision chain involving a third-party product, both the organization and third-party can determine the role played by an entity in the case of a fault. The product may have been inappropriately aligned with personnel obligations they could not reasonably satisfy all the compliance requirements [3]. The compliance requirements vary between different business domains and might even deviate for different enterprises working in the same domain. Further on, two process activities might also be required which may be mutually exclusive to each other, i.e., they are never executed both in one process instance. When a compliance requirement is not satisfied, an approach to provide feedback in a comprehensible way is needed. That is, anti-patterns are derived from the compliance patterns [4]. Therefore a compliance centered reliability approach can be adopted to enhance the functionality and accuracy of the software product amidst a lot of uncertainty and fuzzy data. The other quality factors like safety and security can also be included if the proposed compliance centered approach is extended to those factors in such applications. A recent survey of IT tools based on citation analysis with a focus on privacy and security shows that regulatory compliance is one of

the key requirements that need to be taken into account [5]. In addition to reliability compliance factor, the process of achieving compliance with security regulations is a significant undertaking while developing and maintaining critical software systems. The reliability and security compliances are centered towards the functionality of the software and thereby the organizations now perceive regulatory compliance as a primary driver of life cycle security efforts for their critical software systems and infrastructures [6]. The software reliability compliance model must address the corrective actions for the requirement faults or frauds. Hence the other side of reliability compliance model is to frame disciplinary actions that handle all sorts of non compliances to rules and regulations leading to juridical pursuits as financial scandals [7]. More recently, it became clear that the absence of proper policies, regulations and controls are one of the factors that caused the subprime mortgage crisis which resulted in government bailouts of financial firms, bankruptcies or selling of banks at fire sale prices [8]. The main focus of the work is to propose a software reliability compliance model focussing the regulatory and standard requirements needed for the process certification and product quality assurance.

The paper is organized as follows: Section 2 illustrates the relationship between the nature of errors due to non compliances in the procedure and the expected reliability of the software product. The sub section proposes a compliance flow model to identify the various errors in different phases of compliance checking. Section 3 explains the reliability compliance factors on the software requirements with a set of proposed metrics. Section 4 brings compliance centered logic towards the functional correctness and timed actions on the requirements and explained with sample statements to arrive at a satisfiability based decision in requirement fault detection. Section 5 concludes the work with the verification results of the proposed model and the future works to integrate the idea in the final product documentation phase with its limitations.

2 Reliability Compliance and Error

The reliability of a product or service is the probability of fault-free operation during a specified period in a given environment. The feel such a fault free operation in an environment connects the compliance which is a sub characteristic of the functionality of software as defined by IEC 9126. The fault free operation does directly or indirectly imply that the outcome will be an acceptable one if and only if it is not only based on the user expectations but also based on the development standards. For example, while developing financial software that calculates income tax, the customer may want some of his income to be hidden towards the payment of taxes but the design of that software which calculates the net tax against his income should be legally compliance with the current regulatory standards. The reliability of such financial software is totally evaluated based on the incorporation of many such standards in compliance to gain acceptance not only from the user but also from the government. The compliance-oriented software life cycle now becomes security development lifecycle and model based design concentrates the quality factors in the run time environment which stipulates the essential need to consider the standards and policies. From all these scenarios, it is evident that the compliance becomes an essential functional requirement for any safe and secured software product or code development process. The non compliance is becoming an important factor in elevating the unreliability through specification faults in the requirement phase leading to reliability compliance. At the same time, the amount of uncertainty is to be taken into consideration while checking the requirement compliance of the specifications enumerated by the customer and developer since uncertainty is a quantification of the doubt about the measurement result. The intentional and unintentional compliance is beyond the scope of software engineering paradigm since it is more on the social responsibility of the organization who developed those trust driven services. The purpose of non compliance that is either due to intentional or unintentional failure to comply or refusal to comply is not dealt here. But the term “noncompliance” is to mean that the person or organization refuses to follow an instruction or request, or obey a rule. The organization is aware that the enforcement giving the instruction is a proper authority figure, understands the standard, remembers the instruction, it is able to control impulses, with adequate initiation ability, and is not especially slow in formulating responses. It may even be a rightful approach to think about this adherence that is nothing other than reliability compliance. The objects of reliability compliance consist of people, safety, regulatory, labour and eco-friendly that have several attributes incorporated with the standards. At the same time the object regulatory has several attributes such as authority, legal and financial compliance standards. As per ISO/IEC 9126-I Quality Model, the compliance becomes a sub characteristic of the quality
“functionality”. In the case of collaborative applications, trust and privacy can also be considered as the sub characteristics of the security which is also a sub characteristic of “functionality”.

3 Reliability and Compliance on Requirements

The main purpose of software quality assurance is to maximize the common part of the quality attributes needed, specified and realized. To reach this target, specific activities have to be performed during all software life-cycle phases. Many of these activities can be derived from hardware quality assurance tasks, in particular regarding preventive actions (defect prevention), configuration management, testing and corrective actions. However, auditing software quality assurance activities in a project should be more intensive and with a shorter feedback than for hardware.

In general the various views of compliance can be represented in the form of a pentagon as shown in Figure 1. In compliance centered approach, the macro entities like procedural compliance, regulatory compliance, people compliance, process compliance and policy compliance are treated as the outer components. These can be realized through the certifications in the respective activities shown in the inner micro entities. That is the micro entities can be determined with the relation of legal, regulation compliance with relation of standards, and people compliance can be related with training, process with the relation of quality and management can be related with policy.

The business software and many other domain software even they are reliable, they have to be compliance with the existing standards. The intentional non compliance against the expected regulations or standards can be viewed as severe non compliance faults by which the other third party legal software components cannot collaborate. The distributed systems will face many shortcomings in the integration of large number of heterogeneous components due to the important quality called “compliance”. Apart from performance and other individual quality factors needed for the system, the compliance with the regulations and compliance with policies of the federation in the case may be, become the critical requirement of software based system. The measure of non compliance may be considered as an acceptable standard deviation of the total specifications as stipulated by the standards. These standards are recommended by the respective committee during the particular enforcement-action governed by the authorities and also depend on the previous deviations as audited by the team of inspectors.

All the micro and macro entities are to be realized to achieve the requirements of the software product and process. Hence the requirement compliance is comprised of five layers such as: cost layer, certification layer, action layer, requirement layer and policy layer. These layers are interdependent and form a basis for processes to complete the tasks as shown in Figure 2. The different Forms of Compliances (FoC) are proposed and non compliance in meeting the various standards driven by regulations are computationally specified. The enforcement actions are parameterized and the certainty factor in determining the reliability using correct timed current function logic is discussed below.
4 Correctly Timed Current Functionality (CTCF) Logic

The quality of the software process and product are directly and indirectly connected with the functionalities of the software. In the behavior and architectural approaches, the model behavior and architectural connectivity can only be realized through respective software functionalities. In the case of safety behavior, for example: the component which sensed the external event must raise functionality namely “alarm”. Similarly in an architecture, the interface between one layer and the other layer of software components must exhibit connect( ) and disable( ) function calls to achieve data transfer. Hence in this approach, the quality can be realized through special functions which are the main reasons for the foremost quality factor ‘reliability’. To achieve this reliability compliance, the time with which these functions are completed, the history of total number of similar functions completed or in-completed and the relative distance between one function to other are to be focused in a formal sense. In the semantics of classical logic, the dynamic nature of the any external incident may be a security breach and its duration with the unwanted actions cannot be fully explored so as to reason the future security strategies. According to Arthur Prior, a time-dependent notion of the truth values has to be applied not only for the past but also to the latest incidents. The temporal logic representing any particular interval of time or the exact point of time are to be specified so as to manipulate the future functions in a formal way. The Linear Temporal Logic and Computation Tree Logic are used to capture the different aspects of computation. The work proposes Correctly Timed Current Functionality (CTCF) Logic in which the time operators like before, after, during, since and until are in the correct time category and operators like just now, just before, just after, just before} denoted as JNi, JAi, JBi as shown in Figure 3. The first order logic (FOL) allows the representation of the statements in a very generic manner; it supports only a limited number of specific conditions without focusing the urgent operations and its criticality.

‘if the audit process identifies that the training transfer is not as per the documents DS submitted and the documents were transferred into other database, then that change CP is not permitted as per the standards S and so forth the employee recruitment ER cannot be accepted”.

The statement can be represented as,

\[ \text{Mandatory}::=\{\text{audit(Tr DS)/document transfer(database)}\} \rightarrow \{\text{Change(CP)} \rightarrow \text{Process(ER)}\} \land \text{Standard(S)} \]

This compliance function is a mandatory type since the employee’s recruitment in a company that manufactures life critical products has to be as per the authorities regulations. The number of vehicles provided by the company to its workers may be more than that specified, say 20 in numbers against just 10. But the function that checks this requirement has to be treated as an optional function type.

The public secured domain deals with the academic and business clients where more security flaws can be identified to illegal access of information internally and externally. The attacker can access the information with irrespective time and transmission setup. The elements of the proposed correctly timed current functionality logic (CTCF) are shown in figure 3. It represents the three types of functionalities such as mandatory, optional and required. For example, the function that checks the regulatory compliance is a mandatory whereas the function that checks for any higher salary to workers is an optional one. At the same time, a function that checks the provision of training to the technicians is a required one. Further the mandatory type function can be attached with operators like {never, always, since/until} denoted as Xi, Yi, Si.Ui. The optional type functions can be attached with the operators like {before, after, during} can denoted as Bi, Ai, Di and ‘required’ type functions can be attached with operators like {justnow, just after, justbefore} denoted as JNi, JAi, JBi as shown in Figure 3. The first order logic (FOL) allows the representation of the statements in a very generic manner; it supports only a limited number of specific conditions without focusing the urgent operations and its criticality.

Figure 3: Correctly Timed Current Functionality
Optional:: = \text{Inspect} (\text{conveyance}, 10) \land \text{condition} (\text{conveyance}) \rightarrow \text{approve} (\text{during, 20}) \rightarrow \text{approve} (\text{requirement}).

4.1 Compliance Objects with Relations
The software reliability can be assessed through the compliance checking of requirements other than functional and non-functional attributes. This can be achieved at different levels by the proposed object-compliance relation (OCR) tool which is shown below in Figure 4.

Figure 4: Relations of compliance objects with compliance flow

Some of the compliance flows representing requirements are shown below:

Compliance Flow 1:
The process compliance flow can be carried out in the process level by the corresponding objects and their relationships. For example, standards, specifications, regulations, recommendation and enforcement are some of the compliance “objects”. Some of the relations are managerial, technical, legal and inspects in the process level. These object-relations can be represented by implications as below:

Managerial
Technical
Legal
|\rightarrow Process.Regulation\rightarrow Process.Enforcement
Non-compliance Inspect
|\rightarrow Process.Recommendations\rightarrow Process.Compliant,

In the boolean Satisfiability perspective, the Satisfiability in the process level with respect to the compliance requirements can be defined as a compliance flow across many set of process variables. This can be written as

\text{SAT @ Process:: = CF1\rightarrow Process Compliance = (Standard \lor Specification \lor Regulation \lor Enforcement \lor Recommendations')}
where CF1 represents compliance flow. Hence, \text{SAT @ Process = (S \lor S' \lor R \lor E \lor R')} where S represents Standard, S' represents Specification, R represents Regulation, E represents Enforcement and R' represents Recommendation sets. Similarly the Compliance Flows CF2, CF3, CF4, CF5, CF6 and CF7 are defined in the same way as shown below:

Compliance Flow 2:
In the product level, the “Standard object” is connected with “Compliant object” through the relation “Release”.

Release
Product.Standard\rightarrow - Product.Compliant,
SAT @ Product ::= CF2\rightarrow Product Compliance = (Standard),
SAT @ Product = (S).

Compliance Flow 3:
In the policy level, the compliance flow can be given by the object-relation (OR) as below:

Managerial Training
Policy.Regulation\rightarrow Policy.Enforcement
Technical Training Conformance Check
|\rightarrow Policy.Recomendations|\rightarrow Policy.Compliant,
SAT @ Policy ::= CF3\rightarrow Policy Compliance = (Regulation \lor Enforcement \lor Recommendations'),
SAT @ Policy = (R \lor E \lor R').

Compliance Flow 4:
In the procedure level, the compliance flow can be given by the object-relation (OR) as below:

Change
Procedure.Standard\rightarrow Procedure.Specification
Re-engineering Configuration
|\rightarrow Procedure.Regulation|\rightarrow Procedure.Compliant,

Compliance Flow 5:
In the performance level, the compliance flow can be given by the object-relation (OR) as below:

Test
Performance.Standard\rightarrow Performance.Compliant,
SAT @ Performance ::= CF5\rightarrow Performance.Compliance = (Standard),
SAT @ Performance = (S).
Compliance Flow 6:
In the people level, the compliance flow can be given by the object-relation (OR) as below:

\[
\text{Deploy} \rightarrow \text{Awareness} \\
\text{People.Standard} \rightarrow \text{People.Enforcement} \rightarrow \text{Ethics} \\
\text{People.Recommendations} \rightarrow \text{People Compliant},
\]
\[
\text{SAT} @ \text{People} := \text{CF6} \rightarrow \text{People Compliance} = (\text{Standard} \lor \text{Enforcement} \lor \text{Recommendations}), \\
\text{SAT} @ \text{People} = (S \lor E \lor R').
\]

Compliance Flow 7:
In the third parties level, the compliance flow can be given by the object-relation (OR) as below:

\[
\text{Disclose} \rightarrow \text{Non-Obeyance} \\
\text{Parties.Standard} \rightarrow \text{Parties.Enforcement} \rightarrow \text{Agreement} \\
\text{Parties.Recommendations} \rightarrow \text{Parties Compliant},
\]
\[
\text{SAT} @ \text{Parties} := \text{CF7} \rightarrow \text{Parties Compliance} = (\text{Standard} \lor \text{Enforcement} \lor \text{Recommendations}), \\
\text{SAT} @ \text{Parties} = (S \lor E \lor R').
\]

\[
\text{SAT} @ \text{Compliance Flow} = (\text{CF1} \land \text{CF2} \land \text{CF3} \land \text{CF4} \land \text{CF5} \land \text{CF6} \land \text{CF7}).
\]

Figure 5: Reliability Compliance Checker to detect requirement faults

The different phases in the requirements compliance flow (CF) can be described and computationally represented as below. Based on the compliance entity and its relation, the compliance flows can be checked with the help of their information like, budget, time and policy rules of the authority. The outputs of the compliance flow checker and the assertion manager helps to decide the degree of compliance or the level of non compliance that leads to a fraud specification using fake standards as shown in Figure 5.

The Boolean values are assigned for each entity to find the satisfiability of software compliances. The compliance flow CF1 can be computationally represented by the corresponding entities which are assigned boolean values. For example, as shown in the annexure towards the process compliances, the string for the standard S is of boolean string 1100 where the leftmost bit indicates the presence of a software process group, the second bit represents the completion of auditing, the third bit represents the completion of reporting, and right most bit represent the completion of review process. Similarly all other compliance entities can be represented by boolean strings. The boolean string for “specification S” is 1100, the boolean string for Regulation R is 1001, for Recommendation R’s is 1010, and for Enforcement is 1001.

Therefore, SAT@Process can be calculated by the logical OR of all the entities and the result is 1111, not considering the ‘notsure’ and “dontknow” conditions. Similarly for compliance flows CF2, CF3, CF4, CF5, CF6 and CF7 are considered and the corresponding SAT values are tabulated as shown in Table 1.

The Boolean values are assigned for each entity to find the satisfiability of software compliances. The compliance flow CF1 can be computationally represented by the corresponding entities which are assigned boolean values. For example, as shown in the annexure towards the process compliances, the string for the standard S is of boolean string 1100 where the leftmost bit indicates the presence of a software process group, the second bit represents the completion of auditing, the third bit represents the completion of reporting, and right most bit represent the completion of review process. Similarly all other compliance entities can be represented by boolean strings. The boolean string for “specification S” is 1100, the boolean string for Regulation R is 1001, for Recommendation R’s is 1010, and for Enforcement is 1001.

Therefore, SAT@Process can be calculated by the logical OR of all the entities and the result is 1111, not considering the ‘notsure’ and “dontknow” conditions. Similarly for compliance flows CF2, CF3, CF4, CF5, CF6 and CF7 are considered and the corresponding SAT values are tabulated as shown in Table 1.

<table>
<thead>
<tr>
<th>Compliance Forms</th>
<th>Compliance Flow</th>
<th>SAT @ Compliance Forms</th>
<th>SAT@ Compliance Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process</td>
<td>CF1</td>
<td>1111</td>
<td></td>
</tr>
<tr>
<td>Product</td>
<td>CF2</td>
<td>1010</td>
<td></td>
</tr>
<tr>
<td>Policy</td>
<td>CF3</td>
<td>1110</td>
<td></td>
</tr>
<tr>
<td>Procedure</td>
<td>CF4</td>
<td>1111</td>
<td>1000</td>
</tr>
<tr>
<td>Performance</td>
<td>CF5</td>
<td>1001</td>
<td></td>
</tr>
<tr>
<td>People</td>
<td>CF6</td>
<td>1111</td>
<td></td>
</tr>
<tr>
<td>Parties</td>
<td>CF7</td>
<td>1111</td>
<td></td>
</tr>
</tbody>
</table>

4.2 Uncertainty in Compliance Flows

The uncertainty is involved due to the presence of “maybe” and “not sure” terms in the compliance forms. To decide the satisfiability in such circumstances, the certainty factor can be used. Generally, with an event E and the corresponding user’s observation E’, the “surely yes” may be given +4 score and “surely no” may be given -4. The “not sure” may be given +2 score and “don’t know” may be given +2 as shown in Figure 6. For example the probability of P (Yes|+4) = 1; P (No|-4) = 1; P (Not Sure|-2) = 0.5; P (Don Know|-2) = 0;
The pseudo probability of certainty factor $R$ can be calculated as per the equation [9] given below:

$$P(E \mid \text{Not sure}) = \begin{cases} (P(E) + (1-P(E)))^{R/4} & \text{for } R > 0 \\ P(E) (1-R^4) & \text{for } R < 0 \end{cases}$$

If $P(E) = 0.8$ then $P(E \mid \text{Not sure}) = (0.8+1-0.8)^{R/4}$

The certainty factor $R$ can be assumed to have any value from -4 to +4. If the value taken for $R$ is -1, then the probability of $E$ can be derived as 1.

$$P(E \mid \text{Not sure}) = (0.8+1-0.8)^{1/4}$$

$$P(E \mid \text{Not sure}) = 1$$

Similarly based on the observations the probability for each event has been calculated which is shown in Table 2.

Table 2: Certainty Factor and Probability of Event

<table>
<thead>
<tr>
<th>Event</th>
<th>Observation</th>
<th>Certainty Factor (R)</th>
<th>P(E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process</td>
<td>Not sure</td>
<td>-1</td>
<td>0.25</td>
</tr>
<tr>
<td>Product</td>
<td>Don’t Know</td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td>Policy</td>
<td>May be</td>
<td>-3</td>
<td>0.75</td>
</tr>
<tr>
<td>Performance</td>
<td>Not sure</td>
<td>1</td>
<td>0.25</td>
</tr>
<tr>
<td>Procedure</td>
<td>Don’t Know</td>
<td>-4</td>
<td>1</td>
</tr>
<tr>
<td>People</td>
<td>May be</td>
<td>3</td>
<td>0.75</td>
</tr>
<tr>
<td>Parties</td>
<td>Not sure</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

5 Conclusion

The software reliability compliance model focusing on requirements faults is proposed bringing out the fundamental relationship between non compliance aspects and software reliability. The CTCF logic is used in bringing the effect of mandatory and optional requirements with compliance objects and relations. Different compliance flows using object-relation diagram are proposed in determining the satisfiability with requirements compliance and the uncertainty values are also considered. A formal model for compliance requirements and their elicitation towards governance and risk analysis towards reliability compliance is the planned future work. The future work can be extended in making the proposed compliance model for software reliability with requirement errors as a software tool that can be developed and deployed as a software as a service. The limitation in the above work is to make the model as a domain specific like healthcare information system or pharmaceutical compliance checking system. In that way, more number of compliance pathways is to be declared and the reliability can be estimated using rough set theory.

References:


Annexure
<table>
<thead>
<tr>
<th>Process Compliance</th>
<th>Policy Compliance</th>
<th>People Compliance</th>
</tr>
</thead>
</table>
| Whether organization has created Software Engineering Process Group (SEPG) to improve process?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Whether hierarchy followed with any software policy?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Scientists and quality assurance engineers are fully certified?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  |
| Whether auditing has been completed?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Health policy insurance provided for employees in the organization?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Scientists and Quality Assurance engineers are fully experienced?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  |
| Whether reporting and review has been completed?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | International sales, domestic sales are performed for same quality with policy?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | People and Industry are already faced legal problems?  
  a. Severely penalized and abundant  
  b. Warned and corrected  
  c. Instructed  
  d. But no legal action  |
<table>
<thead>
<tr>
<th>Product Compliance</th>
<th>Procedure Compliance</th>
<th>Parties Compliance</th>
</tr>
</thead>
</table>
| Whether product development is motivated in a business environment by SEPG?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Procedural Compliance investigations are conducted to ensure employees following proper protocols?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Quality assurance activities are incorporated with parties?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  |
| Six sigma principles are applied or not?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Whether general loss prevention strategy is applicable?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Standards have been followed or not?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  |
| Product variability has approved or not?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Performance Compliance | Whether third-party comply with regional parties?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  |
| Defective products have been maintained or not?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | New technology adopted to enhance performance?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  |  
| Performance level increase is maintained?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  | Existing technology loss has been incorporated?  
  a. Yes  
  b. No  
  c. May be  
  d. Not sure  |  
|  |  |  |