



## **Exploring Computing Environment Possibilities for Risk Oriented Testing**

**VINITA MALIK<sup>1</sup> and SUKHDIP SINGH<sup>2</sup>**

<sup>1</sup>Information Scientist, Central University of Haryana , Mahendergarh, Haryana,123029, India.

<sup>2</sup>Assistant Professor, Deenbandhu Chhoturam University of Science  
and Technology , Murthal, Haryana, 131039,India.

### **Abstract**

The present research effort analyses various application areas of risk oriented testing and identifies the gaps from the past .We need risk oriented testing not only for identifying risks in the projects but also for the maximum optimization of resources .Our research stresses on risk oriented testing in pervasive and evolutionary computational areas as due to dynamicity of such computing environment ,the project imbibes risk in great measure and needs to be taken care in early stages of the project . Risk oriented testing requires concentration in both of these application areas as extremely little work has been done in this regard.



### **Article History**

Received: 18 June 2017  
Accepted: 05 July 2017

### **Keywords**

Risk, Risk based testing, Software testing Life cycle, Pervasive computing, Evolutionary Computing.

### **Introduction**


Software testing requires selective and careful planning because it can not be done exhaustively. To increase the test effectiveness proper selection of test cases for testing is mandatory. As we know that risky scenarios requires proper identification because of very limited resources ,lack of time for not complete testing of system we need more focus on the areas where largest risks are found out . For avoiding software failures we need proper identification which can be done by several techniques like by brainstorming, experts knowledge ,brainstorming.

In the present paper we discuss risk oriented or risk based testing application domains and then tries to find out the gaps or shortcomings from the past. We have gone through various search engines to find out the relevant articles i.e. Springer, IEEE Explore, IEEE computer Society, Inder Science, ACM digital library.

In this paper, we present results of analysis of various application domains of risk oriented testing.

**CONTACT** Vinita Malik ✉ [is@cuh.ac.in](mailto:is@cuh.ac.in) 📍 Information Scientist, Central University of Haryana , Mahendergarh, Haryana,123029, India.

© 2017 The Author(s). Published by Enviro Research Publishers

This is an  Open Access article licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License (<https://creativecommons.org/licenses/by-nc-sa/4.0/> ), which permits unrestricted NonCommercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

To link to this article: <http://dx.doi.org/10.13005/ojcs/10.03.18>

The paper has been structured in various sections to answer the questions as followed: RQ1: Which application domains for risk based testing has been explored most in the past? RQ2: Which application areas has not been explored yet in risk oriented testing? We have explained the risk based testing application areas used in past in section 2 and then have discussed about gaps in section 3 and finally concluded in the section 4. Extensive survey has been done on risk oriented testing application areas employed in the past and then we have successfully found out the gaps.

**Risk Based Testing Application Areas In The Past**

**Object Oriented Computing Environment**

Various object oriented metrics can be used for identifying the classes having high risk values. These metrics can be Number of methods, weighted method per class , response per class ,RFC/NOM ,coupling between objects , depth in tree ,number of children .Identification of high risk can be done as a class that contains two or more than two metrics which exceed the recommended limits

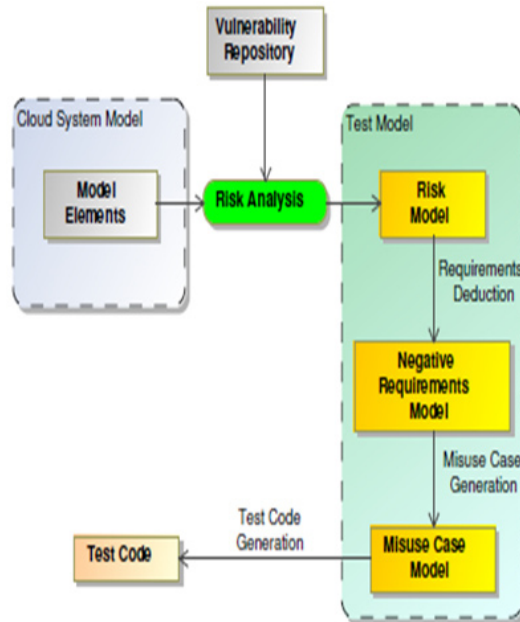
- The purpose is to identify all the high risk classes
- Once the risky classes are identified then testing can be done to lessen the risk effect<sup>1</sup>.

**Cloud Computing Environment**

In cloud computing security testing can be applied on 3 layers which are as follows:

- Service
- Infrastructure
- Platform layer

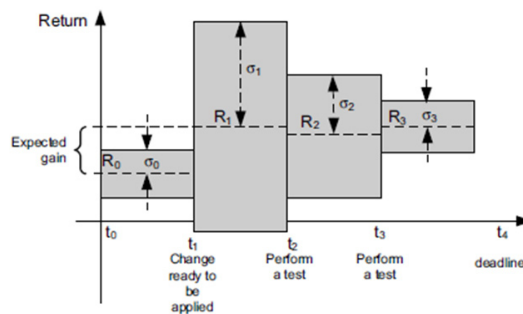
It has be to performed on 2 parties: Cloud providers that offers software as a service and cloud consumers that can develop custom applications .Cloud security can be improved by using risk oriented testing . Negative requirements can be derived by using the risk analysis on cloud under test. Living Models approach can be used which will result in changing risk model by adapting negative requirements. UML can be used for modelling<sup>2</sup> .The following figure 1 depicts the how risk oriented testing can be done in cloud environment.



**Fig.1: Cloud environment risk based testing**

**Services Based Computing**

Various service based computing environment have been used in past for risk based testing. Services may include IT services<sup>3</sup>, Web services<sup>4,5</sup>.For IT services risk profiling can be done which is a function of time. By the help of testing risk profiling can be lowered down .Evolution of risk profile can occur with time is shown below in fig.2.



**Fig. 2: Risk profiling over time**

In adaptive testing of web services we can assume that the tolerance towards a feature’s failure is basically an inverse ratio to the risk .Risky features are tested earlier than others .Ontology based risk calculation is done for the web services .We require run time monitoring system to find out the dynamic changes in configuration of given services.

Failure probability can be evaluated by first estimating the probability of failure of each ontology class and then by adjusting estimation of each class. Again we find out failure rate at interface services also. We do control flow analysis of whole system and do the final failure probability estimation . We quantitatively analyse risk by 3 layers of web services:

- Data layer
- Unit Layer
- Integration Layer

First complexity based analysis is done which is dependent on how much domain ontology model is complex<sup>4</sup> .All control constructs failure probability is estimated as shown in the figure 3 given below .

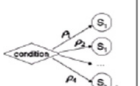
Control Construct (cc) <sup>o</sup>			P(cc) <sup>o</sup>
Category <sup>o</sup>	Graphic Expression <sup>o</sup>	OWL-S Example <sup>o</sup>	
Unconditioned <sup>o</sup>	$!s_1, !s_2, !s_3, !s_4, !s_5, !s_6, !s_7$	Sequence, Split, Any-Order <sup>o</sup>	$P(cc) = 1 - \prod_{i=1..n} (1 - P(s_i))^{o}$
Conditioned <sup>o</sup>		If-Then-Else, Choice <sup>o</sup> , Iterate, While-Repeat <sup>o</sup>	$P(cc) = \sum_{i=1..n} \rho_i P(s_i)^{o}$ Where, $\rho_i$ is the execution probability of a service $s_i$ and $\sum_{i=1..n} \rho_i = 1^{o}$

Fig. 3: Control Construct Failure Probability

Risk assessment can be done in parallel with test development and we need to decide tactics for service ranking<sup>4</sup>. In the following figure 4 we can see how adaptive testing in groups can be done for web based services.



Fig.4: Adaptive testing of web based services

Web services can also be tested by help of fault tree diagrams in which leaf nodes are used to represent the dangerous scenarios<sup>5</sup>. Here we use state model diagram to look for system activities in order to test web transactions as shown in the figure 5 given below.

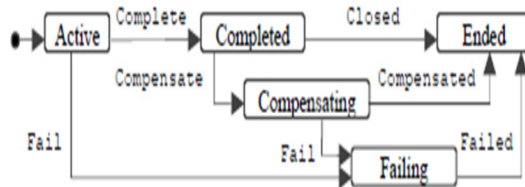


Fig. 5: Activity state diagram of web transactions based system

While drawing system fault tree we need to consider several system properties i.e. Composition, Sorting, visibility, durability, consistency, recovery, controllability and accordingly can draw fault tree diagram which is shown below in figure 6 .

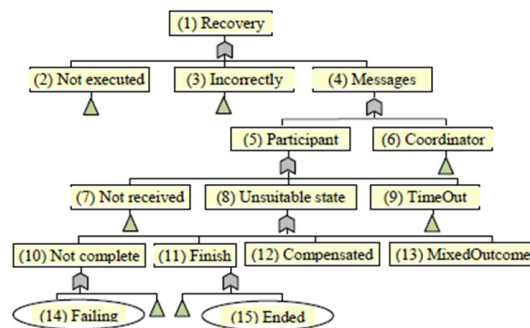


Fig. 6: Fault tree representation

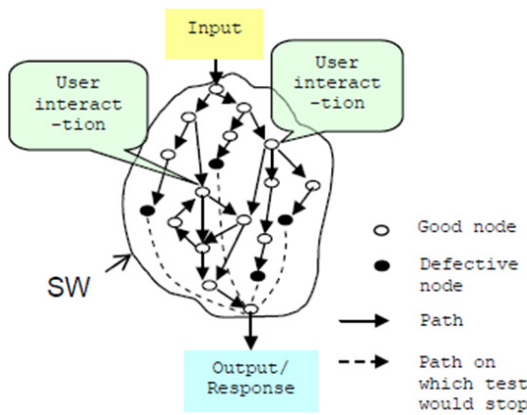
### Software development and manufacturing computing environment

Half of the development time can be taken while manufacturing big machines. Proper test sequencing is to be done which can be done by AND OR graphical method. If the test time and cost exceeds the risk cost then we should stop the testing. For a software test phase its possible to evaluate the optimal test sequence which can do risk reduction in a given time frame<sup>6</sup>. The required amount of subsystem testing can be calculated once the reliability tests can be calculated. Geometric distributed path lengths and Bernoulli type inspection errors can be used for reducing risk in software<sup>7</sup>. Hypothesis testing can be done for Type 1 and Type 2 errors which are shown as in the table 1 given as follows:

		True State of Nature	
		Software is truly defect-free	Software has defects
Your decision based on the tests conducted on the software	Declare that "Software is defect-free"	Decision is correct	Type II error: You have erroneously accepted a defective SW as good
	Declare that "Software is defective"	Type I error: you have incorrectly rejected a good SW	Decision is correct

**Table 1 : Hypothesis testing Type 1 and Type 2 errors**

Node path representation of the software can also help in finding out defects which can be done as shown in the figure 7 given below :



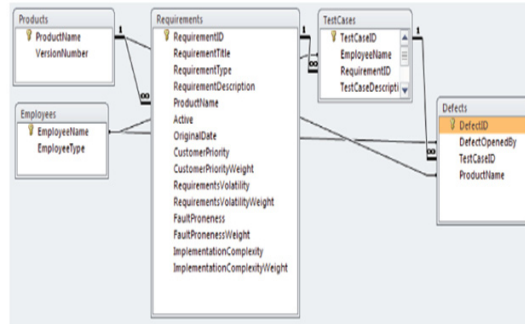
**Fig. 7: (a) To find software defects by node path representation**

**Industrial risk oriented testing Computing**

Test effectiveness can be improved by considering customer priority and the fault proneness. Prioritization of test cases can be done on these factors. Fault detection rates can be evaluated and based on risk value can be prioritized. Risk based prioritization considers risk exposure as the main metric. Requirements volatile nature and implementation complexity also plays very important role in enterprise risk oriented computing environment .We also consider following metrics for the further evaluation which is as follows<sup>9</sup>:

- Prioritization factor value
- Weighted priority
- Total Severity of failures detected

The required database relations can be depicted as follows in the figure 8.



**Fig. 7: (b) Main Database relations**

At the enterprise or industry level security concern is also really high. So threat modelling is also done by capturing high risk during the risk analysis phase by designing security test case scenarios. Abnormal behaviour of system is captured by the help of misuse cases<sup>9</sup>. One example of abuse case template is given in the figure 8 given as follows:

<b>Security Test Case Id -</b>
<b>Description – To check the strength of the authentication mechanism used during logon</b>
<b>Actor Profile – Misuser with knowledge of Password cracking</b>
<b>Preconditions – None</b>
<b>Assumptions –</b> 1. The attack can be performed remotely
<b>Trigger Point – Can happen anytime</b>
<b>Worst Case Impact – Privileged Access Gained</b>
<b>Expected System Behavior –</b> 1. The System expects complex Passwords 2. Three invalid attempts would lock the user’s account.

**Fig. 8: Misuse cases example template**

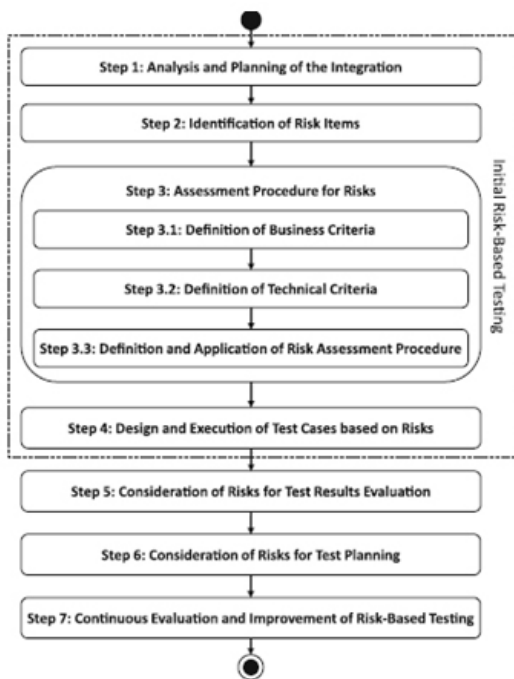
If the risk identification and assessment is synthesized successfully then determination of iteration policies, testing order and comprehensive testing can be done effectively.

In the small and medium enterprises which represent 85 % of software industries we require risk oriented testing for continuous process improvement. Mainly risks involved are project risks, business risks and product risks. Only large enterprises have risk complete understanding based on its definition which include probability and impact. More objective

oriented process are followed for risk management only for the large enterprises not for the small ones. Informed decisions can be taken if risk information is not vague in nature<sup>10</sup>. Risk based testing can be integrated in the industrial process<sup>11</sup>. Steps for the same is given below:

- Plan the Integration Process
- Identify all risk factors
- Assess all risky items
- Design and execute test cases
- Test planning by risk consideration
- Continuous improvement

Steps for the same are shown in the figure 9 given below:

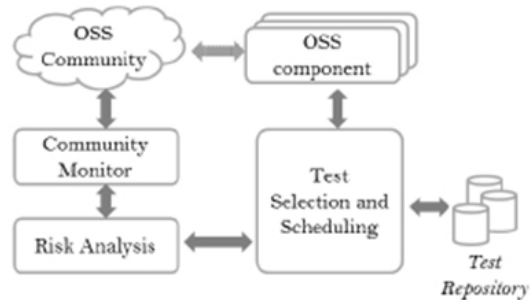


**Fig. 9 : Integrating risk oriented testing in industrial test process**

Risk concept is dependent on context and scope in industries. Even if testing does not incorporate risk explicitly still it plays vital role in implicit manner also. To design the new test cases and for further classification understanding prioritization is also. Information about risk can be used for extension of scope of testing for the risky areas as here most of the critical issues are located<sup>12</sup>.

**Open Source software risk oriented testing Computing**

To handle quality risks of open source software first data pre-processing takes place by community detection then monitoring of community takes place out of which anomalies are detected<sup>13</sup>. The figure 10 given below indicates how risk oriented testing takes place in open source software.

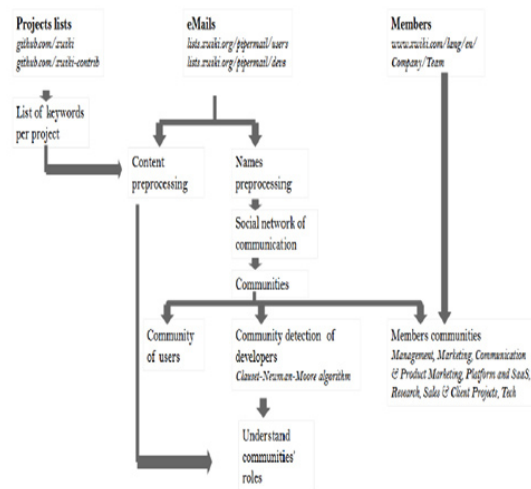


**Fig. 10: Open Source software risk oriented testing**

The community dynamics can be monitored by detection of anomaly between the members of community. Social network of communication is created which contains following as nodes:

- Team Member
- Developer
- User

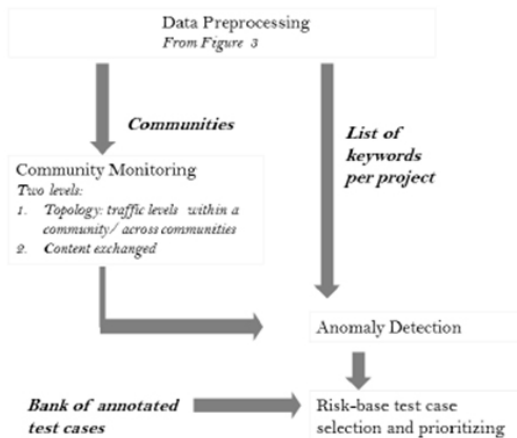
How can be detect communities is shown below in the figure 11 given below:



**Fig. 11: How communities can be detected**



Based on data processing by community detection we can further apply risk oriented testing as shown in the figure 12 given below:

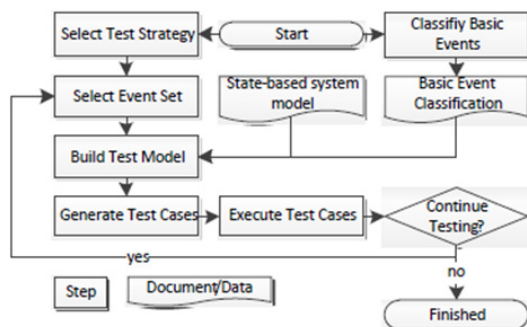


**Fig. 12: Risk oriented Testing by test case selection and prioritization**

Similarity based risk measurement has been applied in past where the execution traces if similar to past failing test cases then it means degree of risk is really high in the project. The execution traces can be mined by Daikon tool<sup>14</sup>.

**Embedded Software Computing and Risk Oriented Testing**

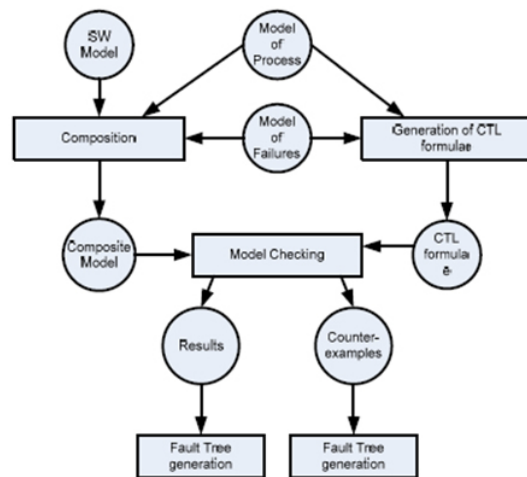
Fault tree analysis can be done on embedded systems and accordingly test cases can be derived, selected and prioritized as per severity level identified. Safety analysis can be done by fault tree analysis in which we have a set of events which are subset of a cut set of the fault tree<sup>15</sup>. WE have the following type of test process in such cases as shown in the figure 13 given below :



**Fig. 13: Risk based test process**

**Networks based risk oriented computing**

Various models of system can be utilized for composition of system models .Safety and availability information can be used to model the artifacts of system. Networked system is made of the various components that may have different or common object of control<sup>16</sup>. Operating modes of the system can be of several types i.e. Initializing, clean up ,emergency ,deadhead .Following in the figure 14 is given how fault tree can be constructed and test cases can be generated .



**Fig. 14: Test case generation by fault tree analysis**

**Quality oriented Risk based testing**

Active continuous quality control can be implemented by automata learning based modelling for prioritization of critical concepts. Alphabetic models are used to steer quality control process to increase the risk coverage. For risk minimization we need to maintain test models, instrumentation, execution environment and test evaluation procedures<sup>17</sup>. Modelling layers for automatic quality control are given as follows in the figure 15:

Quality evaluation based on quality model i.e. QuaMoCo can be used with risk oriented testing<sup>18</sup>.This model also operated on ISO 25010 by providing tool support for quality assessment .Product quality modelling<sup>19</sup> by ISO 25010 standard has been depicted following in the figure 16 as shown below .

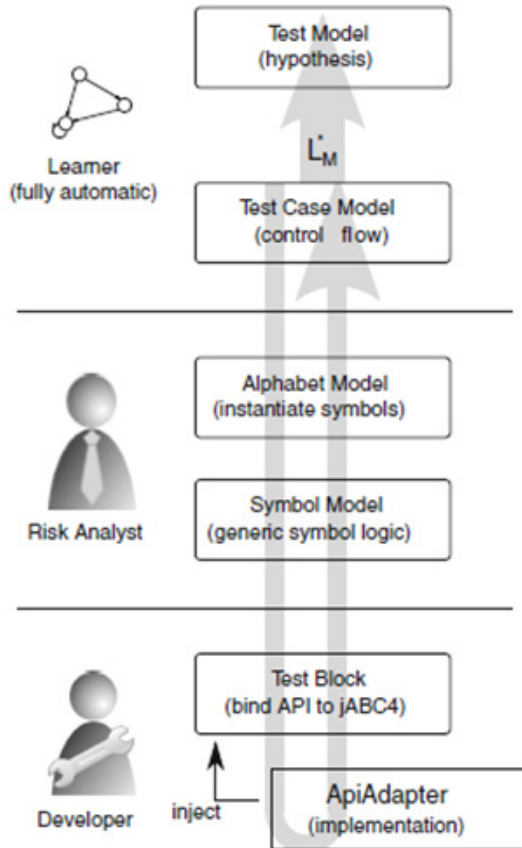


Fig. 15: Quality control modelling layers

**Discussion and findings**

Risk oriented testing has been explored in several areas but no work has been done in case of pervasive computing and very little concentration has been done on evolutionary algorithms.

**Pervasive Computing Environment**

Also named as ubiquitous computing, pervasive computing environment is highly dynamic and evolutionary in nature. Risk oriented testing has not been applied yet in such type of environment. Such kind of computing is really complex in nature as we should be aware of user intent, high level energy for management, proactively handling and

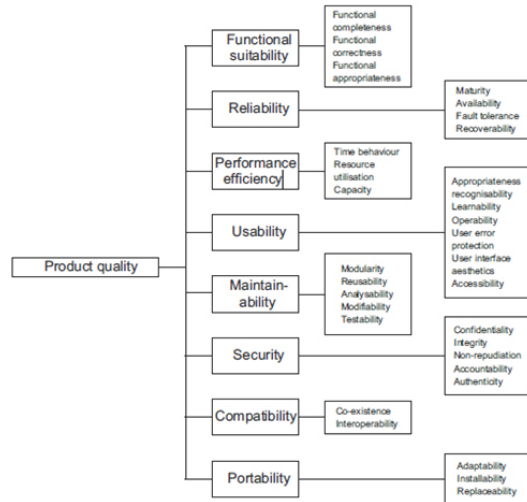


Fig. 16: (a) ISO 25010 quality model

transparency is also required<sup>20</sup>. We can combine distributed computing and mobile computing to form pervasive computing concepts which should be taken care while considering system risk levels for testing as shown below in figure 16 .

Risks should be explored based on challenges in computing environment that include heterogeneity in environment, context awareness, access control, privacy , mobility, data communication and trust<sup>21</sup>.

**Evolutionary Algorithmic computing Environment**

Evolutionary computation that is required for automatic design, problems optimization and machine learning and where selection of individual takes place by evaluation of fitness function. Various risks factors include for testing is finding optimal solution ,computationally expensive and requirement for parameter tuning .Evolutionary computation has also not been explored in risk oriented testing due to very dynamic and evolutionary nature of problems .

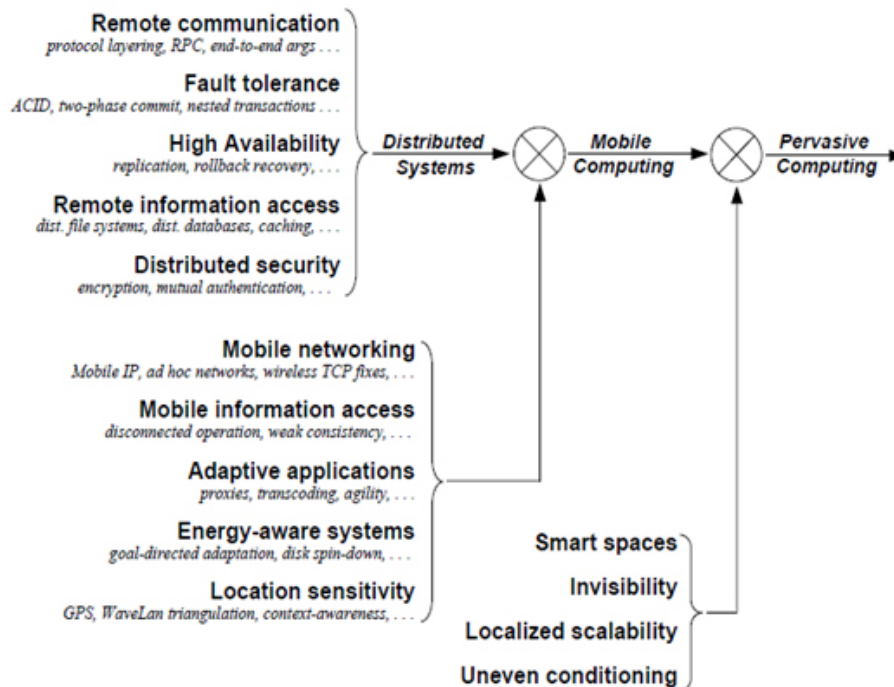


Fig. 16: (b) Pervasive computing Issues

### Conclusions and future work

Risk oriented testing computation has been explored in Object oriented, cloud computing, service based computing , software development , manufacturing process ,industrial ,open source software ,model based systems , embedded systems , network

based systems and quality oriented systems but no work has been done on pervasive computing and evolutionary computation .So we propose work on both of these computational paradigms and risk oriented testing is required in early stages of the system for proper risk management of the system .

### References

1. L. Rosenberg, R. Stapko, A. Gallo, "Risk-based object oriented Testing", Proceeding of the 24th Annual Software Engineering Workshop, NASA, Software Engineering Laboratory, pp.1-6, 1999.
2. P. Zech, "Risk-based security testing in cloud computing environments, Proceeding of the Fourth International Conference on Software Testing, Verification and Validation (ICST'11), IEEE, New York, pp.411-414, 2011.
3. J. Sauve, "Risk based Service Testing, Several approaches to the application of risk to IT Service Management", IEEE Xplore , pp.106-109, 2008.
4. X. Bai, R.S Kenett, W. Yu, "Risk assessment and adaptive group testing of semantic web services", *Int. J. Softw. Eng. Knowl. Eng.* vol.22(5), pp.595-620, 2012.
5. R. Casado, J. Tuya, M. Younas, "Testing long-lived web services transactions using a risk-based approach" , Proceeding of the 10th International Conference on Quality Software (QSIC'10), pp. 337-340.IEEE, New York, 2010
6. R Boumen, "Risk-Based Stopping Criteria for Test Sequencing, IEEE Transactions on Systems, Man and Cybernetics", Part A : Systems and Humans, Vol.38, pp.1363-1373 , 2008
7. T. Bagchi , "Models for software defects and



- testing strategies”, SIGSOFT engineering notes, vol.34, pp.1-4, 2009.
8. H. Srikanth, “Requirements based test prioritization using risk factors: An industrial study, *Information and Software Technology*”, vol.69, pp.71–83, 2016.
  9. K. Murthy, K. RThakkar., S. Laxminarayan, “Leveraging risk based testing in enterprise systems security validation”, *Proceeding of the First International Conference on Emerging Network Intelligence (EMERGING’09)*, IEEE, New York, pp.111–116, 2009.
  10. M. Felderer, “Risk orientation in software testing processes of small and medium enterprises: an exploratory and comparative study, *International Journal of Software Quality J*, vol.24, pp.519–548 ,2016.
  11. M. Felderer, R. Ramler, “Integrating risk-based testing in industrial test processes”, Springer Science+ Business Media, *Software Qual J*, pp.22, 543–575, 2014.
  12. M. Felderer, “A multiple case study on risk based testing in industry”, *International Journal of software tools technology transfer*, vol.16, pp.609-625, 2016.
  13. I. Yahav , “Risk Based Testing of Open Source Software”, IEEE 38th Annual International Computers, Software and Applications Conference Workshops, pp.638-643, 2014.
  14. T. Noor, “Test Case Analytics: Mining Test Case Traces to Improve Risk-Driven Testing”, *International Journal on Software Quality Assurance , IEEE Transactions*, Vol.9(3), pp.119- 127, 2015.
  15. J. Kloos, “ Risk based testing of safety critical embedded systems driven by fault tree analysis”, IEEE, 2011.
  16. T. Hussain, “Automated fault tree generation and risk based testing of networked automation systems”, IEEE, 2010.
  17. J. Neubauer, “Risk based testing via active continuous quality control”, *International journal of software tools technology transfer*, Springer, 2014.
  18. H. Foidl , “Integrating software quality models into risk-based testing”, *Software Qual J*, Springerlink.com, 2016.
  19. ISTQB: Standard glossary of terms used in software testing. version 2.2. Tech. rep., ISTQB, 2012.
  20. M. Satyanarayanan, “Pervasive Computing: Vision and Challenges”, IEEE Personal Communications, 2001.
  21. L.Bhaskar, “Pervasive Computing issues, applications and challenges”, *International Journal of Engineering and computer science* , 2013.