Enhancing Software Secureness in Public ICT Applications

C. K. Raju
Indian Institute of Technology,
Kharagpur, 721 302
INDIA

P. B. S. Bhadoria
Indian Institute of Technology,
Kharagpur 721 302
INDIA

ABSTRACT
Issues related to security and privacy of information under processing have been topics of great public interest. A perception of existence of an insecure channel of communication is usually created, which needed attention by experts. Most discussions also assume, among other issues, neutrality of the operating environment under which the principles of security or control for privacy are applied. Additionally, a standard image of a sender and receiver is posited to convey neutrality of the human agency involved in trans-reception of secure information. This article purports to view contradictions in the theme of security, when neutrality of software environment is contested or when the interests of human agency involved in trans-reception of information are in conflict. A shift of this nature is necessary because a proprietary software environment may be completely transparent to its developer community, even while remaining opaque or insecure to its user community.

General Terms
Software Security

Keywords
Software Secureness, Public ICT applications.

1. INTRODUCTION
Secureness of software has been discussed and debated on identical terms with security of software. Why security-related issues occur in the realm of software development have been put forward by various researchers, and various approaches on how these could be mitigated too have been discussed. A few prominent ones are reproduced. The presence of software defects has been identified as one prominent reason for software to lose its security, allowing intruders or attackers to take advantage of the vulnerability [15]. There had also been studies where vulnerability to software security had been attributed to inadequate methods in hardware design [13]. Indications are given in such studies whereby vulnerabilities would be reduced by following proper design principles in hardware. There are also another set of studies pertaining to software security which wants the project manager or software developer to constantly keep thinking like an attacker or intruder so that the vulnerable areas get visible early onwards.

In one such study [11], it was proclaimed that keeping such an orientation throughout the software development life cycle, improved security in software could be attained. Software processes differ from most other manufacturing processes, in that the products and processes are capable of getting modified, tested or developed by communities of users, maintainers or developers. The users are usually not involved in the process during which the attributes for quality are tentatively fixed for the software to be developed or modified. This means that the user community is relegated to lower priority supervisory role, where the developer community gets to decide most of the prerequisites of quality that is to be self-imposed. In a citizen-centric democracy, software applications that serve public needs, provides a pivotal role to citizens both as users of their own information and as its masters, a proposition mooted in Republic of Peru [23]. Thus, the human agency that ought to stipulate conditions for quality and other related parameters cannot be confined to choices from developer community alone. Additionally, the software environment under which software is developed and verified for secureness also needs to be neutral without having any inherent biases.

A proprietary software environment is often available as closed binaries, without the facility to do any kind of meaningful inspection. In public ICT (Information and Communication Technologies) applications, this may be quite inadequate, given the dual role that citizens get to involve in applications that process public information. It is in this context that this article attempts to understand the concept of software secureness. Once the relationship between software secureness, the role of software environment and varying kinds of human agency get known, it may be possible to devise means to improve software secureness for such public ICT projects. This work attempts to define and view secureness through correctness of software sources, fair implementation of protocols and through the nature of data formats that the software projects use. Initial reviews on software quality measures on software products had prompted detailed studies on issues like as-is utility, portability and maintainability [1]. Significance of portability and maintainability has also been stressed in another software product quality model suggested by Dromey [4]. This situation presupposed availability of software as sources and details of protocols, as without accessibility to such sources, maintainability could be ruled out and portability gets impossible.

Manuals in software quality make adequate references to adherence to use of data standards, fair implementation of protocols and transparency in coding of their implementation [8], [21], and [7]. These manuals, while laying out specifications for achieving software quality, however, do not insist on the dichotomies that arise while applying quality conditions to user and developer domains. While developers working within proprietary software establishments can claim software quality by adhering to these manuals by virtue of

having accessibility to software design and development processes, a majority of users who are out of purview of the development establishment and who would be the major consumers of software, are not in a position to guarantee software quality on their own. This unique contradictory situation insofar as the software users are concerned has even prompted a suggestion that open standards without insistence of open source software would render the whole claim of software quality inadequate [22]. Software sources often undergo changes while catering to evolving demands. Its management too would get complex, when the communities of users, developers or maintainers are allowed varying degree of permissions or restrictions in their access to software sources.

Adherence to established protocols also requires that software is programmed to do, what it claims to do. Even more importantly, software should not be doing, what it is not supposed to do. The task of validating secureness, therefore, needs a fair amount of expertise in programming by the inspecting agency. It is known that errors or deviations from norms in software can be brought out if more people are allowed to inspect the sources [20]. Therefore, access to software sources is a critical factor in establishing secureness of software, especially for those software applications serving public interests or needs.

The properties and nature of data formats used in public applications also need to be scrutinized for their linkages with software secureness. Data formats could be either standards or open formats where ownership has been relinquished. Data formats could also be proprietary which may have owners. When ownership of proprietary formats are in private possession, encoded information risk getting under perpetual control, especially if the private owners shun efforts to convert them into legitimate standards relinquishing ownership. In a draft legislation introduced in Republic of Peru [23], a few issues were referred. It was argued that public agencies have a natural obligation to guarantee permanent availability of information encoded, processed and stored while engaging with software applications. To ensure this, proprietary formats were not desirable for public systems. The significance, as per the bill was due to the fact that private information of citizens gets processed in such software systems, and the state as a legitimate custodian of such information has an obligation to safeguard its availability. Hence usage of data standards or open data formats which do not have owners needs to be mandatory part of any public software initiative. Software that has the data formats encoded in open data formats or data standards will enhance its secureness, as these could be retrieved or made available at any time in future.

Accessibility to software sources allows inspection of the sources for fair implementation of software protocols and coding practices. The draft bill had promoted use of Free Software in public institutions [23]. A study [22] on effectiveness of open standards had pointed out that unless the implementation of such standards are carried out with open source projects, a precarious situation involving vendor lock-in might follow. In a case taken up for study here, the twin requirements that deal with accessibility to software sources and adherence to established data standards or open data formats were scrutinized and their suitability examined towards enhancing software secureness. The public software application is one that monitors a rural employment guarantee scheme introduced on a national scale in India.

2. A RURAL EMPLOYMENT GUARANTEE SCHEME

A government initiative to guarantee 100 days of employment on an annual basis to all rural households in India was legislated [5] in 2005. Commissioned as National Rural Employment Guarantee Scheme (NREGS), the programme was open to all households whose members were willing to offer unskilled labour. Though initially the programme was implemented in select areas, it later got extended to all rural areas of India. The programme, rechristened as Mahatma Gandhi National Rural Employment Guarantee Scheme (MGNREGS), continues to be executed through all local self-government institutions in the Panchayat Raj System which predominantly addresses rural population. The enactment was subsequently amended to place all in- formation about the scheme in public domain through a website. It later became a mandatory requirement [6] for the purpose of introducing transparency in all the transactions within the system. This monitoring scheme which has already commenced is planned to be in operation at over 240,000 rural self-government institutions in India.

The software that fulfills this requirement has been developed by National Informatics Centre (NIC) and is made available to the rural local self-government institutions. Here, the data processed at rural local self-government institutions spread across the country will be received and stored at a central database repository. NREGASoft, the software developed for monitoring these activities is capable of operating in 'online mode' as well as in 'off-line mode' [16]. In the online mode of operation, a dedicated internet connection needs to be established between the local self-government institution and the Ministry of Rural Development (Govt. of India) which hosts the central server. In the online mode, details of all activities are updated on a daily basis with the help of a browser application at the nodes. However, due to the enormity of data, the data-entry operations which even include marking of attendance of the workers at the various work sites are carried out in the off-line mode.

In the off-line mode, data related to MGNREGS are entered by local self-government institutions and updated in a local database repository. Later, at a convenient time or from an alternate location, the incremental updates to local database are synchronized with the remote central repository, which is housed in the premises of Ministry of Rural Development, Government of India. NIC has developed a web-server application integrated with a hypertext scripting engine with the central database server [8], which allows 'online mode' of operation. According to its principal developer [14], the first major award bagged by the project was Microsoft e-Governance Award 2006.

3. SOFTWARE SECURENESS

3.1 Role of Software Environment in Determining Software Secureness

On analysis of NREGASoft it was observed that the central server which received information from rural local bodies was configured using proprietary software. The information received was stored in database in a proprietary format.

Table 1. Ownership of Client Software Sources (Offline)

Software	Nomenclature	Ownership
Operating System	Windows XP SP-2	Microsoft Inc
Web Server	IIS Server	Microsoft Inc
Database Management System	MS SQL Server 2000	Microsoft Inc
Application	NREGASoft	NIC

The minimum essential configuration for becoming the client of the monitoring network, as per the manual [16], is listed in Table 1. It can be seen that a software firm has exclusive ownership over the software environment which embeds the application software developed by NIC during execution. Users of this rural software application do not have access to software sources that are owned by this software firm, and hence software secureness of the environment for the users gets reduced.

Table 2. Ownership of Server Software Sources

Software	Nomenclature	Ownership
Operating System	MS Windows Server	Microsoft Inc
Web Server	IIS Server	Microsoft Inc
Database Management System	MS SQL Server 2000	Microsoft Inc
Application Scripts	NREGASoft	NIC

For both the off-line mode and the online mode of operation, the server configuration is listed in Table 2. The secureness of the scripts that make up NREGASoft is dependent on access to its sources. NIC owns and maintains the scripts of NREGASoft. Since these scripts are made available to local self-government institutions, secureness of NREGASoft will be dependent on the extent of access to the scripts that is made available to the users. However, when a software application is embedded inside an insecure software environment, the software project will become insecure for its users. In a study carried out by Jones, it was pointed out that at the user end, it is almost impossible to build a meaningful software metrics even for identifying its inadequacies or highlighting its worthiness as good, bad or missing [9]. The study even went ahead and claimed a metric as hazardous which was unrelated to real economic productivity. Therefore for any software project to be completely secure to its users, it should be operated only in an environment that can extend secureness of any software that is in execution. From the database description used in the application, it is evident that information related to public is encoded in proprietary data format and is opaque to its users. Deprived of the neutrality that is required in data standards or open data formats and transparency in implementation of its encoding, the secureness of data diminishes.

3.2 Role of Human Agency in Determining Software Secureness

In NREGASoft, the community of users is mostly those from the rural local bodies in India, belonging to different states and union territories in India. The developers and maintainers of the application of NREGASoft happen to be from National Informatics Center (NIC), which is a public agency under the administrative control of Government of India. The developers of the software environment of NREGASoft happen to be from a private software firm. In this proprietary software project it can be seen that the communities of users, developers and maintainers are not the same. NIC has some definite control over the sources (server scripts) it develops and maintains. The community of users, which happens to be citizens in local self government institutions, does not enjoy the same privileges for access to the sources as that of the maintainers.

A proprietary developer of kernel and similar services related to Operating System may have complete control over the entire project. This is because user-level software applications get embedded inside a proprietary operating environment, where such an operating environment can oversee any aspect of its functioning. A recent study suggested that exposure to software sources would help in reducing the number of faults which can be taken as an important factor while creating a process metrics [12], but the dilemma of software secureness would continue, so long as sources are not made available to user community.

Secureness of software is directly related to access and control over source code of software by the users. The software project may be secure enough to Microsoft Inc., who has access to all the code it develops. NIC's sense of secureness, however, is limited to its control over the sources NIC has developed. Still lesser sense of secureness will prevail on the programmers and other users in rural local self-government institutions, who may have access to some portions of the program developed by NIC. For the common rural citizens in whose service the application is created, however, the application can never be declared secure. This is because there are no legal provisions that facilitate rural citizens to inspect, test or debug the code or entrust such inspection to third-parties as brought out in the draft bill introduced in Peru [23]. In a democracy, where state serves its people, excluding people from accessing software sources is akin to excluding masters of the state. The secureness of software vis-a-vis ordinary citizens, whose information is getting processed, is therefore not prominent in NREGASoft.

3.3 Secureness through adherence to Data Standards

Software scenario is replete with instances of multiple choices for data formats available for the purposes of storage or processing in certain application domains. Wherever data formats have been declared as data standards or open data formats, it can be presumed that issues over ownership over such data standards too have been settled. This is primarily because data standards or open data formats are devoid of owners claiming exclusive rights over such formats.

Data standards or open data formats play a vital role in ensuring interoperability of encoded data between systems as they become neutral to applications that use them. Retention of ownership or rights over some or all parts of standards would dent this neutrality, in the process rendering it a non-standard. Its status then would be as a proprietary data format.

The scope of discussion on proprietary formats in which data are encoded and other related protocols used in NREGASoft is limited, as their implementation details are not available for inspection by any user, other than the firm that developed it. Additionally, there cannot be a fool-proof mechanism for validating any claims of adherence to protocols, as these are available only in binaries, mostly in a non-decodable format whose ownership entirely lies with a single agency. The licensing conditions, under which these utilities are made available to users, strictly prohibit any attempts to reverse-engineer or decode. Thus, the existing state of the art is severely limited in its scope for evaluation or scrutiny, from a technological perspective. The data encoded cannot be guaranteed to be available permanently [23]. Secureness of the system, therefore, is further compromised through the usage of proprietary formats and non-verifiable protocols.

Operations from client-side have been categorized into two modes. In the off-line mode, a local database is created and updated, from where data is updated with the central database server. Most of the software utilities are available only in binary formats. The state of client in off-line mode is almost the same as that of server. Secureness of client, therefore, is poor as in the case with secureness of server. In the online mode, it will be a web-application which will be used to update the remote database. Here too, the encoding of data for storage in remote database will be carried out in proprietary formats.

The tendency of software secureness to vary can be gauged from the interest shown by the owner of proprietary format to have it converted into a legitimate standard, relinquishing any kind of ownership. Absence of ownership over any part of format, if published as a standard, and made available for public use, would naturally mean that everyone has an equal share of ownership, enforcing neutrality. In the event of non-neutrality of encoding process, the format may need alteration to become a standard. In the case of NREGASoft, Microsoft Inc currently holds the ownership of proprietary data formats used in its systems. Hence, software secureness is severely restricted with regard to encoding of information.

3.4 A framework that indicates secureness of software

In a similar description, one can find a range associated with accessibility of software. At one end of the spectrum is making available software source codes with freedom to inspect, modify and publish, to the community that uses them. At the other end of the spectrum lie software, extended as binaries with two different variants. One variant is a type of software binaries with access to their respective sources with varying degrees of freedom over their usage to inspect, modify, alter, distribute or publish as in the case with Free Software or other Open Source projects. The other is extension of mere binaries of software with no access to their sources, denying the user community to build the binaries from their respective sources. Thus, inspection, modification, alteration etc are impossible.

A framework that adequately represents this model of arrangement is produced in Figure 1. The first and fourth quadrants deal with sources of software code, with a difference. While sources and standards are available in public domain in the case of first quadrant, the sources and data formats used in the fourth quadrant are available only within the proprietary establishments that develop them. The secureness in the first quadrant is highest, which are enjoyed by users, developers, maintainers and testers. The secureness for software in fourth quadrant is however enjoyed by only developers and testers of proprietary software. Since users of proprietary software deal only with software binaries and proprietary formats, secureness of software is absent for users of proprietary software.

Cases that deal with standards and binaries (with no access to source code) as well as cases that deal with proprietary formats and binaries (with access to source code) are both deviations from the usual norm, and hence their representation is not considered important in this framework. They are not testable too, by virtue of having binaries or proprietary formats, often legally protected from any detailed scrutiny. NREGASoft as a product independent of its environment lie in third quadrant, and the software environment that facilitates its functioning lie in the fourth quadrant. It is pertinent to note that users of NREGASoft have their software secureness seriously compromised. This analysis sets off an inquiry whether it is possible to elevate the secureness of software, and if so, what should be conditions that favour this transition.

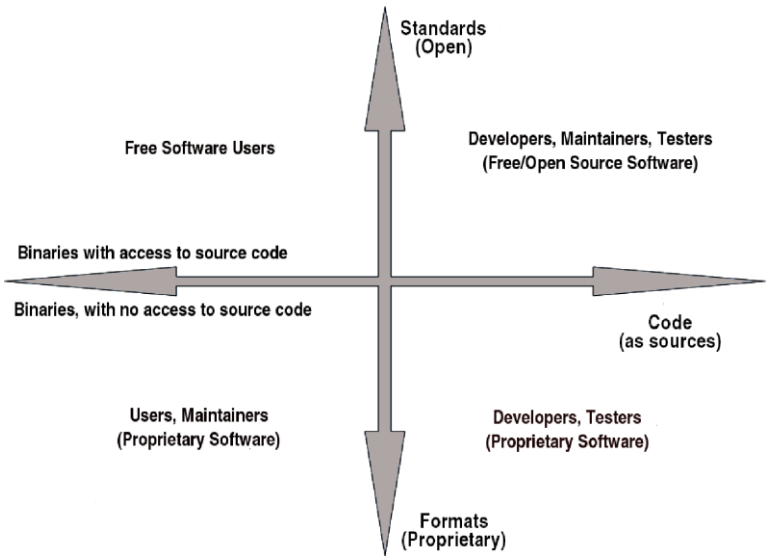

Fig 1. A framework highlighting the differing environments for Users, Developers and Maintainers

3.5 Software Monitoring Application with Enhanced Secureness

A scalable prototype for a local database management system that captures and stores information pertaining to MGNREGS was developed using Free Software applications during late 2009 and early 2010.

Table 3. Alternate Software Specifications

Software	Nomenclature
Operating System	GNU/Linux (Ubuntu 9.10)
Web Server	Apache 1.3.42
Database Management System	MySQL 5.1.31
Webserver Script Engine	PHP 5.2.12
Content Management Software	Drupal 6.15

The following software components described in Table 3 were deployed. A scalable prototype was developed with Drupal and the essential functions of a work-activity were captured and made available as reports. Assessment of work requirements and its processing was carried out at Panskura-I, a block panchayat in East Medinipore district, West Bengal. Information generated through the reports, validated the functional aspects of the prototype at the developer's level. In a rural application developed with Free Software, the transparency of the solution would be highest if the rural citizens are allowed to inspect the code that processes their information. This meant that making available packages in Free Software over an Operating System built over Free Software is inadequate.

The entire sources of the database that created the application too need to be made transparent. The new conditions made the publishing of the Structured Query Language (SQL) database dump for the application under a GNU General Public License (GPLv3), imperative. A mere replication of a database, too, is inadequate if inspections are to be carried out. The metadata design pertaining to the database that processed all work activities of MGNREGS were made part of the original design. This meant that the application displayed, as part of its features, the database design too, with relations, entity relationship diagrams and detailed description of every attribute in all relations, and the forms that invoke these relations. Moreover, all future transactions that are committed on this database would also retain the same openness and transparency, when copied for distribution. An SQL dump would then cause not only the data captured through the application available for inspection, but also the semantics of its usage. Since access privileges are controlled by the MySQL database which is separated from the database meant for storing information related to MGNREGS, unauthorized intrusions are blocked. Releasing the SQL dump under a GNU General Public License would ensure that every amendment incorporated would need to be published if the solution is made available to another party. These measures would in no way affect the operational capabilities of the monitoring software and would enhance the relationship between software design quality, development e ort and governance in open source projects as carried out in a study [2]. Rather, it would reinforce the requirements for transparency in all its operational details, which had been a condition for setting up an information processing and monitoring system [6]. The new way for replication was, thus, to install all the packages mentioned in Table 3 above, superimpose the SQL dump of backed up Drupal application database and install MGNREGS database in MySQL. The entire application would be recreated that would not only have the application, but also one that contains the design aspects too of the database. By implementing such a design, secureness of software was enhanced with ease of reproduction for the purpose of analysis or modification. The authentication codes were the only elements that were not part of the transparent package, for obvious reasons.

For developers, updating of software tools as and when new releases are distributed is essential, as most Free Software projects evolve continuously. A new challenge, therefore, would be to make available newer releases of software to all the nodes. A version control system would ensure seamless integration of the application to any versions of the software environment. Software projects may involve user, developer, tester and maintainer communities. Here, one can find that the privileges of all the communities are almost the same with regard to access to software sources, which is crucial to ascertain adherence to established protocols and adherence to open data formats or data standards.

The privileges of user communities, here, are better than those in NREGASoft. For the user community, secureness of software has been enhanced in Free Software application when compared to secureness of NREGASoft. To enhance the software secureness of NREGASoft, therefore, the conditions require that the solution be re-designed in a Free Software environment. Additionally, the proprietary formats in which encoding of public information is currently being carried out are to be abandoned in favour of open data formats or data standards, devoid of any ownership. To ensure that the application scripts too can never be closed for inspection, they too should be released under an open public license that prevents its closure in future. By having the software secureness of NREGASoft enhanced considerably to the user community, it can be safely presumed that software quality too would be improved as depicted in Fig 2.

The authors would like to point out that while this software development work merely validates the claim that secureness of software with respect to the user community can be enhanced, the study does not claim that such development work is beyond the capabilities of private software development companies. On the contrary, the authors may even recommend entrusting such development work to leading software developers in the private sector in India to make use of their vast experience and access to human resources. This study, however, accords priority to the licenses under which the transfer of rights of software and sources ought to take place that would reveal the extent of secureness of software to its users.

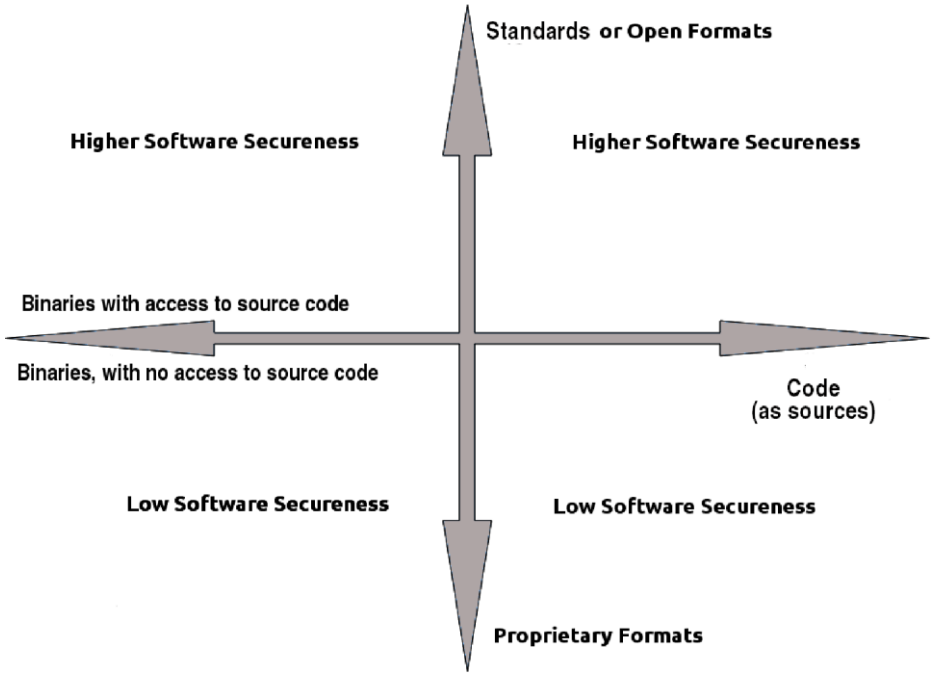

Fig 2. Varying software secureness in different software environments.

4. CONCLUSION

Software secureness varies considerably with the neutrality of the software environment under which the software gets operationalised. The concept of software secureness gets an added dimension when it is perceived from the role of a user community which is more empowered than the developer community. As a principle, software secureness is associated with adherence to use of data standards, fair implementation of protocols and transparency in coding of their implementation. Software that adheres to these criteria extends secureness to the users, developers and maintainers of software. In many software projects, especially those that process information related to public citizens, the communities of developers, maintainers and users could be different. There exist possibilities wherein software which may appear secure to developer community could become insecure to user community.

Different works in software which are released only as binaries cannot be verified for their adherence to data standards, protocols or for rules associated with those software. It is therefore vital to ensure that any software that are to be assured for its quality, adheres to established data standards or published open formats (after relinquishing ownership, so that these could be taken up for converting into a standard). Additionally, releasing the software sources would ensure that implementation details of software are transparent and do not violate any existing protocols. Rigorous methods of control have been suggested in a software quality management system adopted from standards Australia [8], which insisted on review of code and documents to assure their compliance with design criteria.

Additionally, in a constitutional setup under which such public software services are developed, operated and maintained, the user community is the one which is constitutionally the most empowered. Therefore in cases like these, software secureness should be evaluated from the viewpoint of users to ascertain software quality. NREGA-Soft, a software implementation for monitoring information processed in the employment guarantee scheme (MGNREGS) is found wanting in areas related to data standards and transparency in implementation as the current environment and software platforms are proprietary in nature.

In order to enable the government in extending the necessary guarantees over processing of information related to public, adherence to published protocols and its encoding, NREGASoft should be re-designed to be implemented with Free Software using published data standards. This variation in design and implementation would eventually enhance the software secureness to the user community of the software, thereby accomplishing better software quality. The experiment carried out with Free Software as a case study by the author, further exemplifies that by resolving to release the database dump under a GNU General Public License (GPLv3), the legal mechanisms would help in retaining the transparency of implementation in future too.

5. ACKNOWLEDGMENTS

Our sincere thanks to the block development officer and his staff at Panskura Block Panchayat in West Bengal. We are also thankful to Dr Nikhil Dey, Ms Aruna Roy and Dr Jean Dreze, the principal architects behind MGNREGS for offering valuable insights.

6. REFERENCES

[1] Boehm, B., Brown, J., Lipow, M.: 1976. Quantitative evaluation of software quality. Proceedings of the 2nd International Conference on Software Engineering, IEEE Computer Society pp. 592 605

[2] Capra, E., Francalanci, C., Merlo, F.: 2008. An empirical study on the relationship between software de- sign quality, development e ort and governance in open source projects. IEEE Transactions on Software Engineering 34(6), pp. 765 782

[3] Cowan, C.: 2003. Software security for open-source systems. Security Privacy, IEEE 1(1), pp. 38 45

[4] Dromey, R.G.: 1995. A model for software product quality. IEEE Transactions on Software Engineering 21, pp. 146 162

[5] Government of India: 2005. The National Rural Employment Guarantee Act NREGA 2005. Government Gazette (India)

[6] Government of India: 2008. Government Notification on Transparency in NREGA. Government Gazette (India) p. 9

[7] IEEE Guide for Software Quality Assurance Planning. 1986. ANSI/IEEE Std 983-1986 pp. 1 31

[8] IEEE standard for Software Quality Assurance Plans. 1989. IEEE Std 730.1-1989 pp. 0 1

[9] Jones, C.: 1994. Software Metrics: Good, Bad and Missing. Computer, IEEE 27(9), 98 100, ISSN:0018-9162

[10] Joshua Gay, editor. 2002. Free Software, Free Society: Selected Essays of Richard M. Stallman. pub-GNU-PRESS,

[11] Julia Allen, Sean Barnum, Robert Ellison, Gary McGraw, and Nancy Mead.: 2008. Software engineering: a guide for project managers Software security. Addison-Wesley Professional, First edition

[12] Khoshgoftaar, T.M., Liu, Y., Seliya, N.: 2004. A multiobjective module-order model for software quality enhancement. IEEE Transactions on Evolutionary Computation 8(6), pp. 593 608

[13] Li, X., Tiwari, M., Hardekopf, B., Sherwood, T., Chong, F.T.: 2010. Secure information flow analysis for hardware design: using the right abstraction for the job. In NY, USA, ACM.

[14] Madhuri, S., Mishra, D.: 2008. Strengthening National Rural Employment Guarantee Scheme (NREGS) through E-Governance. In: E-Governance in Practice

[15] McGraw, G.: 2004. Software Security. Security Privacy, IEEE 2(2), pp. 80 83 ISSN 1540-7993

[16] NIC, Government of India: 2007. User manual of NREGA. MIS for National Rural Employment Guarantee Act (NREGA) 2005

[17] Planning Commission, Government of India: 2003. India's Five Year Plans : Complete Documents: First Five Year Plan (1951-56) to Tenth Five Year Plan 2002-2007. Academic Foundation

[18] Rajalekshmi and K. Gopakumar.: 2007. E-governance services through telecenters: The role of human intermediary and issues of trust. Inf. Technol. Int. Dev. Proceedings of the 5th ACM SIGPLAN Workshop on Programming Languages and Analysis for Security, PLAS '10, pages 8:1 8:7, New York, , 4(1): pp. 19 35

[19] Raju, C. K., and Bhadoria, P. B. S.: 2011. Software secureness for users: Significance in public ICT applications. In Ajith Abraham, Jaime Lloret Mauri, John F. Buford, Junichi Suzuki, and Sabu M. Thampi, editors, Advances in Computing and Communications , volume 193 of Communications in Computer and Information Science , pages 211 222. Springer Berlin Heidelberg.

[20] Raymond, E.S.: 2001. The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. O'Reilly

[21] Software Quality Management System. Part 1: Requirements. Adopted from Standards Australia. 1993. IEEE Std 1298-1992; AS 3563.1-1991 pp. 0 1

[22] Tiemann, M.: An objective definition of open standards. Computer Standards and Interfaces, Science Direct 28(5), pp. 495 507 (2006), ISSN 0920-5489

[23] Villaneuva, E.: 2001. Use of Free Software in Public Agencies. Bill No 1609, Republic of Peru