ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

Assurance Driven Software Design using Assurance Case Based Approach

Dipak Gade[1], Dr. Santosh Deshpande[2]

Research Scholar, Department of CSE, Shri Jagdishprasad Tibrewala University, Chudela, India[1]

HOD, Department of CA, MES IMCC, Savitribai Phule Pune University, Pune, India[2]

ABSTRACT: Software design for dependable and critical systems is very complex. There are lot of regulations and guidelines for developing of software for such systems. These guidelines mostly demands practical evidence or documentary proof to have a justified confidence that the system shall meet its all critical requirements. To achieve this primary aim, assurance cases can be very helpful. A well-structured Assurance Case facilitates developers to state goals and sub-goals of the system and to determine the required artifacts which can be used as an evidence to prove that system is operating as per requirements. With this, one can also check if a complete set of evidences satisfies the stated requirements. Goal Structuring Notations (GSN) or Claims-Arguments-Evidence (CAE) allows graphical presentation of an Assurance Case. Graphics notations facilitate easy to present and understand assurance cases. The present paper has provided an overview of Assurance Case along with its structure and illustration taking example of Assurance Case for Door Access Control Software for better understanding. The paper has also briefly covered the details of the relevant projects carried out globally, using Assurance Cases.

KEYWORDS: Assurance Case, Software Assurance, Software Reliability, Safety, Evidence

I. INTRODUCTION

Software designing process, offering reliable and fit for use software under given constraints is challenging. In conventional approach the software is produced first and then efforts are made to verify if the designed software is trustworthy and fit for use. Under such condition, many times it is observed that the developed software is not as robust enough as required, this is mostly true specifically while development of critical systems software. To improve the reliability of such software, it may require heavy modifications in developed code which can be very cumbersome and lengthy process. The efforts may go futile since the architecture of such software may not support or allow rigorous changes at later stage. Though this may not be the case always, discovering errors lately or heavy modifications in software at final stage during development life cycle can turn out very costly. Hence it will be logical to ensure embedding of design assurance in the software during design phase of the software itself.

Software Design using Assurance Case based approach is one of the ways in ensuring Software Assurance. Software Assurance is defined as "the level of confidence that software is free from vulnerabilities, either intentionally designed into the software or accidentally inserted at anytime during its lifecycle and that the software functions in the intended manner" [5]. Software assurance is an important part of the software development process to ensure reducing risks and producing of dependable and trustworthy software. Assurance Driven Software design using Assurance Case approach is the suggestive design process for enabling software assurance. Assurance Case based software development approach facilitates combining software development and software assurance hand in hand. This ensures detecting and avoiding of potential assurance risks at earlier stage as they are realized, instead of detecting them after development is finish when they are much difficult to address. This can provide justified confidence that system shall work as expected and fit for use under stated operating conditions.

II. STRUCTURE OF AN ASSURANCE CASE

An Assurance Case can be defined as "A reasoned and compelling argument, supported by a body of evidence, that a system, service or organisation will operate as intended for a defined application in a defined environment" [11]. Assurance cases can be used to showcase some critical properties of system such as reliability, security, safety in a given operating environment.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

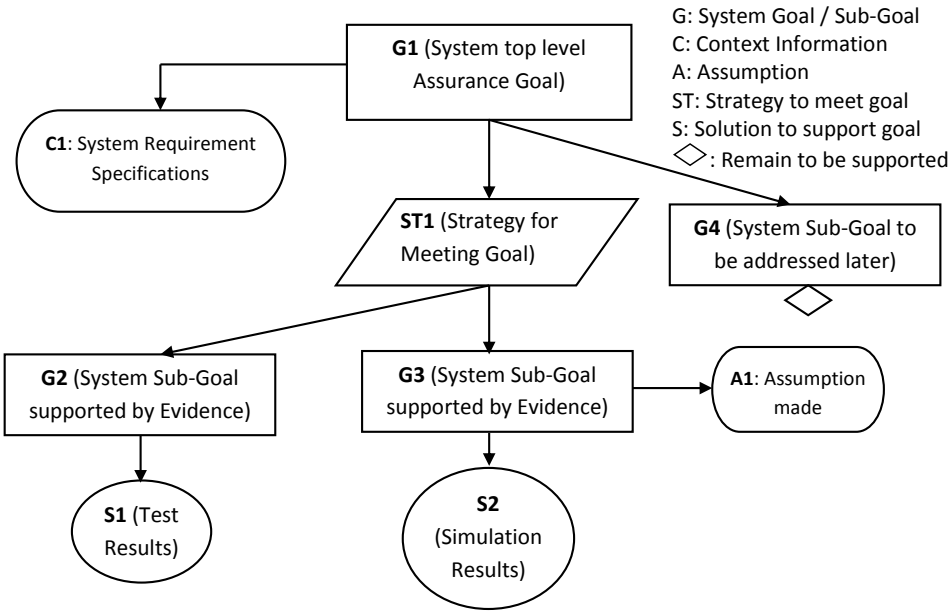

Fig. 1 Assurance Case with its basic elements

An Assurance Case presents an argument that a system is acceptably safe, secure, reliable, etc. in a given context. Where, a system could be physical or a combination of hardware and software. Based on the system goals identified in an Assurance Case, Assurance Case can also be referred as security case, dependability case, and safety case or by other relevant name as per goals applicability.

For better clarity, uses, critical engineering decisions and to ensure consistency, it is required to meet some minimum requirements for the contents and structure of an Assurance Case. These minimum requirements are specified by an International Standard ISO/IEC 15026-2:2011. To present an Assurance Case in a way to make it easy for visualization, understanding and reviewing purpose, following Graphical notation tools are used

- Goal Structuring Notation (GSN) and
- Claims-Arguments-Evidence (CAE)

CAE defines nodes for Claims, Arguments and Evidence whereas GSN uses goal oriented presentation style and defines nodes for Goals (claims), Strategy (arguments) and Solutions (evidence). Both these graphics notations are mostly similar, with some difference of progression approach. GSN follows Top –Down approach while creating the Assurance Case starting with top level goal of the system where as CAE supports Bottom-UP view starting with evidence to determine the possible claim, while preparing Assurance Case [10]. There is no thumb rule as such to decide which approach should be followed, it can be decided by developers based on their choice and information available in hand before proceeding ahead with creating of Assurance Case. Arguments presented using GSN can help provide assurance of critical properties of systems, services or organizations (such as safety or security properties). Such arguments can form a key part of an overall assurance Case [11]. Refer figure 1, which is showing the typical structure of an Assurance Case represented with Goal Structuring Notations.

Assurance Case in its simple form basically consists of following main components.

- **Claim or Goal**: This is generally some functionality, characteristics, requirement or behavior of the system that needs to be fulfilled. This can include all the essential requirements, functionalities and behavior of the system which is supposed to be met to ensure that system is fit for use. All the goals/claims are required to be supported by valid arguments based on valid evidences. The higher level goal/claim can be further

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

decomposed into sub-goals or sub-claims. To meet the higher level goal, it is required to meet each individual sub-goal or sub-claim. It is possible to have further decomposition of each sub-goal/ sub-claim to the lowest logical level. Good design needs to identify all the goals of the system which need to be fulfilled. The goals can be expressed in positive or negative statements. The Positive claims can express requirements based quality properties of the system e.g. system availability, system reliability etc. Whereas the negative claims can express specific vulnerabilities and weaknesses in the design and implementation that may lead to system failure or compromise the system.

- **Evidence or Solution:** Evidence is valid artifact to support the respective goal or claim. Valid evidence is traceable to its source, repeatable, reproducible and provides concrete support in satisfying the goal or claim. Evidence is an essential element in creating credible Assurance Case. Without valid Evidence, there is no way to prove that the goal or claim is satisfied. Evidence may be produced and or demonstrated either automatically or manually depending upon the requirements and system context. Valid Evidence can include results of analysis, simulation, modeling, test results, working prototype demonstration, inspection reports and similar dependable and reliable artifacts which can provide deterministic, qualitative and quantitative data or information. Though logically it is desirable to have high quality and ambiguity less evidence to ensure meeting system goals without any doubt, often in practical scenario it is not the case so. It is very difficult to determine up to what level the clarity and data/information coverage an evidence should have. Under such cases, combinations of expert opinion and available test results/reports are collectively taken into account to have justified evidence supporting the respective goal or claim.

- **Argument or Strategy**: Arguments are basically logical links between evidence and goal. Argument links the evidence with goal in such a way that it can justify the goal without any doubt. This is done by defining the relationships directly linking each goal or claim, sub-goals or sub-claims with the respective evidence used for supporting the goal or claims. Arguments can also include any unusual events or conditions that are within the context of the claim. Arguments can cover potential causes of failures and the necessary corrective actions if failure occurs. Thus, an argument may include system assumptions, conditions, judgments about the system, its use and operational environment, threats, and likelihood of occurrence, for which the respective evidence and claims are made as part of an overall Assurance Case for the specified system [10]. For a successful achievement of Assurance, clarity of Assurance Argument is important in convincing the stakeholders. Many times, arguments may get lost in sections of detailed text, lacking in mapping to the supporting evidence. Developers hence need to be careful, and should provide compelling and sound arguments.

Apart from these basic elements, an Assurance Case can also have additional elements to structure and represent it properly. These additional elements can bring more clarity to Assurance Case and also helps in better understanding and reviewing the Assurance Cases. These additional elements are as follows.

- **Context:** This is basically additional supporting information which can define basis for claims. It can include information related to system requirements, system operating environments etc. which can be relevant to specified goal or claim to bring clarity in the Assurance Case. Without system context information, it can be difficult to understand the stated claims.

- **Assumptions:** As name suggests, this includes assumptions made with respect to claims defined or strategies decided for meeting the respective goals or claims.

Additionally, while preparing an Assurance Case, one can come across certain goals or claims which needs to be supported and may be required to address at later stage, such goals or claims are presented by attaching diamond symbol to it. Refer figure 1 where sub-goal G4 is a goal shown attached with hollow Diamond and stated as to be addressed later.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

III. ASSURANCE CASE CHARACTERISTICS AND FRAMEWORK

Writing an Assurance Case in first instance is not an easy job. To start with one must be very clear the ultimate requirements software system needs to essentially fulfill. This job is very challenging since for most of the critical software systems all the critical requirements may not be available in the beginning itself, rather they will be evolving as we start progressing. It is necessary that Assurance Case needs to be clear and complete. To cover all the required information, the assurance cases are applied repeatedly to produce a hierarchic kind of structure with the overall goal at the root level for the real system [13]. Evidence at one level becomes goal at the subsequent lower level which facilitates argument to be manageable at each level [6]. Table 1 has presented a typical reference framework for an Assurance Case, which can be very helpful while developing the assurance cases.

Assurance Case Terms	Goal/ Claim		Context	Strategy	Evidence/ Solutions		
Development Life Cycle	System Definition and Feasibility Study		System Requirements	Design and Implementation	Testing and Acceptance		
Questions Mapping	What is the ultimate goal of the system?	What are the sub-goals of the system?	Under what environment System is going to Operate?	How we are going to achieve System Goals?	On what basis you can declare that the system goal/ sub-goals are successfully achieved?		
Assurance Case	System is Safe to Operate	All sub-systems are acceptably safe against specified operating conditions	Identification of System and Sub-system Hazards	System and subsystems Hazards are eliminated	Test Results	Simulation	Working Prototypes Demonstration

Table 1 Assurance Case Reference Framework

Creating a proper Assurance Case is a time consuming and complex job. It is an iterative and cumulative process. To ensure that all relevant goals and sub-goals are covered in an Assurance Case, it is necessary to decompose each goal and sub-goal to next level of sub-goals till the point where it is possible to achieve the valid evidences systematically to support each arguments and sub-goals and collectively the highest level of goal or claim of the system. Such Assurance Case is then formed as a well-structured Assurance Case. On the other hand, missing information, missed goals and or sub-goals, invalid arguments can lead to poorly structured Assurance Case. Poorly structured Assurance Case will not help in producing reliable system/software design rather it will damage the entire design process and may lead producing faulty or non-reliable system/software. Table 2 has listed down the characteristics of poorly structured and well-structured Assurance Case.

Sr. No.	**Poorly Structured Assurance Case**	**Well Structured Assurance Case**
1	Missing Essential Information	All Goals and Sub-Goals are stated correctly
2	Covering of Irrelevant Information	System Context included wherever required
3	Unclear logic	Logical Information Flow
4	Use of Jargon Terminology	Proper References and Terminology
5	Providing Weak Evidences	Relevant and Complete Evidences
6	Improper Arguments	Compelling, Valid and Sound Arguments
7	Too brief or too many details	Covers reasonable and sufficient details

Table 2 Assurance Case Characteristics

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

IV. ASSURANCE CASE ILLUSTRATION

An example of Assurance Case for a typical Door Access Control using thumb impression is taken here for illustration of Assurance Case creation and presentation. Refer Figure 2, where Door Access Control Software Assurance Case is presented using GSN.

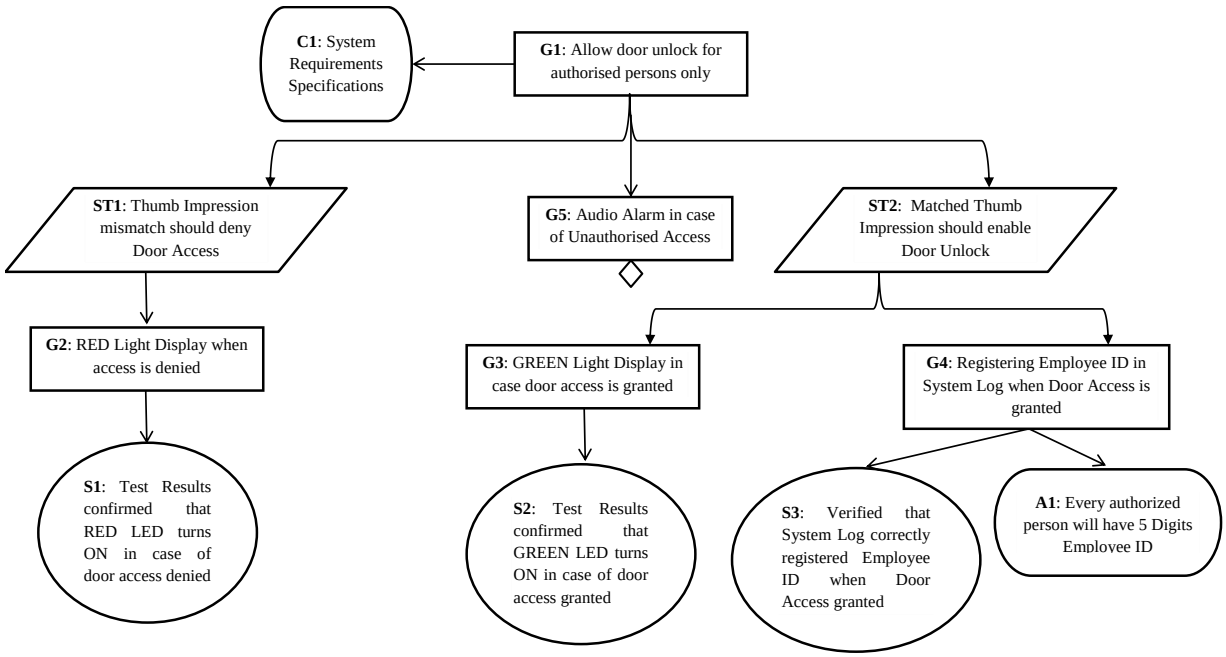

Fig. 2 Assurance Case for Door Access Control Software

The highest level assurance goal of the door access control software is to provide access by unlocking the door for an authorised person only whose thumb impression correctly matches with any of the approved thumb impressions available in its database. In other words, any unauthorised access should be blocked by keeping the door locked. This goal has been derived from the system requirements specifications.

As shown in figure 2, Goal G1 is the top level assurance goal that the Door Access Control Software needs to fulfill. As per this goal the software need to permit the door unlocking only for authorized persons. To meet this goal the strategy adopted is represented by ST1 and ST2 which says that authorized people who will be granted door access are those whose thumb impression matches with any of the approved thumb impressions available in the system database. Matching of thumb impression should unlock the door else door access should be denied. The top level goal G1 is further decomposed in to sub-goals. The sub-goals have identified additional requirements that door access control software need to fulfill. G2 has claimed that in case of denied door access Red light should be displayed to indicate door access is denied whereas in case door access is granted, goal G3 claims that Green light should be displayed. To support sub-goals G2 and G3, solutions S1 and S2 are provided through direct test results. S1 verifies the goal G2 by testing the door access denied scenario and by confirming that the Red LED turns ON in this case. Also S2 verifies through testing that in case of door access granted Green LED turns ON. Note that there is sub-goal G4 which has put forward an additional requirement to be met. Goal G4 demands logging of Employee ID when the person is granted door access. Since it is not clear if every authorized person will have employee Id or not, the Assurance Case has stated an Assumption A1 for Goal G4 that each authorized person will have a 5 digits Employee Id. Goal G4 is associated with solution S3. Solution S3 meets Goal G4 by verifying system log and by confirming that Employee Id of an authorized person gets registered correctly when the person gets door access. There is one additional requirement specified by Goal G5 which demands audio alarm for an unauthorized door access. The Assurance Case has shown that

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

Goal G5 is attached with Diamond symbol which means that this goal is not yet supported and will be addressed at later stage.

It should be noted that figure 2 has presented only part of structured Assurance Case for Door Access Control Software. It has covered almost all the essential elements of Assurance Case represented through GSN, however this Assurance Case is not complete. There are many other goals which need to be covered by Door Access Control Software Assurance Case. This Assurance Case has been explained and presented as an example for better concepts understanding. The Assurance Case for a production kind of Door Access Control Software system will be much comprehensive and will cover all the required goals, sub-goals, along with strategies and required solutions to meet the specified goals. It will also cover additional relevant information such as system context, assumptions made, notes etc. to provide more clarity.

V. RELATED WORK

Assurance Case based software design can be considered as comparatively new design methodology for software designing against the popular traditional software development methodologies still followed by larger community of software developers. Some Software organisations including research institutions and software developers working on software development specifically for critical systems have carried out some significant work of software development using Assurance Case approach. This section has provided overview of some of the relevant work carried out globally using Assurance Cases for software development.

In [1] Peter, Robin and Sofia have described in detail the goal based Assurance Case and its structure. The authors have also stated the possible issues associated with safety case approach. Authors have highlighted how to structure safety case by separating claims about the system from claims about safety case. This differentiation can assist in distinguishing system quality and quality of arguments and evidence which support the safety case. In [3] Luke and Sofia have described the software Tool ASCE (The Assurance and Safety Case Environment) which is a graphical hypertext software system that can be used to develop, review and maintain assurance and safety cases and relevant technical documentation in structured fashion. Authors have also claimed that the ASCE software tool can be configured to support in software certification process. In [6] Patrick, John and Elisabeth in their paper have explained the Assurance Based Development (ABD) model for development of critical computing system. This approach has used Assurance Case as backbone for constructing and verifying the system goals and evidences and thus providing of justified confidence that the system will work as per the requirements without any issues in given circumstances. Authors also provided details on how they have used ABD approach to develop part of a research prototype for a software system meant for alerting pilots to runway incursions at airports. Authors pointed out that ABD approach facilitates them having various system development choices to choose from and evaluate to ensure having an assurance that the system shall meet its dependability goals. In [7] Nguyen, Greenwell and Hecht discussed a specific industrial application of Assurance Case for transitioning from a legacy global positioning system to its replacement a new AEP system which is basically a ground control system. Through the Assurance Case approach, authors ensured that the system transition should not pose any compromise to its mission assurance objectives. This was accomplished through an Assurance Case by restructuring the procedure based documentation to easy manageable analysis kind of documentation. It was confirmed by authors that there were no major hazards faced during system transitioning and the results were validated by a successful transition. In [9] Eunkyoung, Insup and Oleg discussed how they have developed Pacemaker software using structured Assurance Case. The authors constructed an assurance case for the model driven development of cardiac Pacemaker software and demonstrated that the developed code is safe to operate. It is also pointed out by authors that Assurance Case needs to be properly structured to get real advantage out of it; else a poorly structured Assurance Case can adversely affect the entire software development process rather than helping it.

VI. CONCLUSION AND FUTURE WORK

Critical Software Systems requires adequate risk reduction, high reliability and safety, to ensure its safe and intended operation under given operating conditions. This demands dependable design approach which can provide guarantee that designed software is absolutely fit for use. In present paper we have discussed the Assurance Driven Software Design using Assurance Case based Approach. The Assurance Case structure is discussed in detail and it is also highlighted that designed software using well-structured Assurance Case can be more reliable and can produce trustworthy and fit for use software. To get full advantage of Assurance Case based design, it is necessary to properly

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

structure the Assurance Case and need to cover all essential Goals, valid Solutions and Strategies for meeting the specified goals. It is also recommended that Assurance Case should include sufficient details about system context, assumptions made and relevant notes wherever required. This will ensure making Assurance Case comprehensive enough and easy to review. Assurance Cases are represented by graphical notations mainly by GSN or CAE and developers may use any of these notations based on their convenience for creating Assurance cases.

In future, it is recommended to implement real life practical system software design using Assurance Case based Approach. It should also be checked if we can automate the process of creating Assurance Cases by providing software specifications.

REFERENCES

1. Bishop P., Bloomfield R., Guerra S., "The future of goal-based assurance cases", Conference proceedings for International Conference on Dependable Systems and Networks, 2004
2. Samantha Lautieri, David Cooper, David Jackson, Trevor Cockram, "Assurance Cases: how assured are you?", The proceedings of the 2004 International Conference on Dependable Systems and Networks, 2004
3. Emmet, L., Guerra, S., "Application of a Commercial Assurance Case Tool to Support Software Certification Services", Software Certificate Management Workshop, ASE Conference, 2005
4. Rob Weaver, Georgios Despotou, Tim Kelly, John Mcdermid, "Combining Software Evidence: Arguments and Assurance", Proceedings of the 2005 workshop on Realising evidence-based software engineering, 2005
5. National Information Assurance Glossary,NSS Instruction No. 4009, available at www.cnss.gov/CNSS/ and also at http://jitc.fhu.disa.mil/pki/documents/committee_on_national_security _systems_instructions_4009_june_2006.pdf, June 2006
6. Graydon P.J., Knight J.C., Strunk E.A., "Assurance Based Development of Critical Systems", Dependable Systems and Networks, 2007
7. Nguyen E.A., Greenwell W.S., Hecht M.J., "Using an Assurance Case to Support Independent Assessment of the Transition to a New GPS Ground Control System", Conference proceedings for Dependable Systems and Networks With FTCS and DCC, IEEE Publication, June 2008
8. Charles B. Weinstock, Presentation on "Assurance Cases", Software Engineering Institute Carnegie Mellon University Pittsburgh, December 2008
9. Jee E., Lee I., Sokolsky O., "Assurance Cases in Model-Driven Development of the Pacemaker Software", 4th International Symposium On Leveraging Application of Formal Methods Verification and Validation (ISoLA), Part II, pp. 343-356, 2010
10. Thomas Rhodes, Frederick Boland, Elizabeth Fong, and Michael Kass, "Software Assurance Using Structured Assurance Case Models", Journal of Research of the National Institute of Standards and Technology, Volume 115, Number 3, May-June 2010
11. GSN Community Standard Version 1, www.goalstructuringnotation.info/documents/GSN_Standard.pdf, November 2011
12. Baxter, Presentation on "Adapting Classic Assurance Case Theory to Medical Device Development: A Manufacturer's Perspective", 62nd Meeting of IFIP 10.4 Working Group on Dependable Computing and Fault Tolerance, Rockport MA, June 2012
13. Dipak Gade, Dr. Santosh Deshpande, "A Literature Review on Assurance Driven Software Design", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 4, Issue 9, September 2015