Software Risk Assessment: Fuzzy Logic Approach to Risk Estimation (FLARE)

Willie Fitzpatrick, PhD; US Army AMRDEC Software Engineering Directorate; Redstone Arsenal, AL, USA

David Skipper, PhD; SAIC; Huntsville, AL, USA

Josh McNeil; US Army AMRDEC Software Engineering Directorate; Redstone Arsenal, AL, USA

J.P. Rogers; APT Research, Inc.; Huntsville, AL, USA

Keywords: Possibility Theory, Fuzzy Logic, System Safety, Airworthiness, Software Safety, Risk Assessment, Software Risk Assessment

Abstract

Industry standard methods for hardware and operations risk assessments include hazard severity and hazard likelihood in risk predictions. Software "hazard" assessments include the same hazard severity employed in hardware and operations risk assessments, but use software control authority in lieu of failure likelihood (ref. 1) to determine a software safety assurance rather than a "risk". Since risk is estimated by the combination of event severity and likelihood, hazard assessments for software do not result in "risk". In the absence of a software risk, it is difficult to characterize the system level risk as a composite of hardware, operations, and software risk. To correct this deficiency, a qualitative estimate for the likelihood of software safety failures is needed. Attempts to determine the quantitative "failure probability" of software have not resulted in wide acceptance or consistent application. This paper presents an estimation formalism based on a fuzzy logic approach to software risk estimation. This approach utilizes current accepted software safety processes so that a fuzzy estimate for software safety failure likelihood can be combined with severity category to estimate software contribution to system level risk.

Overview

Risk is the product of hazard severity and event likelihood (ref. 2). Industry standard methods have been developed for hazard risk assessment. These methods are used for hazard risk assessments of hardware failures and operations errors. Standard hazard assessments include both the hazard severity and the event likelihood risk components. However, estimating software "risk" is not a standard activity in software hazard assessments (ref. 3). Predicting system level risk in terms of the composite of hardware, operations, and software risks is a desirable, but difficult objective, given the absence of a true software risk assessment (see Figure 1). This paper proposes a method to address the software risk assessment deficiency in the system level risk assessment. A fuzzy logic based approach is employed to develop a qualitative likelihood of software safety failures. This qualitative likelihood is then combined with hazard severity to produce a fuzzy software risk level estimate that is qualitatively similar to the hardware and the operations risks levels (*e.g.* High, Medium, Low) (ref. 4). The hazard severity risk component is assumed to be developed from analysis performed prior to initiating the FLARE process. FLARE is then employed to estimate the missing software risk component - the likelihood of a software safety failure event (see Figure 2). The FLARE process is currently being assessed using actual system and safety data.

Assessing hazard severity in linguistic terms (*e.g.* catastrophic, critical, etc.) is a straightforward activity (ref. 4). However, estimating the likelihood of a software safety failure in a Safety Significant Function (SSF) is not a trivial process (ref. 3). Software safety assessments are historically not probabilistic in nature. They are based on individual decisions analysts make. The assessment is evidence/artifact driven and it reflects the analyst's confidence or belief in the "goodness" of the software's safety characteristics (ref. 5) relating to software failures. The analyst's confidence or belief is then the basis for developing risk likelihood. The Software System Safety discipline has adopted a safety assessment process for analysts that use both software hazard analysis objectives and software development objectives that are designed to reduce the likelihood of software safety failures (refs. 6, 7). These are the analyst's primary evidence/artifacts and they are used to increase/decrease the analyst's belief that the

software has reduced/increased likelihood of failure. Prior to addressing the concepts associated with the analyst's belief, there are some common definitions that require a quick review.

Different analysis and development objectives are required depending on the SSF hazard criticality. The hazard criticality is defined by the Software Criticality Index (SwCI). SwCI Level A is the highest hazard criticality index level and results in the most rigorous development effort. SwCI Level D is the lowest hazard criticality index level and results in the least rigorous development effort. SwCI Level E classifies software as not having any impact on safety or mission success (ref. 8).

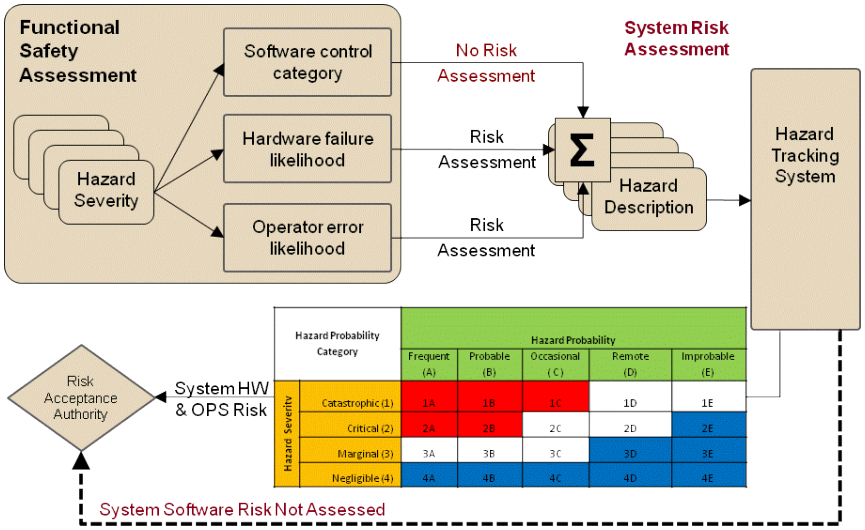

Figure 1: Current System Safety Process

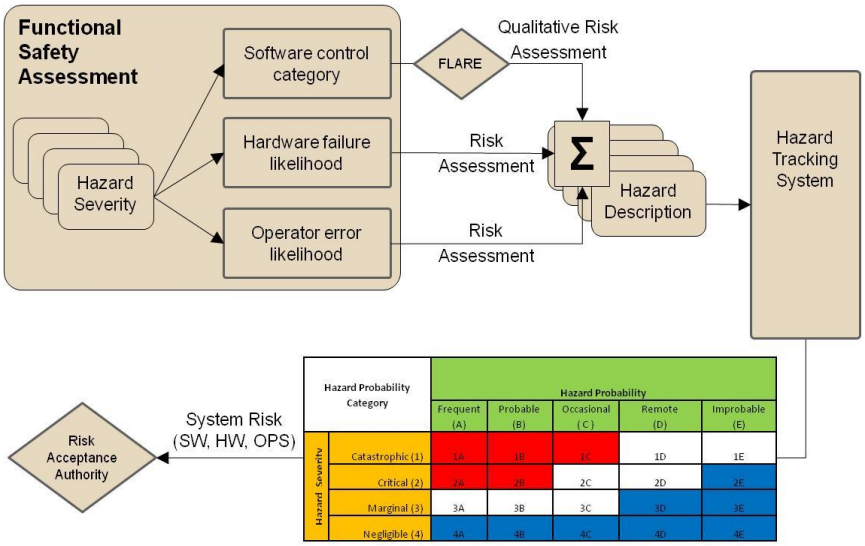

Figure 2: Proposed System Safety Process

The set of hazard analysis and software development objectives prescribed by a given SwCI linguistic category (*e.g.* A, B, C, or High, Medium, Low) presently produce necessary and sufficient evidence to prove to the safety personnel that the software safety assurance meets the SwCI safety goals for the category. The software safety analyst is responsible for assessing the veracity of the evidence submitted as proof. Several assessment guidelines are available (Refs. 1, 6, 9, 10). Uncertainties in the assessment process may be due to: (1) human factors, *e.g.* insufficient peer review, informal analyst training, lack of analyst experience, and diversity of analyst pass/fail perceptions, how evidence ambiguity is treated, and (2) inadequate data, *e.g.* poor software hazard analyses,

incomplete software development processes, poor requirements development/traceability, ambiguous presentations. These sources of uncertainty are not addressed in the FLARE process. Instead, FLARE, focuses on standardizing the aggregation of the assessment results from individual evidence items and analyses. FLARE is based on the perception that the analyst's assessment findings portray his/her "belief" that the desired software safety assurance has been achieved, which in turn is based on the analyst's "belief" in how much the software safety assurance is supported by his/her assessment of the efficacy of the evidence to provide assurance that identified hazards have been mitigated. Therefore, the analyst's "belief" in software safety assurance is assumed to be a qualitative estimate of the likelihood for a safe response to software errors. This assumption is stated in the following foundational FLARE axiom:

The analyst's "belief" in software safety assurance is a *qualitative* estimate for likelihood of software safety failure.

FLARE does not specify how the safety analyst must reach their assessment only that they can and do make such an assessment. Instead, the challenge for FLARE is to formalize a process which maps the analyst's cognition, as it relates to belief in software safety assurance, to bounded likelihood categories so that likelihood of an event can be qualitatively described in linguistic terms such as "frequent", "occasional", or "improbable". These linguistic terms are then modeled as fuzzy numbers to form the basis for FLARE as described in the next section.

FLARE Introduction

Fuzzy numbers [ref. 11] represent a possibility distribution [ref. 12] over a real number line. Possibility distributions capture what is possible versus what is probable. As such, possibility is less "constraining" than probability. However, in cases where probability is not available, possibility theory offers a framework to model the data limitations and manipulate them to develop boundaries for decisions. FLARE employs fuzzy numbers to model the analyst's beliefs. These fuzzy numbers are manipulated by fuzzy logic to arrive at bounded decisions. As always, the extent of the bounds is driven by the bounds on the data and the processing of the data. Therefore, the FLARE process does not "magically" provide "good" decisions from an imperfect data set, merely traceable possibility boundaries.

Fuzzy logic concepts and operations employed in FLARE help to characterize and manage the qualitative characteristics found in software safety assessments. The FLARE process then associates qualitative belief in software safety assurance to a Software Risk Possibility (SRP) matrix. FLARE provides a method for "assessment of confidence" by the analyst for each safety-significant requirement and function as required by MIL-STD-882E (ref. 5). Confidence in this context is not the same as the mathematical confidence interval commonly used in probability and statistics. Here, it is a qualitative measure of analyst "belief" that satisfactory compliance with specific objectives will improve the "software safety goodness", and thereby reduce the likelihood of software safety failures. For the remainder of this paper, we will use "belief" in lieu of "confidence" to avoid confusion with probability terminology.

The FLARE results represent a qualitative estimate of the software contribution to system-level risk. FLARE is based on the following assumptions: (1) As each SwCI objective is completed, with sufficient quality, software safety assurance is increased/decreased, which directly correlates to an increased belief in a safe/unsafe response to software errors; (2) Completion of all the prescribed objectives for a given SwCI, with sufficient quality, will represent all due diligence required to result in the desired software safety assurance; (3) The qualitative estimate for likelihood of software safety failures depends on the specific objectives completed, the quality of the evidence, and the objective's contribution to software safety assurance.

The FLARE process has three high level steps (see Figure 3):

(1) **Scoring objectives**: Each compliance evidence artifact, of which there may be multiples, is assessed against the SwCI objective's requirements. Three assessment criteria are used for each artifact: (a) Completeness, (b) Quality, and (c) Contribution. Completeness is an assessment of the percentage of key information provided by each of the evidence artifacts. If combining scores for individual artifacts is a requirement, this then provides an assessment of the objective's completeness based on all the evidence. Quality is an assessment of the goodness of each artifact. Combining the evidence assesses the quality of the evidence supporting the completion of the objective. The

Contribution criteria is an assessment of what extent the objective contributes to changing the likelihood of software safety failures.

(2) **Processing Scores**: The scores for each SwCI objective are numeric based inputs. These inputs are processed through a fuzzy logic transformation system to result in a range of possible values for the likelihood- Likelihood Range (P).

(3) **Estimating Risk Possibility**: The Likelihood Range is paired with the SSF's hazard severity category to estimate the Software Risk Possibility (SRP) (*e.g.* High, Medium, or Low).

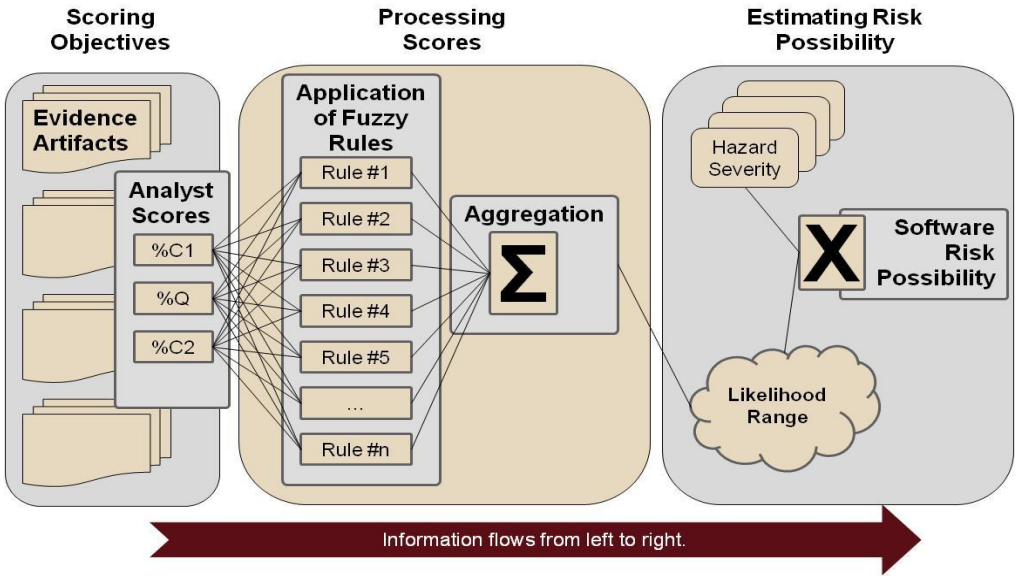

Figure 3: FLARE Process

Using FLARE

This section illustrates the FLARE method using the following information set:

Hazard Description:
Source: Failure condition that prevents continued safe flight and landing, or results in loss of aircraft.
Mechanism: Undetected incorrect flight information.
Outcome: Death or permanent total disability; system loss
Software Contribution: Yes
Severity Category: Catastrophic
Software Control Category: Autonomous
Software Hazard Criticality Index: High (1[I])
Level-Of-Rigor (LOR): High (or SwCI-A) (requires significant analysis and testing resources)

The Program Manager Handbook for Flight Software Airworthiness (Ref. 13) provides SwCI objectives that must be fulfilled. Our example data would require all 108 possible objectives be accomplished to ensure SwCI-A compliance. In this context, compliance means complete and high quality evidence/artifacts that establish levels of belief that software error leading to software safety failures have been eliminated or acceptably mitigated. The FLARE process is used to evaluate each objective independently. For brevity, only three objectives are shown in Table 1 and only one is further examined here.

Score the objectives: As discussed in the previous section the analyst scores the objectives for Completeness, Quality and Contribution. The assessment could be in linguistic terms (*e.g.* bad, okay, great) or exact values or interval based values (*e.g.* between 20 and 30%). All of these expressions of assessment can be represented as fuzzy

numbers. The FLARE process example illustrates with exact values. Three example objectives are scored in Table 1.

Table 1: Scoring Example

List of Objectives	DAL Level	Complete (%)	Quality (%)	Contribution (%)
Integrated master schedule for the system/software development is established	A	25	50	15
FHA is developed	A	64	28	84
System safety requirements are traceable to the FHA	A	30	45	95

For the remaining steps in the FLARE description, the example test case only considers a single SwCI objective, "FHA is developed". The associated scores are: Completeness = 64%, Quality = 28%, and Contribution = 84%. In order to understand the elements of fuzzy numbers and fuzzy logic used in FLARE, the following brief section discusses some background fuzzy logic concepts that are used in processing these scores.

Background Concepts:

Fuzzy sets and fuzzy numbers are used to represent possible values either as discrete items in a set or as continuous numeric values. The idea of what is possible is important to FLARE since there is some research (ref. 14) that suggests that analysts assess possibilities in problems with uncertainty. Given that FLARE uses analyst assessments, representing and manipulating possibility seems natural. Fuzzy logic provides methods for performing logical operations on these fuzzy values and a fuzzy calculus can provide methods for performing math operations on fuzzy numbers. Each fuzzy set can be identified by a linguistic variable scale to facilitate human interaction. Odd numbers of values in the scales are used to permit a middle ground to be stated. FLARE utilizes linguistic values to characterize five key variables with 5 possible values in each scale: three are input variables, one is an intermediate variable, and one is an output variable. The input variables are Completeness (X), Quality (Y), and Contribution (Z). The intermediate variable is Belief (T) and the output variable is Likelihood Range (P). Each linguistic variable can have a defined set of values such as are described below:

> Completeness = [*Mostly Incomplete, Some Information, Some Key Information, Most Key Information, All Key Information*]
>
> Quality = [*Inferior, Below Average, Average, Above Average, Superior*]
>
> Contribution = [*Very Small, Small, Moderate, Large, Very Large*]
>
> Belief = [*Very Low, Low, Medium, High, Very High*]
>
> Likelihood Range = [*Frequent, Probable, Occasional, Remote, Improbable*]

Using human analyst oriented value ranges described using words like those above or words like "Small" and "Very Small" gives a relative association without defining hard boundaries. However, in order to make these relative associations meaningful, they must be associated with numeric sub-ranges of possible values that match reasonable responses from the linguistic population, *i.e.* the analysts. Representation of these responses is accomplished through the development of a range of possible numeric values for each linguistic value. In fuzzy logic, this range of possible values is represented by the membership function. Thus the membership function relates the members of a given linguistic value on a numeric value scale.

In the case of the input variables, Completeness, Quality, & Contribution, the fuzzy value sub-ranges are taken from the full range of possible compliance scores (*i.e.* 0% to 100%). An example membership function for Completeness = "*Some Key Information*" is shown in Figure 4. Note that the "*Some Key Information*" membership function will only respond to the sub-range of analyst's estimates of Completeness scores ranging from 30% to 70%. The shape of this membership function is not a square or rectangle of abrupt change because this function represents a decreasing possibility of membership in "Some Key Information" as the values move away from 50%.

Figure 4

It must be noted that the degree-of-membership does not represent a "percentage". Rather it maps a set of values with their possibility or degree of "belonging" to a value in a linguistic set value. A degree-of-membership of zero (0) denotes complete absence of membership in a specific linguistic "value" while a degree of one (1) represents full membership in the linguistic set value.

Processing scores: Continuing with the exact value analysis, Figure 5, Figure 6, and Figure 7 show the example values from the previous section plotted on their respective membership functions to show the relationship of specific numeric values to membership functions.

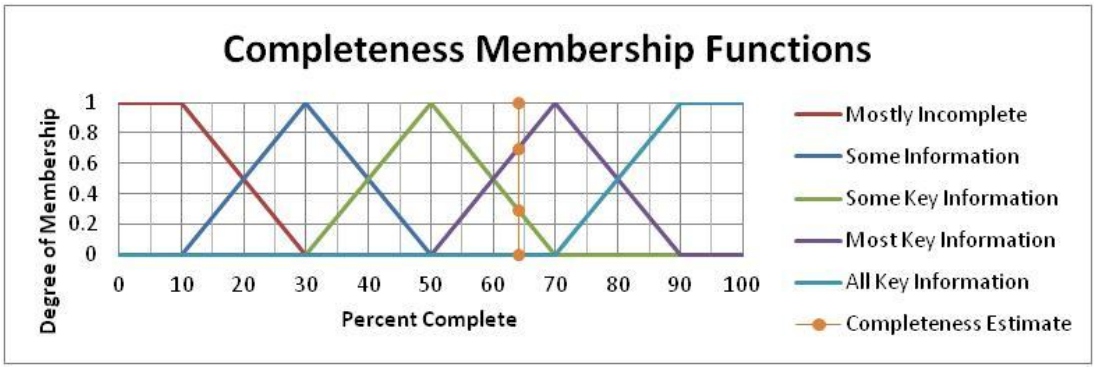

Figure 5

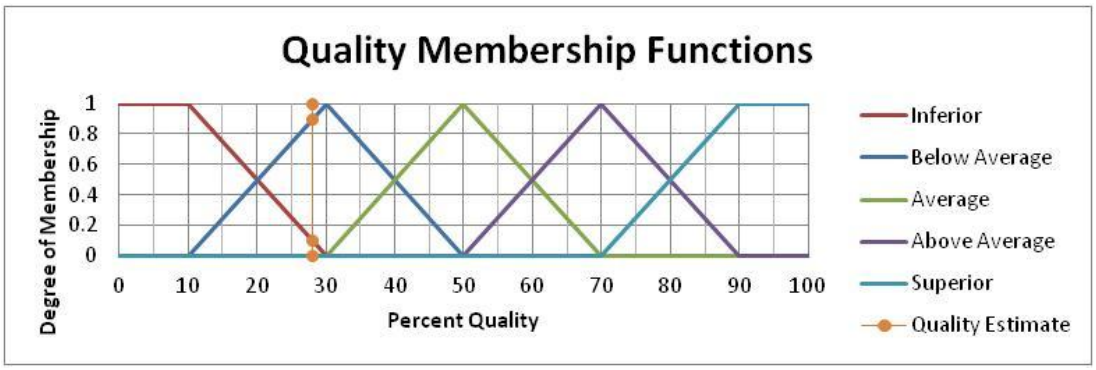

Figure 6

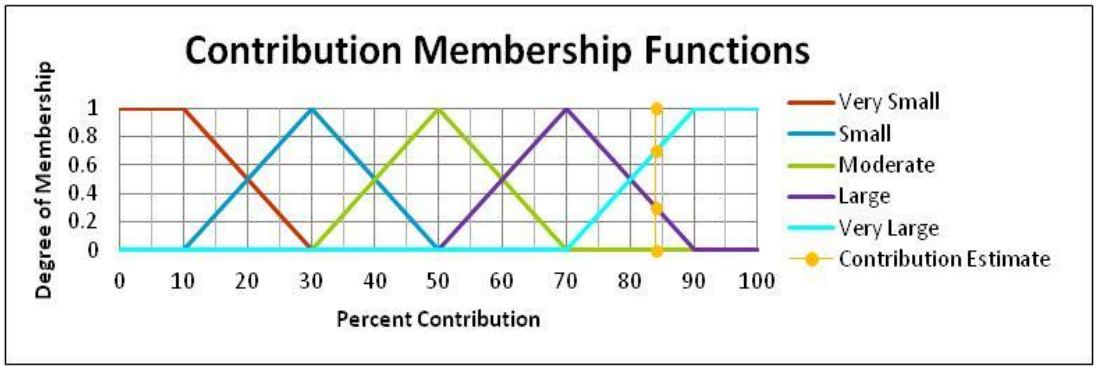

Figure 7

Table 2 summarizes the results of the degree-of-membership plots:

Table 2: Degree of Membership

Linguistic Variable	Analyst Score Value	Linguistic Set Values	Degree-of-Membership
Completeness	64%	Some Key Information	0.3
	64%	Most Key Information	0.7
Quality	28%	Inferior	0.1
	28%	Below Average	0.9
Contribution	84%	Large	0.3
	84%	Very Large	0.7

Examine the Completeness variable. Note that the Completeness variable has a degree of membership of 0.3 that completeness is described by the linguistic value "Some Key Information". It also has a degree of membership of 0.7 that Completeness is described by "Most Key Information". FLARE must account for both possible values. FLARE uses a fuzzy "rule" approach vice a fuzzy numeric approach to accomplish this. The "rules" describe relationships between values and linguistic variables.

FLARE rules use the "IF antecedent THEN consequent" rule format to relate the linguistic variables. Note that FLARE currently does not use a belief in the rule implication itself, which is distinct from the belief in the data. Here are example rules employed to relate the linguistic variables:

1. If Completeness = X and Quality = Y then Belief = T
2. If Contribution = Z and Belief = T then Likelihood_Range = P

The possible linguistic values for X, Y, Z, T, and P are defined from the respective fuzzy sets for each linguistic variable. Numeric values for X, Y, and Z are determined using the membership functions as shown in Figures 5, 6, and 7 above. Linguistic values for T and P are determined using the Belief and Likelihood Range rule matrices (see Table 3and Table 4). The Belief rule matrix maps Completeness and Quality values to Belief values.

Using the associations from the Belief rule matrix the following rules are derived for the example data. Values in parentheses are specific degree-of-membership values.

If Completeness = *Some Key Information* (0.3) **and** Quality = *Inferior* (0.1) **then** Belief = *Very Low* (0.1)

If Completeness = *Most Key Information* (0.7) **and** Quality = *Inferior* (0.1) **then** Belief = *Very Low* (0.1)

If Completeness = *Some Key Information* (0.3) **and** Quality = *Below Average* (0.9) **then** Belief = *Low* (0.3)

If Completeness = *Most Key Information* (0.7) **and** Quality = *Below Average* (0.9) **then** Belief = *Low* (0.7)

Table 3: Analyst's Belief Rule Matrix

Belief (T)		Completeness (X)				
		Mostly Incomplete	Some Information	Some Key Information	Most key Information	All Key Information
Quality (Y)	Inferior	VL	VL	VL	VL	VL
	Below Average	VL	L	L	L	L
	Average	VL	L	M	M	M
	Above Average	VL	L	M	H	H
	Superior	VL	L	M	H	VH

Very Low (VL), Low (L), Medium (M), High (H), Very High (VH)

The Likelihood Range rule matrix maps the Belief and Contribution values to the Likelihood Range values.

Table 4: Likelihood Range Rule Matrix

Likelihood Range (P)		Belief (T)				
		Very Low	Low	Medium	High	Very High
Contribution (Z)	Very Small	O	R	R	R	I
	Small	O	O	R	R	I
	Moderate	P	O	O	R	I
	Large	P	P	O	R	I
	Very Large	F	P	O	R	I

Frequent (F), Probable (P), Occasional (O), Remote (R), Improbable (I)

The Likelihood Range rules are shown below:

If Belief = *Very Low* (0.1) **and** Contribution = *Large* (0.3) **then** Likelihood Range = *Probable* (0.1)

If Belief = *Very Low* (0.1) **and** Contribution = *Very Large* (0.7) **then** Likelihood Range = *Frequent* (0.1)

If Belief = *Low* (0.3) **and** Contribution = *Large* (0.3) **then** Likelihood Range = *Probable* (0.3)

If Belief = *Low* (0.3) **and** Contribution = *Very Large* (0.7) **then** Likelihood Range = *Probable* (0.1)

If Belief = *Low* (0.7) **and** Contribution = *Large* (0.3) **then** Likelihood Range = *Probable* (0.3)

If Belief = *Low* (0.7) **and** Contribution = *Very Large* (0.7) **then** Likelihood Range = *Probable* (0.7)

Figure 8 now shows the membership functions for Likelihood Range. Note that this is a decreasing value size logarithmic scale on the positive X-axis. The sub-ranges for the membership functions are derived from MIL-STD-882 (ref. 15).

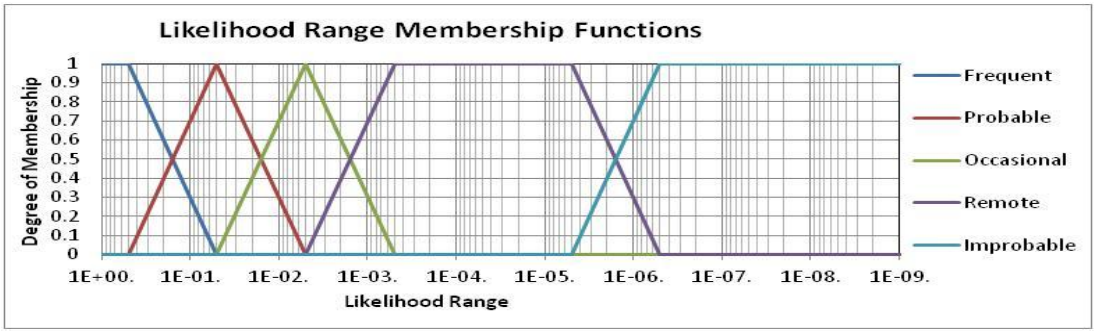

Figure 8

From the Likelihood Range membership functions, a composite membership polygon is created. The individual modified membership functions (see Figure 9) create membership polygons which are combined to form a composite membership polygon (see Figure 10). The highest membership of the "Frequent" value is 0.1 and for the "Probable" value is 0.7. No other membership functions were intersected. The composite membership polygon is the dotted black line in Figure 10.

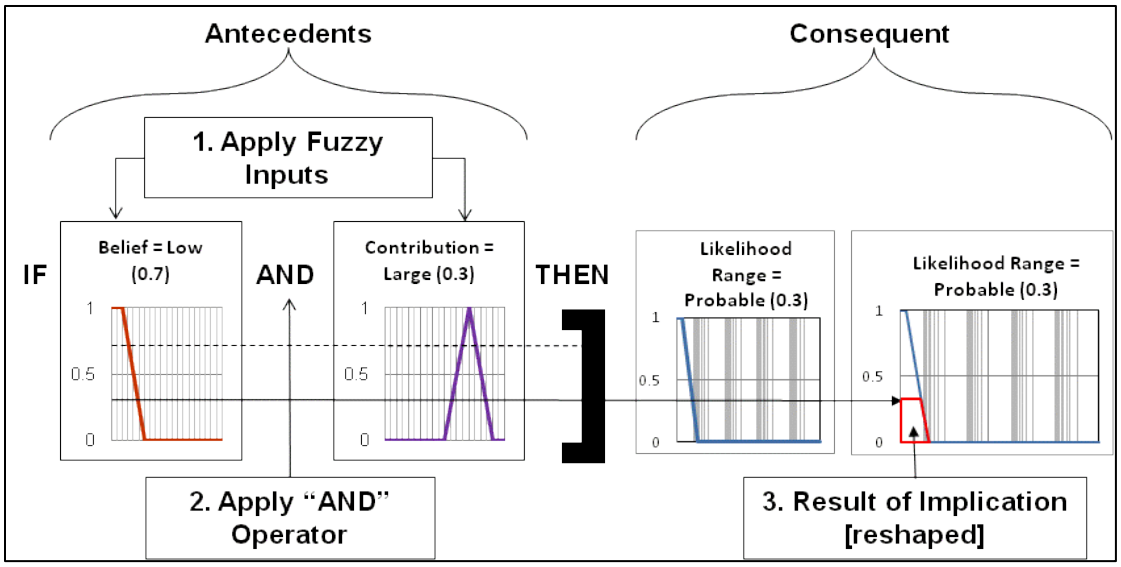

Figure 9

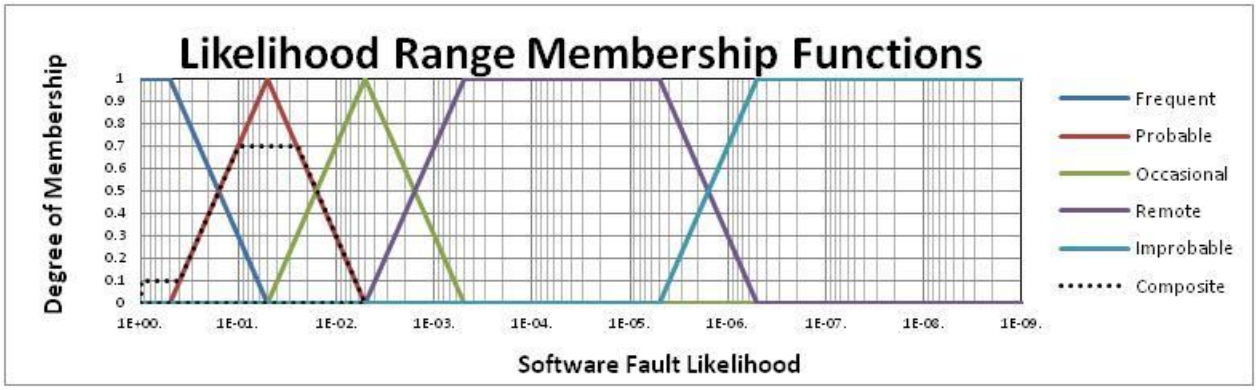

Figure 10

The FLARE process uses a conservative approach and chooses the Likelihood Range value with the highest degree-of-membership, *i.e.* the most possible, to estimate the likelihood for software safety failure. If the degrees-of-membership are equal, FLARE chooses the left-most membership function (highest likelihood) on the graph. The result for the example data is Likelihood Range = "Probable" (see Table 5).

Table 5: Likelihood Range

List of Objectives	LOR	Complete (%)	Quality (%)	Contribution (%)	Likelihood Range (P)_
FHA is developed	A	64	28	84	Probable

Estimating Risk Possibility: In order to express this likelihood in terms of qualitative risk the likelihood must be paired with the SwCI severity. The FLARE team is currently examining approaches for this calculation. This section discusses one approach currently being evaluated.

Using Likelihood Range value "Probable" and the Severity Category of "Catastrophic" from the example data the Software Risk Possibility (SRP) is 1B (see Table 6 and Table 7). The color coding in the SRP table corresponds to risk acceptance levels High (red), Serious (orange), Medium (yellow), and Low (green).

Table 6: Software Risk Possibility

Software Risk Possibility (SRP)		Likelihood Range (P)				
		Frequent (A)	Probable (B)	Occasional (C)	Remote (D)	Improbable (E)
SHCI Severity	Catastrophic (1)	1A	1B	1C	1D	1E
	Critical (2)	2A	2B	2C	2D	2E
	Marginal (3)	3A	3B	3C	3D	3E
	Negligible (4)	4A	4B	4C	4D	4E

Table 7: Qualitative Risk 1

List of Objectives	DAL Level	Complete (%)	Quality (%)	Contribution (%)	Likelihood Range	SRP	Qualitative Risk
FHA is developed	A	64	28	84	Probable	1B	High

Using the FLARE process allows the compliance evidence for each SwCI objective to be assessed independently from all other SwCI objectives. This in turn allows the analyst to portray the specific SwCI objectives which need the most attention. For example, if all the SwCI objectives for the example hazard information are assessed the results would provide the SRP value and qualitative risk for each objective. The qualitative software risk information can be portrayed with intrinsic resource allocation priorities for risk reduction activities. In Table 8 we assume two out of the 108 SwCI-A objectives contribute "Frequent" likelihood of software safety failures and 106 objectives contribute "Improbable" likelihood. Since "Catastrophic" severity and "Frequent" likelihood indicate the overall risk is "High", the program office (PO) will need to reduce the "Frequent" likelihood for two specific objectives to the "Improbable" range in order to accept the residual risk without higher command approval. With this method, the PO can target unique risk reduction actions to specific SwCI objectives based on the analysis details.

Table 8: Software Risk Category

Software Risk Category (SRC)		Likelihood Range (P)				
		Frequent (A)	Probable (B)	Occasional (C)	Remote (D)	Improbable (E)
SHCI Severity	Catastrophic (1)	**2**	0	0	0	106
	Critical (2)	0	0	0	0	0
	Marginal (3)	0	0	0	0	0
	Negligible (4)	0	0	0	0	0

All the Likelihood Range values in the example need to be "Improbable" at the least in order to lower the overall qualitative SRP to Medium (yellow colored blocks in Table 8). Table 9 shows the risk gaps in terms of percent Complete and percent Quality.

Table 9: Qualitative Risk 2

List of Objectives	DAL Level	Complete (%)	Quality (%)	Contribution (%)	Likelihood Range	SRP	Qualitative Risk
FHA is developed	A	Increase score from 64 to 81	Increase score from 28 to 81	84	Improbable	1E	Medium

The Completeness and Quality gaps are now known in terms of percent. This knowledge must be transitioned into actions that close or minimize the compliance gaps. Since the analyst has already reviewed the SwCI compliance evidence it is assumed the analyst kept a log of the review. The log may look similar to Table 10. From the information contained in the review log the analyst can very specifically identify recommendations to assist the developer in providing the necessary compliance evidence that would lead to achieving the desired risk category.

Table 10: Analyst's Log

#	Date	Page	Section	Paragraph	Comment Text (Provide clear succinct comments)	Recommendation (Must provide recommended rewording or appropriate solution)	Rationale	Comment Initiator

Summary and Future Directions

The FLARE process incorporates and uniquely handles four difficult issues that plague software system safety hazard analyses: (1) estimating software failure probability is very difficult and expensive, (2) decisions are subjective, (3) data are imprecise, and (4) software safety risk is never quantified or qualified. Two key advantages of FLARE are specific (highly focused) risk reduction activities can be recommended to the PO and/or developer and qualitative software risk estimates can be compared on par with hardware and operations risk estimates. Almost every step in the FLARE process can be tailored to a program's unique requirements and the FLARE process is easily automated.

During the development of the basic FLARE process, the team encountered several items that require additional examination. Among the high interest items are the analyst's belief in the rules and membership functions stated previously. This is distinct from the belief in the data sets and it requires additional steps to account for this factor. These steps are not addressed in this paper and require further research. A second high interest item is utilization of a fuzzy mathematical approach as an alternative to the rule based approach presented in this paper.

Acknowledgements

The authors want to thank Mr. Clifton A. Ericson, Mr. James H. McDuffie (SAIC), Mr. Glenn Morris (SAIC), Ms. Rhonda Barnes (A-P-T Research, Inc.) and Ms. Melissa Emery (A-P-T Research, Inc.) for their insightful comments and contributions to this paper.

References

1. Joint Software System Safety Engineering Handbook, Version 1.0, 27 August 2010, pg 64.
2. MIL-STD-882, Revision E, 11 May 2012, pg 7, para. 3.2.28.
3. MIL-STD-882, Revision E, 11 May 2012, pg 14, para. 4.4.
4. MIL-STD-882, Revision E, 11 May 2012, pg 7, para. 3.2.29 and pg 12, Table III.
5. MIL-STD-882, Revision E, 11 May 2012, pg 95, para. B.2.2.5.d.
6. SED, *SED-SES-PMHFSA, Program Manager Handbook for Flight Software Airworthiness*, April 2010, pgs 6 – 10.
7. US Army Aviation and Missile Command, AMCOMR 385-17, AMCOM Software System Safety Policy, Appendix E, pg 73

8. SED, *SED-SES-PMHFSA, Program Manager Handbook for Flight Software Airworthiness*, April 2010, pg 32.
9. US Army Aviation and Missile Command, AMCOMR 385-17, AMCOM Software System Safety Policy, Appendix E, pg 28
10. Hazard Analysis Techniques for System Safety, Ericson, Clifton A., 2005
11. Possibility Theory An Approach to Computerized Processing of Uncertainty, Didier Dubois and Henri Prade, 1988
12. Fuzzy Sets and their Applications to Cognitive and Decision Processes, Zadeh, L.A., 1975
11. SED, *SED-SES-PMHFSA, Program Manager Handbook for Flight Software Airworthiness*, 23 September 2011, pgs 36 – 38.
12. E. Raufaste, R. Da Silva Neves, C. Marine (2003), Testing the Descriptive Validity of Possibility Theory in Human Judgments' of Uncertainty. Artificial Intelligence, 148: 197 – 218.
13. MIL-STD-882, Revision E, 11 May 2012, pg 91, Table A-II.

Biographies

Willie J. Fitzpatrick, Jr., Ph.D., U.S. Army, Research, Development, and Engineering Command, Redstone Arsenal, AL 35898, USA, telephone − (256) 876-9945, facsimile − (256) 876-9950, e-mail − willie.fitzpatrick@us.army.mil. Dr. Fitzpatrick has over 36 years of experience in the software/systems engineering area. His experience includes the development and assessment of automatic control systems, systems engineering, and software engineering on various aviation and missile systems. He is currently Chief of the Aviation Division, in the Software Engineering Directorate of the U.S. Army Research, Development, and Engineering Command's Aviation and Missile Research Development and Engineering Center. Dr. Fitzpatrick is responsible for the management of life cycle software engineering support and airworthiness assessments for several aviation systems, including the Apache, Blackhawk, Chinook, and Kiowa Warrior aircrafts. He is also the directorate's manager for software safety analysis and assessments for aviation and missile systems. He has co-authored several technical reports and publications. Dr. Fitzpatrick is a member of the System Safety Society and an active Senior Member of the IEEE. He served as Chair of the IEEE Huntsville Section during 2007-2008.

David Skipper, BS, Physics, PhD Physics, Senior Systems Engineer, SAIC, 6263 Hackberry Road, Redstone Arsenal, AL 35898, USA – telephone – (256) 842-1698, email – david.j.skipper@us.army.mil. Dr. Skipper has over 30 years experience in Software Development and Systems Engineering. This experience has included teaching at the University of Alabama – Huntsville and various U.S. Army projects. He is currently working on the Kiowa Warrior system at the Software Engineering Directorate (SED). Dr. Skipper is a member of the American Physical Society and the IEEE.

Jonathan McNeil Sr., BS, Electrical, Software Safety & Airworthiness Engineer, US Army AMRDEC Software Engineering Directorate, 6263 Hackberry Road, Attn: AMSRD-AMR-BA-AV, Redstone Arsenal, AL 35898, USA, telephone – (256) 876-4295, facsimile – (256) 876-9950, email – josh.mcneil@us.army.mil. Mr. McNeil received his BS in Electrical and Computer Engineering from the University of Alabama - Huntsville. Mr. McNeil has worked in the Aerospace Industry for 20 years as a Software Safety and System Safety Engineer. In his current position, Mr. McNeil is the Software Safety lead and UAS Software Airworthiness lead for the Aviation Missile Research Development and Engineering Center SED responsible for performing software safety analyses on various US Army military programs and software airworthiness assessments on various US Army Manned and Unmanned Aviation Systems (UAS). Mr. McNeil has given several tutorials and written numerous papers on software safety. He has also been an active member of System Safety Society (SSS) for over 17 years, serving as: the SSS Director of Publicity and Media (2001-2005); Executive Chair for the 19th International System Safety Conference (ISSC) (2001); and Past President of the Tennessee Valley Chapter (1997).

J.P. Rogers, BS, Mathematics, MS Space Systems, Software Safety Engineer, APT Research, Inc., 4950 Research Drive, Huntsville, AL 35805, USA, telephone – (256) 327-3704, facsimile – (256) 837-7786, email – jprogers@apt-research.com. Mr. Rogers holds a Bachelors Degree in Mathematics from Rollins College and Masters Degree in Space Systems from Florida Institute of Technology. He has 35 years experience in explosive ordnance disposal, managing and performing program analysis for space and ballistic missile launches, authoring FAA launch site and launch operator license requirements, developing methods for FAA license evaluations, developing FAA license analysis software, and developing risk prediction tools supporting public safety analysis of artillery and rocket firings. Mr. Rogers currently serves as a Software Safety Engineer supporting the Software Engineering Directorate

(SED) of the Aviation and Missile Research, Development, and Engineering Center (AMRDEC) where he evaluates hazard analyses, software program plans and program specifications to assess compliance with software airworthiness and safety requirements for U.S. Army missiles and missile launchers.