

A Value-Based Process for Achieving Software Dependability

Liguo Huang

Computer Science Department, University of Southern California,
Los Angeles, CA 90089-0781, USA
liguohua@usc.edu

Abstract. Since different systems have different success-critical stakeholders, and these stakeholders depend on the system in different ways, using traditional one-size-fits-all dependability metrics to drive the system and software development process is likely to lead to delivered systems that are unsatisfactory to some stakeholders. This paper proposes a Value-Based Software Dependability Achievement (VBSDA) process generated from the WinWin Spiral Model's risk-driven approach coupled with a set of value-based dependability analysis frameworks, methods, and models for reasoning about software and system dependability. It helps project success-critical stakeholders define, negotiate and develop mission-specific combinations of dependability attributes. The NASA/USC Inspector SCROver (ISCR) project is used as a case study to elaborate the process.

1 Introduction

The key objectives of the NASA High Dependability Computing Program (HDCP) are to develop NASA mission-relevant definitions of system and software dependability metrics, to use the metrics to drive development processes, and to evaluate the contributions of existing and new computing technologies to the improvement of an information-intensive system's dependability. Such evaluations require one or more evaluation criteria or metrics that enable quantitative comparisons of candidate technology solutions to be performed. Ideally, one would like to have a single dependability metric by which the development process could be driven, and by which the contributions of each technology could be ranked. However, in practice, such a one-size-fits-all metric is unachievable. Different systems have different success-critical stakeholders, and these stakeholders depend on the system in different ways [1].

This paper proposes a Value-Based Software Dependability Achievement (VBSDA) process generated from the WinWin Spiral Model's risk-driven approach coupled with a set of value-based dependability analysis frameworks, methods, and models for reasoning about software and system dependability. It helps project success-critical stakeholders define, negotiate and develop mission-specific combinations of dependability attributes. The NASA/USC Inspector SCROver (ISCR) project [2] is used as a case study to elaborate the process.

1.1 Case Study: NASA/USC Inspector SCROver (ISCR) Project

The ISCR system was developed to serve as a distributable HDCP testbed for evaluating current and emerging dependability-enhancing technologies. It involved obtaining requirements from the USC Department of Public Safety (DPS) for an autonomous robot that could investigate the possible presence of hazardous materials in an environment unfit or dangerous for human intervention. Such an environment could be caused due to an earthquake or a failed chemical/biological experiment in a chemistry/biological laboratory. It would have several risks, such as loss of human health or life due to failure to identify a dangerous target with chemical leak or radiation, and the damage of robot itself.

Here are the top-level requirements that were determined for the ISCR system. The robot shall be able to autonomously maneuver around in the area designated by the robot operator and identify the potentially hazardous targets. The robot shall simultaneously return pictures taken by the camera mounted on the robot and the available sensor information to the designated host computer. Additionally, the robot shall maintain enough power so that it can return back to the initial designated location.

The development of the SCROver was planned in 3 increments. The case study is based on the increment 3 of the project which covers most of the important mission scenarios.

2 Related Work

The International Federation for Information Processing Working Group WG 10.4 on Dependability Computing and Fault Tolerance (IFIP/WG 10.4) defines dependability as “the trustworthiness of a computing system which allows reliance to be justifiably placed on the service it delivers” [13]. IFIP/WG 10.4 also mentions that, in different circumstances, the focus will be on different properties of services such as continuity, performance, real-time response, etc. Therefore, dependability is not a single aspect of a computing system but a combination of various aspects of stakeholders’ interests such as availability, reliability, safety, confidentiality, integrity, maintainability and so on.

Because different stakeholders depend on different system capabilities in different situations, dependability is necessarily a multi-attribute construct with situation-dependent attribute values. At the Economics-Driven Software Engineering Research (EDSER) workshops (www.edser.org) and elsewhere [14, 15], researchers have developed general frameworks for making software engineering decisions that enhance the value of delivered software systems. Some contributions that explicitly address dependability aspects include

- Carnegie Mellon University’s work on value-based security investment analysis [16], warranty models for software [17], and value-based software fault detection [18]
- The University of Virginia’s application of real-options theory to the value of modularity [19] and application of utility-theory and stochastic control approaches to reliable delivery of computational services [20]

- University of Maryland’s work on modeling dependability for a diverse set of stakeholders [21]

3 Value-Based Software Dependability Achievement Process

Scenario-based stakeholder/value dependency analysis and risk-driven concurrent engineering are two critical factors emphasized in our VBSDA process. In order to avoid using one-size-fits-all metrics to measure the achievement of software dependability, we use a scenario-based approach to identify stakeholders’ value propositions with respect to dependability (D-) attributes and to help stakeholders define the detailed D-attribute requirements in different scenarios. This approach also helps to identify and resolve their value conflicts on D-attributes and to perform tradeoff analyses in order to engineer the stakeholder WinWin-balanced D-attribute requirements. Further the scenario-based approach enables us to use real earned value to monitor and control the progress toward achieving the D-attribute requirements. Finally, the VBSDA process generated from the WinWin Spiral Model provides a workable framework for dealing with risk-driven concurrency.

This section discusses the major steps in the VBSDA process and identifies other sources of methods and tools for elaborating each step. The most complete source is the “Guidelines for Model-Based (System) Architecting and Software Engineering (MBASE)” [3] at <http://cse.usc.edu/research/MBASE>. Shorter summaries are [4] and [12]. The top-level process steps are listed as following:

1. Identify top-level mission objectives and stages
 - including dependability objectives
2. Develop results chain and identify success-critical stakeholders and their top-level value propositions
3. Stakeholders negotiate mutually satisfactory (Win-Win) dependability (and other) goals and relevant mission scenarios
4. Concurrently engineer top-level D-attribute and other requirements and solution tradeoff spaces
5. Identify top-level risks, execute risk-mitigation spirals
6. Develop initial Feasibility Rationale; hold Life Cycle Objective Review
 - Pass: go to 7. Fail: go to 5.
7. Concurrently engineer detailed D-attribute and other requirements and solutions; resolve risks
8. Develop detailed Feasibility Rationale; hold Life Cycle Architecture Review
 - Pass: go to 9. Fail: go to 7.
9. Construct, test, and deploy system
 - Use the mission scenarios and D-attribute requirement levels as progress metrics and test cases
 - Monitor progress and change requests; perform corrective actions

3.1 Identify Top-Level Mission Objectives and Stages

The top-level objectives and stages for the Inspector SCROver were summarized in Section 1.1.

3.2 Develop Results Chain and Identify Success-Critical Stakeholders

The Results Chain technique, developed by the DMR Consulting Group [5] is a way to identify missing initiatives and success-critical stakeholders in a system development project. It involves initially defining the project's Initiatives (rectangles), Contributions (arrows), Outcomes (circles, ovals), and Assumptions (hexagons) for its nominal-case operation. It then involves identifying risks and vulnerabilities that may go wrong with the initial Results Chain, and establishing additional Initiatives, Contributions, and Outcomes to avoid or resolve them.

Fig. 1 shows the dependability-elaborated Results Chain for developing the Initial Operational Capabilities (IOC) of the ISCR increment 3 operational scenarios. We have omitted the Assumptions for simplicity, but added the identification of success-critical stakeholders in parallelograms. Note that the text in *italic* shows the original simple initial Results Chain for the project developing the Initial Operational Capability (IOC) of the ISCR increment 3 information processing (IP) and operational capabilities without the dependability considerations. The full Results Chain identifies additional success-critical Initiatives, such as prevention and avoidance of ISCR risks and vulnerabilities (R&Vs), training operator and maintainers. Besides the Acquirers and Developers identified in the simple initial Results Chain, the additional dependability initiatives identify success-critical stakeholder class (Dependability Experts), and also the employment of additional dependability-enhancing tools and techniques such as verification and validation (V&V). Other success-critical stakeholders are also identified whose inputs are needed for the risk and vulnerability analysis: ISCR System Dependents (i.e. USC Lab Faculty, Students and Staff), Operators and Maintainers.

3.3 Stakeholders Negotiate Dependability Goals and Relevant Mission Scenarios

Table 1 identifies a matrix of the primary ISCR success-critical stakeholders as rows and their prioritized goals with respect to the ISCR system development, operation, and evolution as columns. The specific columns represent the primary categories of system requirements to be negotiated by the stakeholders.

Project goals and requirements include desired constraints on the system and project such as choices of programming language, infrastructure packages, and computing platforms; development and operational standards; and constraints on budgets, schedules, and other scarce resources as listed in Table 2. Capability goals include the functions the SCROver should perform. Interface goals include message formatting and content, and interaction protocols with other interoperating systems. Level of Service goals include the dependability attributes, except for cost and schedule (covered under Project goals) and interoperability (covered under Interface goals). Evolution goals include downstream goals that the initial system architecture should support, such as deferred capabilities or scalability to accommodate workload growth.

Instead of using one number to define the ISCR system availability goal, we distinguished three classes of mission scenarios. As shown in Table 3, the ISCR system

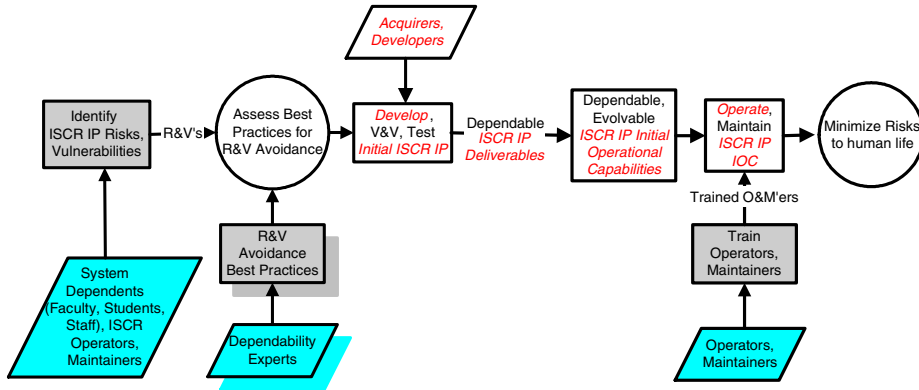


Fig. 1. Dependability-Elaborated Results Chain for ISCR Increment 3

Table 1. Inspector SCROver (ISCR) Stakeholder/Goal Matrix (Priorities: High, Medium, Low)

Goals \ Stakeholders	Project	Capability	Interface	Level of Service	Evolution
ISCR System Dependents (USC lab faculty, students, staff, etc.)			▶ Table 3		▶
ISCR Operators	Table 2				
ISCR Acquirers (USC DPS)					
ISCR Developers, Maintainers (USC CSE)	↓				

Table 2. Inspector SCROver (ISCR) Stakeholder/Goal Matrix (Priorities: High, Medium, Low)

Goals \ Stakeholders	Project Goals	Priority
ISCR System Dependents (USC lab faculty, students, staff, etc.), Operators	Develop an autonomous mobile robot that shall help the USC DPS perform its goals of investigating hazardous agents in the USC labs.	H
	Post-Mission data analysis	M
ISCR Acquirers (USC DPS)	Acquire the Core Initial Operational Capabilities (IOC) within budget and schedule	H
ISCR Developers, Maintainers (USC CSE)	Develop IOC within \$200K and 9 months	H
	Use MDS (Mission Data System) Framework	H

Table 3. Inspector SCROver (ISCR) Stakeholder/Goal Matrix II: ISCR System Dependents/Operators Goals and Priorities (High, Medium, Low)

Goals \ Stakeholders	ISCR System Dependents/Operators
Project	...
Capability	...
Interface	...
Level of Service	H: Availability ≥ 0.9998 for ISCR mission critical scenarios H: Availability ≥ 0.993 for ISCR on-line operational scenarios M: Availability ≥ 0.807 for ISCR post-mission data analysis scenarios H: Accuracy of Target Sensing $\geq 99\%$...
Evolution	...

dependents’ and operators’ goals for system availability are 0.9998 for mission-critical scenarios, 0.993 for on-line operational scenarios, and 0.807 for post-mission data analysis scenarios. Such numbers are traditionally difficult to determine. We will show how the Information Dependability Attribute Value Estimator (iDAVE) [6] helps determine them.

iDAVE Analysis of ISCR Availability Goals. Multiple stakeholder negotiation of ISCR system goals involves a mix of collaborative win-win option exploration with prototyping and analysis of candidate options. Here, the iDAVE model can be used to help the stakeholders determine how much availability is enough for the three primary classes of ISCR scenarios. Table 4 shows the key availability-related parameters for the software related to the three classes of ISCR scenarios; the size in thousands of source lines of code (KSLOC), the cost per line of code and total cost independent of investments in software reliability, and the dollar mission value of risk if the class of the scenarios fails. For example, there is 15 KSLOC of software for mission-critical scenarios: *Target Sensing* and *Target Rendezvous*. Its cost per instruction of a Nominal COCOMO II Required Reliability level is \$6.24/LOC (at graduate-student labor rates), leading to a nominal cost of \$93.6K. A failure in the mission-critical software is likely to cause complete contamination and replacement of the robot and the lab, with an impact equal to the \$2.5M of an entire lab. A failure and loss of availability of the on-line operational ISCR scenario (i.e., display continuous video images and sensor data to operator) would require repair and rerun of the mission, possibly losing \$200K of lab equipments. A failure of post-mission data analysis would require debugging, fixing, and regression testing the software, typically costing about \$14K.

Table 5 summarizes an iDAVE analysis of the return on investment involved in increasing the reliability level from Nominal to High; High to Very High; and Very High to Extra High. As determined from the calibrated parameters in the COCOMO II [22], and COQUALMO [23] models on which iDAVE is based [6], increasing the reliability level of the ISCR On-Line Operational software from Nominal to High involves an additional \$45K(0.10) = \$4.5K investment. It results in an increase in MTBF from 300 to 10,000 hours, which at an experienced-based Mean Time To Repair (MTTR) of 72 hours results in an increase in availability from .807 to .993. Using a linear relation between fraction of downtime and fraction of lost value as in [24], this 0.186 increase applied to the \$200K risk impact of the On-Line Operational scenario results in an added benefit of \$200K (0.186) = \$37.2K, and a resulting ROI = (37.2-4)/4 = 7.29. However, an additional \$45K (0.16) = \$7.19K investment to take the software from High to Very High only gains in a (\$200K)(.9998-.993)=\$1.38K benefit, for a negative ROI of -0.81.

Table 4. Size, Cost, and Risk Impact of Three Classes of SCROver Scenarios

Classes of Scenarios	Size (KSLOC)	Nominal		Risk Impact (\$K)
		\$/LOC	\$K	
Mission-Critical	15	6.24	93.6	2500
Online-Operational	8	5.62	45	200
Post-Mission Data Analysis	6	4.48	26.9	14

Table 5. iDAVE Analysis of ISCR Availability Goals for Three Classes of Scenarios

COCOMO RELY Level	Nom	High	Very High	Extra High
MTBF(hrs)	300	10,000	300,000	1,000,000
Availability (MTTR=72hrs)	.807	.993	.9998	.99993
Incremental Availability		.186	.0069	.00001
Incremental Cost		0.10	0.15	0.24
Mission-Critical				
Incr. Cost @ \$93.6K		\$9.36K	\$14.55K	\$16.84K
Incr. Benefit @ \$2.5M		\$466K	\$17.27K	\$.42K
ROI = (B-C)/C		+48.8	+0.15	-0.98
Online-Operational				
Incr. Cost @ \$45K		\$4.5K	\$7.19K	
Incr. Benefit @ \$200K		\$37.28K	\$1.38M	
ROI = (B-C)/C		+7.29	-0.81	
Post-Mission Data Analysis				
Incr. Cost @ \$26.9K		\$2.69K	\$4.3K	
Incr. Benefit @ \$14K		\$2.61K	\$100	
ROI = (B-C)/C		-0.03	-0.98	

This and the ROI results for the other two classes of ISCR scenarios calculated in Table 5 are summarized in Fig. 2. The incremental cost of achieving the higher availability levels still keeps the total cost below \$200K. From a pure calculated ROI standpoint, one could achieve some potential savings by interpolating to find the availability-requirement levels at which the ROI goes from positive to negative, but it is best to be conservative in a safety-related situation. Or one can identify desired and acceptable availability levels to create a tradeoff space for balancing availability with other dependability attributes.

3.4 Concurrently Engineer Top-Level D-Attribute and Other Requirements and Solution Tradeoff Spaces; Identify Top-Level Risks and Execute Risk-Mitigation Spirals

Step 4 and 5 are often coupled with each other during the software development process. In this section, we propose a scenario-based approach to identify stakeholders' value propositions on dependability attributes and help stakeholders define the detailed D-attribute requirements in different scenarios. The approach also helps identify and resolve value conflicts on dependability attributes and to perform tradeoff analysis on dependability attributes in order to engineer stakeholder WinWin-balanced dependability attribute requirements. Fig. 3 shows the process elements for stakeholders to engineer top-level dependability attribute requirements, identify dependability risks and select the most cost-effective dependability technology combination to mitigate risks for different scenarios. The entry criteria of the D-attribute requirement engineering and risk mitigation process are shown in the box in the top-left corner of Fig. 3.

E1. Identify dependability (D) attributes

This is the entry of the process where the top-level dependability attributes for the whole project are established. The results obtained from Step 1 and 3 in the VBSDA process are usually used as the inputs of this step

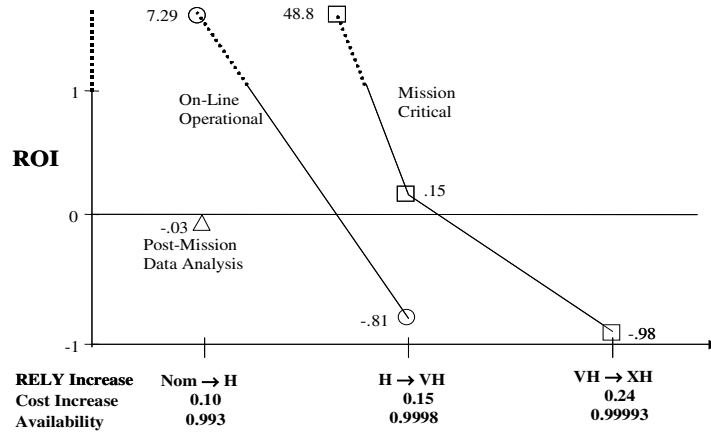


Fig. 2. Summary of iDAVE Analysis of ISCR Availability Goals

The top-level dependability attributes for the ISCR project are availability, accuracy, performance, usability, cost and schedule.

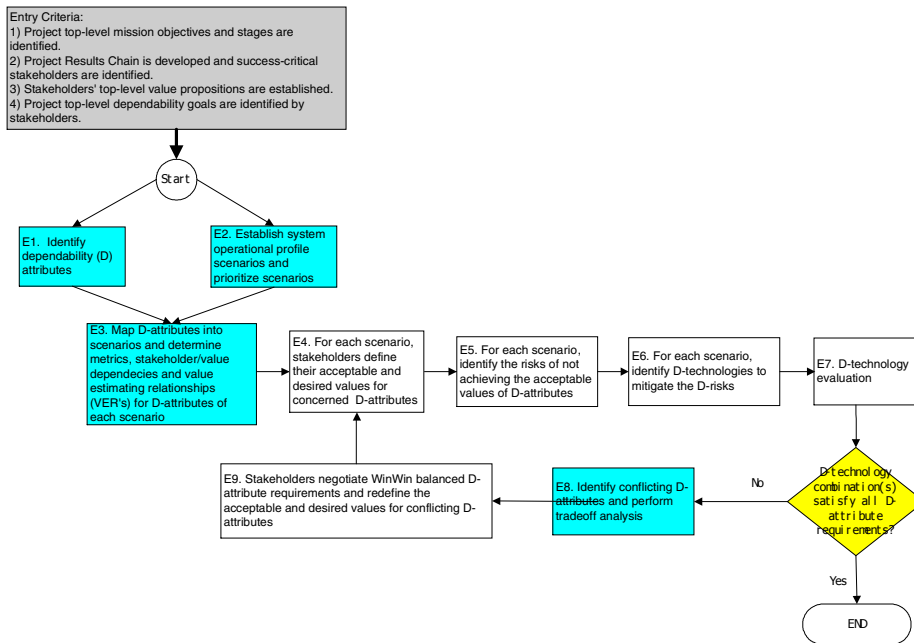


Fig. 3. A Scenario-Based Approach to Engineer Dependability Requirements and Risk Mitigation Plans

E2. Establish system operational profile scenarios and prioritize scenarios

The scenarios can be defined as mission sequences, environmental inputs or D-objective threats and their frequencies. A scenario is used to describe a proposed use

case of the system and/or an interaction of one of the stakeholders with the system [7]. Scenarios provide a vehicle for converting vague dependability attributes/requirements into concrete use cases of a system and make dependability attributes/requirements measurable and testable. The top-level scenarios of a software system can be established from its use case description (e.g., MBASE Operational Concept [3] use case description). A complex scenario can be decomposed into several component scenarios if it's necessary for testing purposes. On the other hand, several component scenarios can be composed into a high-level scenario for analysis or testing purposes. We provide a framework with three factors to be associated with each scenario S_i , which will be directly leveraged in our scenario-based approach:

- **Value (v).** The value loss (can be measured either in dollars or in utility) if a scenario execution fails. It indicates the impact of a scenario on the total mission value.
- **Probability of occurrence (p).** The probability that a scenario occurs in a specific mission mode. When several scenarios have the comparable value impact on the entire mission, a scenario that is more frequently executed affects the system dependability more extensively, than if it were less. In a given mission mode, $\sum p_i = 1$. The operational profile of a mission mode can then be established based on the scenario probability distribution.
- **Dependability attribute metrics (m).** All scenarios are mapped into dependability attributes based on their relevance, as we have shown for availability in section 3.3.

Scenarios can then be prioritized based on their *value (v)* and *probability of occurrence (p)*.

The operational scenarios of the ISCR initial operational capability (IOC) identified by the stakeholders are shown in the Table 6. The lower-priority scenarios can be added in post-IOC increments.

E3. Map D-attributes into scenarios and determine metrics, stakeholder/value dependencies and value estimating relationships (VER's) for D-attributes of each scenario

The D-attributes are mapped into each scenario based on their relevance. The metric for a D-attribute may be different in different scenarios. For instance, Performance can be measured in response time (s) or in storage space (MB) in different scenarios.

To understand the nature of the software system dependability, we have to identify the major classes of success-critical project stakeholders, and to characterize the relative strengths of their dependencies on various attributes of each scenario of the software system [1]. Table 7 shows the top-level direct stakeholder/value dependencies on the D-attributes in the *Target Sensing* scenario. Acquirers, developers, and maintainers are not directly concerned with availability and accuracy, but become concerned with them when their operational stakeholders are.

If needed, the value estimating relationships (VER's) of each D-attribute can be also established based on the impact of the D-attribute on a particular scenario. Note that the VER's for a D-attribute may also be different in different scenarios since the same D-attribute's impact on different scenarios may be different.

Table 6. ISCR Increment 3 Operational Profile Scenarios

Scenarios	Component Scenarios	Priority
Target sensing		H
Target rendezvous	Trajectory planning	H
	Localization	
	Obstacle avoidance	
Display environment state information to operator	Return target state info state variable to operator	H
	Return terrain state variable to operator	
Display sensor and actuator health state information to operator	Return camera state variable to operator	H
	Return range finder health state variable to operator	
	Return wheel motor health state variable to operator	
	Return battery state of charge to operator	
Display continuous camera video images to operator		M
Post-mission data analysis		L
Goal conflict identification and resolution		L

Table 7. Target Sensing Scenario: Top-level Stakeholder/Value Dependencies on D-attributes (** Critical, * Significant, () Insignificant or indirect)

Stakeholders D-attributes	System Dependents	Operators	Acquirers	Developers	Maintainers
Availability	*	**			*
Accuracy		**			
Cost			**	*	
Schedule			**	**	
Evovability					**

E4. For each scenario, stakeholders define their acceptable and desired values for concerned D-attributes

The results of the iDAVE ROI analysis for three ISCR scenario classes discussed in section 3.3 can be used as guidance for stakeholders to define their expected and desired levels for D-attributes based on the priority of a particular scenario and their value dependencies on the scenario-related D-attributes.

E5. For each scenario, identify the risks of not achieving the acceptable values of D-attributes

E6. For each scenario, identify the D-technologies to mitigate the risks

E7. D-technologies evaluation

Scenario-based Fault Tree Analysis (FTA) [8], Failure Modes and Effects Analysis [8] and Dependability Cases [9] are three useful techniques to trace scenario failures to the potential risks causing them.

Risks are quantitatively linked to the D-attributes of each scenario. For each pair of risk and D-attribute in each scenario, stakeholders provide an estimate (expert judgement) of the potential impact of the risk in the scenario. We define the “impact” as the proportion of the scenario value that would be lost were that risk occur. The probability of occurrence of each risk is also estimated. D-technologies are quantitatively linked to risks. For each pair of risk and D-technology in each scenario, we provide an estimate (expert judgement) of the mitigation of the risk in the scenario. We define the “mitigation” as the proportion by which the risk would be reduced were that D-technology to be applied. At the same time, the cost/effort of applying a particular D-technology should also be recorded. Tools such as the JPL Defect Detection and Prevention (DDP) model [25] can be used for such analysis.

In addition, the parameters used in the iDAVE analyses may be based on incompletely-validated assumptions such as the scalability of a model-checking tool. This is also identified as a risk, and a risk-mitigation plan to validate the scalability of the model-checking tool is developed and executed. The next step is to look for conflicting combinations of D-attributes.

E8. Identify conflicting D-attributes and perform tradeoff analysis

E9. Stakeholders negotiate WinWin balanced D-attribute requirements and redefine the acceptable and desired values for conflicting D-attributes

If the existing technologies can't satisfy the acceptable values of all the D-attributes, or if the estimated cost/schedule to satisfy all the D-attribute requirements is too high, then the tradeoff function between the conflicting D-attributes will need to be constructed and the tradeoff analysis will be performed in conjunction with additional stakeholder negotiation. Multi-attribute preference analyses [10] and stakeholder win-win negotiation support tools [26] are useful techniques to help stakeholders perform such negotiations based on the stakeholders' value propositions. Note that the conflicting D-attribute tradeoff analysis can be performed concurrently with the D-technology evaluation.

3.5 Develop Initial Feasibility Rationale; Hold Life Cycle Objective Review

The initial Feasibility Rationale Description (FRD) [3] furnishes the rationale for the product being able to satisfy the stakeholders' system requirements and specifications. The initial FRD in LCO stage includes an initial business case analysis (i.e., cost, benefits and ROI analysis) based on the Results Chain.

Then a Life Cycle Objective (LCO) Review is to be held with the participation of all the project key stakeholders, and independent experts (these were NASA Jet Propulsion Laboratory (JPL) planetary mission software experts for ISCR). This indicates a milestone of the LCO phase in the WinWin Spiral Model. The exit criteria of LCO Review are to provide at least one feasible architecture to satisfy the requirements, and to provide proofs of requirement satisfaction including the dependability requirements.

The initial risk analysis should identify all the major risks and propose an initial risk mitigation plan. Risks without mitigation in LCO stage have to be resolved in Life Cycle Architecture (LCA) stage.

The result of LCO ARB was to Pass and go to Step 7. However, a risk was identified that the tool evaluation needs for the HDCP tool researchers had been incompletely defined for ISCR.

3.6 Concurrently Engineer Detailed D-Attribute and Other Requirements and Solutions; Resolve Risks

Thus, the major new activity in Step 7 involved surveying HDCP interventionists for additional evaluation needs. The primary emerging need identified was for a three-dimensional graphic uses interface (3D GUI).

Originally, developers planned to use Player/Stage as the robot simulator platform. Because of the 3D GUI requirement, we had to reevaluate the existing technologies or identify new technologies to cover this D-risk. In this case, since Stage doesn't

support a 3D GUI, the developers had to find a replacement. After the evaluation, stakeholders finally chose Gazebo because it supports the new stereo camera model and a 3D GUI which also enabled most devices to be directly controlled/inspected through the simulator GUI. Since Stage and Gazebo are both Player-compatible, client programs written using one simulator can usually be run on the other with little or no modification. The key difference between these two simulators is that whereas Stage is designed to simulate a very large robot population with low fidelity, Gazebo is designed to simulate a small population with high fidelity [27]. Thus Gazebo fits with most of DPS missions which can be accomplished by a few robots. Furthermore, Gazebo is more valuable to stakeholders since it improved the usability and evolvability of the system.

3.7 Develop Detailed Feasibility Rationale; Hold Life Cycle Architecture Review

The Life Cycle Architecture (LCA) Review was held with the participation of all the project key stakeholders and the JPL experts. The exit criteria of LCA Review is to commit one architecture to satisfy all the requirements of the system. Thus the LCA FRD has to provide detailed proofs of all requirement satisfaction including the dependability requirements.

The LCA FRD risk analysis should propose a detailed risk mitigation plan to resolve all known risks.

The result of the LCA ARB was again to Pass, and proceed to Step 9.

3.8 Construct, Test, and Deploy System

ISCR Increment 3 is currently under development. It is using the dependability scenarios to simulate the ISCR performance and evaluate whether the dependability attribute levels will be achieved. The framework of the value-realization feedback process [11] is shown in Fig. 4.

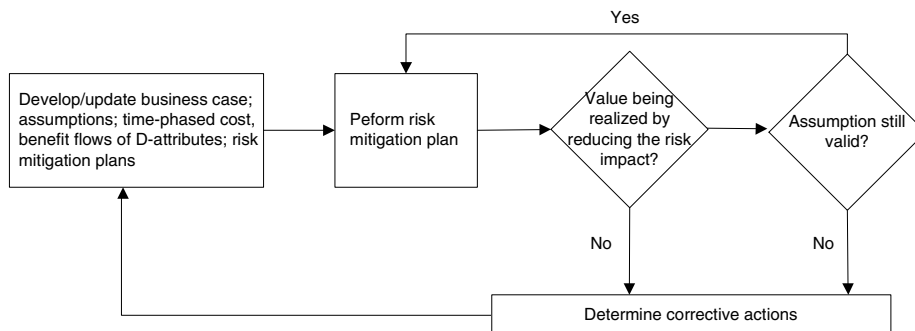


Fig. 4. A Value-Realization Feedback Process to Monitor and Control the Achievement of D-attribute Requirements

A matrix with the capability to track the value-based expected versus actual outcomes (i.e., dependability cost, reduced value loss, ROI) is a useful technique to support the monitoring and control of the actual progress of the dependability achievement. A case study on how to perform the value-based monitoring and control using such a matrix is discussed in [11].

4 Conclusions and Lessons Learned

In practice, value-dependencies on D-attributes vary significantly by stakeholders and scenarios. The universal one-size-fits-all metrics for software dependability (SWD) achievement are unachievable in most project situations. We need to balance stakeholders' value propositions on D-attributes. Thus, a critical first step in understanding the nature of information system dependability is to identify the major classes of success-critical information system stakeholders, and to characterize the relative strengths of their dependencies on various attributes of a given software system. Furthermore, stakeholder value dependencies are often in conflict and require negotiated situations. There is an increasing need for value-based approaches to SWD analysis and its achievement monitoring and control.

The iDAVE model provides a technique for reasoning about the ROI of software dependability. It helps project decision-makers determine how much dependability investment is enough based on their project's business case. It also provides a way to define different dependability levels for different software classes or scenarios based on stakeholders' value propositions, which can avoid the one-size-fits-all dependability metrics for a project. Based on the stakeholder/value dependency analysis frameworks, and value-based methods and models for reasoning about software and system dependability, we have developed and are experimentally applying a value-based process for using these artifacts integrated with the scenario-based approach to cost-effectively achieve desired dependability attribute levels for a given project, and for improving the cost-effectiveness of an organization's capability to develop dependable software and systems. It helps project success-critical stakeholders define, negotiate and develop mission-specific combinations of dependability attributes. To date, the value-based process and scenario-based approach have enabled us to successfully perform value-based feedback control of the actual progress of the ISCR project's dependability achievement.

Acknowledgements

This work was supported by NASA-HDCP contract to University of Southern California. This material is also based upon work supported by the National Science Foundation. The author specially thanks the valuable inputs and comments from Dr. Barry Boehm. And she also appreciates the help of Zhihao Chen in USC CSE in preparing this paper.

References

1. Boehm B., Huang L., Jain A., and Madachy R.: The Nature of Information System Dependability: A Stakeholder/Value Approach (Draft 6). USC-CSE (2004)
2. USC-CSE Inspector SCROver Project Team: Inspector SCROver Project, <http://cse.usc.edu/iscr/pages/ProjectDescription/home.htm> (2004)
3. Guidelines for Model-Based (System) Architecting and Software Engineering (MBASE), <http://cse.usc.edu/research/MBASE>. USC-CSE (2003)
4. Boehm B. and Hansen W.: Understanding the Spiral Model as a Tool for Evolutionary Acquisition. CrossTalk May (2001) 4-11

5. Thorp J. and DMR: The Information Paradox. McGraw Hill (1998)
6. Boehm B., Huang L., Jain A. and Madachy R.: The ROI of Software Dependability: The iDAVE Model. *IEEE Software*, Vol. 21, No. 3, May/June (2004) 54-61
7. Clements P., Kazman R., and Klein M.: *Evaluating Software Architecture: Methods and Case Studies*. Addison Wesley (2002)
8. Leveson N. G.: *Safeware, System Safety and Computers*. Addison Wesley (1995)
9. Weinstock C. B., Goodenough J. B.: Dependability Cases. Technical Note, CMU/SEI-2004-TN-016 May (2004)
10. Keeney R. L., Raiffa H.: *Decisions With Multiple Objectives*. Cambridge University Press (1993)
11. Boehm B. and Huang L.: Value-Based Software Engineering: A Case Study. *IEEE Computer*, Vol. 36, No. 3, March (2003) 33-41
12. Boehm B. and Port D.: Balancing Discipline and Flexibility With the Spiral Model and MBASE. *CrossTalk*, Vol. 14, No. 12, December (2001) 23-28
13. Laprie J.C. (ed.): *Dependability: Basic Concepts and Terminology*. Springer-Verlag, Vienna (1992)
14. Reifer D.: *Making the Software Business Case*. Addison-Wesley (2002)
15. Nejme B. and Thomas I.: Business-Driven Product Planning Using Feature Vectors and Increments. *IEEE Software*, Nov./Dec (2002) 34-42
16. Butler S.: Security Attribute Evaluation Method: A Cost-Benefit Approach. Proc. of 24th Int'l Conf. Software Eng. (ICSE 02), IEEE CS Press (2002) 232-240
17. Li P., Shaw M., Stolarick K., and Wallnau K.: The Potential for Synergy between Certification and Insurance. Special edition of ACM SIGSOFT from the 1st International Workshop on Software Reuse Economics (in conjunction with the 7th International Conference on Software Reuse) (2002) <http://www.sei.cmu.edu/staff/kcw/icsr02.pdf>
18. Raz O. and Shaw M.: Software Risk Management and Insurance. Proceedings of 3rd International Workshop on Economics-Driven Software Engineering Research. IEEE CS Press (2001) <http://www.cs.virginia.edu/~sullivan/edser3/raz.pdf>
19. Sullivan K. et al.: Software Design as an Investment Activity: A Real Options Perspective. *Real Options and Business Strategy: Applications to Decision Making*. Trigeorgis L. (ed.), Risk Books (1999)
20. Cai Y. and Sullivan K.: Stochastic Optimal Switching. Proceedings of 4th Workshop on Economics-Driven Software Eng. Research, IEEE CS Press (2002) 71-72
21. D. Huynh, M. Zelkowitz, V. Basili, and I. Rus: Modeling Dependability for a Diverse Set of Stakeholders. The International Conference on Dependable Systems and Networks (2003) (DSN-2003) <http://hdcp.org/Publications/dsn-fast-abstract-031603.pdf>
22. Boehm B. et al.: *Software Cost Estimation with COCOMO II*. Prentice Hall (2000)
23. Steece B., Chulani S., and Boehm B.: Determining Software Quality Using COQUALMO. *Case Studies in Reliability and Maintenance*. Blischke W. and Murthy D. (eds.), John Wiley & Sons (2002)
24. DeMillo R.: Why Software Falls Down. *Mutation Testing for the New Century*. Wong W.E. (ed.), Kluwer Academic (2001)
25. Feather M. S., Cornford S. L., Dunphy J.: A Risk-Centric Model for Value Maximization. Proceedings of 4th Workshop on Economics-Driven Software Engineering Research, IEEE CS Press (2002)
26. WinWin Spiral Model & Groupware Support System. <http://sunset.usc.edu/research/WINWIN/index.html>
27. Koenig N. and Howard A.: Gazebo: 3D Multiple Robot Simulator With Dynamics. <http://playerstage.sourceforge.net/gazebo/gazebo.html>