HAL
open science

ARRL: A Criterion for Composable Safety and Systems Engineering

Eric Verhulst, de La Vara Jose Luis, Bernhard H.C. Sputh, de Florio Vincenzo

To cite this version:

Eric Verhulst, de La Vara Jose Luis, Bernhard H.C. Sputh, de Florio Vincenzo. ARRL: A Criterion for Composable Safety and Systems Engineering. SAFECOMP 2013 - Workshop SASSUR (Next Generation of System Assurance Approaches for Safety-Critical Systems) of the 32nd International Conference on Computer Safety, Reliability and Security, Sep 2013, Toulouse, France. pp.NA. hal-00848521

HAL Id: hal-00848521

https://hal.science/hal-00848521

Submitted on 26 Jul 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ARRL: A Criterion for Composable Safety and Systems Engineering

Eric Verhulst[1], Jose Luis de la Vara[1], Bernhard H.C. Sputh[1], and Vincenzo de Florio[3]

[1]Altreonic NV, Linden, Belgium, [2]Simula Research Laboratory, Lysaker, Norway
[3]PATS / Universiteit Antwerpen & iMinds research institute, Antwerpen, Belgium
{eric.verhulst,bernhard.sputh}@altreonic.com, jdelavara@simula.no,
vincenzo.deflorio@ua.ac.be

Abstract. While safety engineering standards define rigorous and controllable processes for system development, safety standards' differences in distinct domains are non-negligible. This paper focuses in particular on the aviation, automotive, and railway standards, all related to the transportation market. Many are the reasons for the said differences, ranging from historical reasons, heuristic and established practices, and legal frameworks, but also from the psychological perception of the safety risks. In particular we argue that the Safety Integrity Levels are not sufficient to be used as a top level requirement for developing a safety-critical system. We argue that Quality of Service is a more generic criterion that takes the trustworthiness as perceived by users better into account. In addition, safety engineering standards provide very little guidance on how to compose safe systems from components, while this is the established engineering practice. In this paper we develop a novel concept called Assured Reliability and Resilience Level as a criterion that takes the industrial practice into account and show how it complements the Safety Integrity Level concept.

Keywords: safety engineering, system engineering, safety standard, Safety Integrity Level, Assured Reliability and Resilience Level.

1 Introduction

One of the emerging needs of embedded systems is better support for safety and, increasingly, security. These are essentially technical properties. The underlying need is trustworthiness. Although systems engineering standards and in particular safety standards were developed to support achievement of these properties, these standards do not cover the full spectrum of trustworthiness. They aim to guarantee safety properties because they are concerned with the risk of people being hurt or killed. It is because of this risk that safety-critical systems are generally subject to certification as a legal requirement to put them in public use.

While safety standards exist, a first question that arises is why each domain has specific safety standards. They all aim to reduce the same risk of material damages and human fatalities to a minimum, so why are they different from one domain to another? One can certainly find historical reasons, but also psychological ones. Safety standards such as IEC 61508 [8] also concern mostly systems with programmable electronic components. The reason for this is that with the advent of programmable components in system design, systems engineering became dominantly a discrete domain problem, whereas the preceding technologies were dominantly in the continuous domain. Components have the inherent property of graceful degradation in the continuous domain, while this is not the case for discrete domain systems. A second specific trait is that, in the discrete domain, the state space is usually very large, with state changes happening in nanoseconds. Hence it is very important to be sure that no state change can bring the system into an unsafe condition.

Notwithstanding identifiable weaknesses, safety engineering standards impose a controlled engineering process resulting in relatively well predictable safety that can be certified by external parties. However, the said process is relatively expensive and essentially requires that the whole project and system is re-certified whenever a change is made. Similarly, a component such as a computer that is certified as safe for use in one domain cannot be reused as such in another domain.

This is often in contrast with the engineering practice. Engineers constantly build systems by composing and reusing existing components. This is not only driven by economic benefits but it often increases the trust in a system because the risk for residual errors will be lower, at least if a qualification process is in use. Nevertheless, engineering and safety standards contain very few rules and guidelines on reusing components, which hampers developing safe systems by composition.

This paper analyses why the current safety driven approach is unsatisfactory and introduces a new criterion called the Assured Reliability and Resilience Levels. This criterion allows components to be reused in a safety-critical context in a normative way, while preserving the safety integrity levels at the system level.

The rest the paper is organized as follows. Section 2 introduces the background of the paper. Section 3 presents the ARRL criterion. Section 4 reviews related work. Finally, section 5 presents our conclusions.

2 Background

This section presents SIL, analyses the differences of is application in different domains, compares it to Quality of Services (QoS) levels, and discusses the weaknesses of SIL application.

2.1 SIL

As safety is a critical property, it is no wonder that safety standards are perhaps the best examples of concrete systems engineering standards, even if safety is not the only relevant design property. Most domains have their own safety standards, partly for historical reasons, partly because the heuristic knowledge is very important, or because the practice in the domain has become normative. We consider first the IEC61508 standard [8] as it is relatively generic.

IEC61508 considers mainly programmable electronic systems. The goal is to bring the risks to an acceptable level by applying safety functions. IEC61508 is based

on the principle that safety is never absolute. It considers the likelihood of a hazard (a situation posing a risk) and the severity of the consequences. A third element is the controllability. The combination of these factors is used to determine a required SIL, categorized in four levels. These levels correspond to normative-allowed Probabilities of Failure per Hour, and require corresponding Risk Reduction Factors that depend on the usage pattern (infrequent vs. continuous). The risk reduction itself is achieved by combining reliability measures (higher quality) and functional measures, as well as by following a rigorous engineering process. The categorization is more or less as shown in Table 1, whereby we added a SIL-0 for completeness.

Table 1 SIL levels according to IEC61508

Category	**SIL**	**Consequence upon failure**
Catastrophic	4	Loss of multiple lives
Critical	3	Loss of a single life
Marginal	2	Major injuries to one or more persons
Negligible	1	Minor injuries or material damage only
No consequence	0	No damages, except user dissatisfaction

The SIL level is used as a directive to guide selection of the required architectural support and development process requirements. For example, SIL-4 imposes redundancy and positions the use of formal methods as highly recommended.

While IEC61508 has resulted in derived domain specific standards (e.g., ISO26262 for automotive [9] and EN50128 [4] for railway), there is no one to one mapping of the domain specific levels to IEC61508 SIL levels. Table 2 shows an approximate mapping whereby we added the aviation DO-178C standard [13], which was developed independently. It must be mentioned that the Risk Reduction Factors are vastly different as well. This is mainly justified by the usage pattern of the systems and the "fail safe" mode. For example, while a train can be stopped if a failure is detected, a plane must at all cost be kept in the air in a state that allows it still to land safely.

Table 2 Mapping of the safety levels of different domains

Domain	**Domain-specific Safety Levels**				
General (e.g., IEC61508)	(SIL-0)	SIL-1	SIL-2	SIL-3	SIL-4
Automotive (e.g., ISO26262)	ASIL-A	ASIL-B	ASIL-C	ASIL-D	-
Aviation (e.g., DO178C)	DAL-E	DAL-D	DAL-C	DAL-B	DAL-A
Railway (e.g., EN50128)	(SIL-0)	SIL-1	SIL-2	SIL-3	SIL-4

The SIL levels (or the domain specific ones) are mostly determined during a HARA (Hazard and Risk Analysis) executed before the development phase and updated during and after this phase. The HARA tries to find all hazardous situations and classifies them according to the three main criteria mentioned above (probability of occurrence, severity, and controllability). This process is however difficult and complex, partly because the state space explodes very fast, but also because the classification is often not based on historical data (absent for new systems) but on expert's opinion. It is therefore questionable if the assigned Safety Levels are accurate enough and if the Risk Reduction Factors are realistic. We elaborate on this further.

Once an initial architecture has been defined, another important activity is executing a FMEA (Failure Mode Effect Analysis). While a HARA is top-down and includes environmental and operator states, the FMEA analyses the effects of a failing component on the correct functioning of the system, and in particular in terms of the potential hazards. However, there is no criterion defined that allows us to classify components in terms of their trustworthiness, even if one can estimate parameters like MTBF (Mean Time Between Failures). In the last part of this paper we introduce a criterion that takes the fault behaviour into account.

2.2 Analysis of the Application of SIL

Sectors like railway and aviation are statistically very safe. As an example, about 1,000 people are killed every year worldwide in aircraft related accidents, which makes aviation the safest transport mode in the world [1]. In contrast, the automotive sector adds up to about 1.2 million fatalities per year worldwide and even developed regions like the USA and Europe experience about 35,000 fatalities per year (figures for 2010; [14]). Although both sectors have their safety standards, there is a crucial difference.

Whereas in most countries aircrafts and railway systems are strictly regulated and require certification, in the automotive sector the legal norms are much less strict, partly because the driver is considered as the main cause of accidents and not so much the road infrastructure or the vehicle itself. The latter biases significantly the "controllability" factor in the required SIL determination.

Taking a closer look at the SIL classifications of IEC61508 and the automotive derived ones in ISO26262, we notice three significant differences:

1. Whereas IEC61508 and ISO26262 both define four levels, they do not map to each other. In particular, SIL-3 and SIL-4 do not map to ASIL-C and ASIL-D.
2. The highest ASIL-D level corresponds to a SIL-3 level in terms of casualties, although it is not clear if this means a few casualties (e.g., not more than five like in a car) or several hundreds (like in an airplane).
3. The aviation industry experiences about 1,000 casualties per year worldwide, whereas the automotive industry experiences 1,200 times more per year worldwide, even in developed regions 35 times, whereby the automotive highest safety level is lower.

When we try to explain these differences, we identify the following reasons:

- ISO2626 was defined for automotive systems that have a single central engine. As a direct consequence, such a vehicle cannot be designed to be fault-tolerant and therefore cannot comply with SIL4, which mandates a fault-tolerant design.
- While ASIL-C more or less maps onto SIL-3 (upon a fault the system should transition to a fail-safe state), ISO26262 introduces ASIL-C requiring a supervising architecture. In combination with a degraded mode of operation (e.g., limp mode), this weaker form of redundancy can be considered as fault tolerant if no common mode failure affects both processing units [7].
- Automotive systems are not (yet) subject to the same stringent certification requirements as railway and aviation systems, whereby the manufacturers as well as the operating organization are legally liable. In general, the individual driver is often considered the responsible actor in case of an accident.

- The railway and aviation sectors are certified in conjunction with a regulated environment and infrastructure that contributes to the overall safety. Automotive vehicles are engineered with very flexible requirements in terms of where and when they are operated, and are used on a road infrastructure that is developed by external third parties. This partly explains why the high number of worldwide casualties is not reflected in the ASIL designation.
- One should not conclude from the above that a vehicle is by definition unsafe. It is however a bit of a contradiction that the SIL levels for automotive are lower than those for aviation and railway if one also considers the fact that vehicle accidents happen in a very short time interval and confined spaces with almost no controllability by the driver.

2.3 QoS Levels

An inherent weakness from the systems engineering and user's point of view is that safety is not the only contributing part to the trustworthiness of a system. A system that is being developed is part of a larger system that includes the user (or operator) as well as the environment in which the system is used (Fig. 1).

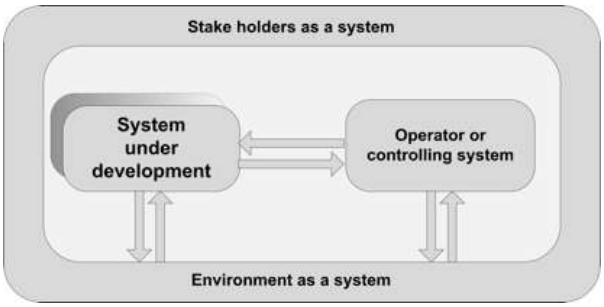

Fig. 1. A system in its larger context

Both additional systems do not necessarily interact in a predictable way with the envisioned system, and both have an impact on the safety properties and assurance. From the user's point of view, the system must deliver an acceptable and predictable level of service (i.e., a QoS level). A failure in a system is not seen as an immediate safety risk but rather as a breach of contract on the QoS whereby the system's malfunction can then result in a safety related hazard. As such we can see that a given SIL is a subset of the QoS. The QoS can be seen as the availability of the system as a resource that allows the user's expectations to be met. Aiming to reduce the intrinsic ambiguities of the Safety Levels we now formulate a scale of QoS as follows:

- **QoS-1** is the level whereby there is no guarantee that there will be resources to sustain the service. Hence the user should not rely on the system and should consider it as untrustworthy. When using the system, the user is taking a risk that is not predictable.
- **QoS-2** is the level whereby the system must assure the availability of the resources in a statistically acceptable way. Hence, the user can trust the system but knows that the QoS will be lower from time to time. The user's risk is mostly one of annoyance and dissatisfaction or of reduced service.

- **QoS-3** is the level whereby the system can always be trusted to have enough resources to deliver the highest QoS at all times. The user's risk is considered to be negligible.

We can consider this classification to be less rigorous than the SIL levels because it is based on the user's perception of trustworthiness and not on a combination of probabilities even when these are questionable (see section 2.4). On the other hand, QoS levels are more ambitious because they define minimum levels that must be assured in each QoS level. Of course, the classification leaves room for residual risks but those are not considered as design goals but rather as uncontrollable risks. Neither the user nor the system designer can control them.

2.4 Weaknesses in the Application of SIL

As discussed above, the use of SIL levels does not result in univocal safety. We can identify several weaknesses:

- A SIL level is a system property derived from a prescribed process whereas systems engineering is a mixture of planning, prescribed processes, and architecting/development. As such a SIL level is not a normative property as it is unique for each system.
- SIL objectives are the result of probabilities and estimations, while often no analytical historical data is present to justify the numbers. Also here we see a difference between the automotive domain and the aviation and railway domains. Nevertheless, when new technologies are introduced the process can fail, as was recently demonstrated by the use of Lithium-ion batteries by Boeing [2].
- SIL levels, defined as a system level property, offer little guidance for reusing and selecting components and sub-system modules, whereas engineering is inherently a process whereby components are reused. Hence very little guidance is offered on how to achieve a given SIL level by composing a safe system out of different components. One exception is the ISO-13849 machinery standard and its derivates [9]. In addition, new advanced digital electronics and their interconnecting contacts have not well known reliability figures. They are certainly subject to aging and stress (like analog and mechanical components), but can fail catastrophically in a single clock pulse measured in nanoseconds.
- An increasing part of safety-critical systems contains software. Software as such has no reliability measures, only residual errors while its size and complexity is growing very fast, despite efforts in partitioning and layering approaches that rather hide than address the real complexity. This growth is not matched by an equal increase in controllability or productivity [5]. Transitions to an erroneous state cannot be estimated up front during a SIL determination.
- The SIL level has to be seen as the top level safety requirement of a system. In each application domain different probabilistic goals (in terms of risk reduction) are applied, with an additional distinction between intermittent and continuous operation. Hence cross-domain reuse or certification can be very difficult, because the top level SIL requirements are different, even if part of the certification activities can be reused.
- A major weakness of the SIL is however that it is based on average statistical values. Not only are correct figures very hard or even impossible to obtain, they also depend on several factors such as usage pattern, the operating

environment, and the skills and training of the human operator. Correct statistical values such as the mean value assume a large enough sampling base, which is often not present. Moreover it ignores that disruptive events such as a very unlikely accident can totally change these values. As an example, the Concorde airplane had been deemed as the safest aircraft in the world until one fatally crashed. After the catastrophic event, it "became" almost instantly one of the most unsafe airplanes in the world.

The last observation is crucial. While statistical values and estimations are very good and essential design parameters, very low residual risks can still have a very high probability of happening. This is often known as the Law of Murphy: if anything can happen, eventually it will happen. No lives will be saved by referring to their low statistical probability.

3 The ARRL Criterion

Despite the weaknesses of the SIL criterion, safety standards are still amongst the best of the available engineering standards and practices in use. In addition, those standards contain many hints as of how to address safety risks, though not always in an outspoken way.

As an example, every standard outlines safety pre-conditions. The first one is the presence of a safety culture and recognizes the human factor in engineering. Another essential principle in safety engineering is to avoid any unnecessary complexity. In formal terms: keeping the project's and system's state space under control. A further principle is that quality, read reliability, comes before safety otherwise any safety measure becomes unpredictable. This is also reflected in the requirements for traceability and configuration management.

We focus on the last one to define a novel criterion for achieving safety by composition. Traceability and configuration management are only really possible if the system is developed using principles of orthogonal compensability, hence we need modular architectures whereby components that are (re)used carry a trustworthiness label. In addition, in practice many components are developed independently of the future application domain. The conclusion is clear: we need to start at the component level and define a criterion that gives reusability guidance, namely guidance on how to develop components in a way that allows reusing them without any negative impact on safety at the system level.

3.1 ARRL Levels

In previous sections we have discussed why SIL might not be a suitable criterion. In the attempt to deal with the shortcomings of SIL, we introduce the ARRL (Assured Reliability and Resilience Level). The different ARRL classes are defined in Table 3. They are mainly differentiated in terms of how much assurance they provide in working correctly in the presence of faults, which is mainly an issue for programmable electronics.

Before we elaborate on the benefits and drawbacks of the ARRL criterion, we should mention that there is an implicit assumption about a system's architecture. A system is composed by defining a set of interacting components. This has two important consequences:

1. The components must be designed to prevent the propagation of errors. Therefore the interfaces must be clearly identifiable and designed with a

"guard". These interfaces must also be the only way in which a component can interact with other components.

2. The interaction mechanism (e.g., a network connection) must carry at least the same ARRL credentials as the components it interconnects with. Actually, in many cases, the ARLL level must be higher if one needs to maintain a sufficiently high ARRL level at the level of the (sub)system composed of the components.
3. Hence, it is better to consider the interface as a component on itself, rather than for example assuming an implicit communication between the components.

Note that when a component and its connected interfaces meet the required ARRL level, this is a required pre-condition, not a sufficient condition for the system to meet a given ARRL and SIL level. The application itself developed on top of the component and its interfaces must also be developed to meet the corresponding ARRL level.

Table 3. Definition of ARRL levels

ARRL level	**ARRL Definition**
ARRL-0	The component might work ("use as is"), but there is no assurance. Hence all risks are with the user.
ARRL-1	The component works as tested, but no assurance is provided for the absence of any remaining issues.
ARRL-2	The component works correctly, if no fault occurs. This means that it is guaranteed that the component has no implementation errors, which requires formal evidence as testing can only uncover testable cases. The component still provides ARRL-1 level assurance by testing as also formal evidence does not necessarily provide complete coverage but should uncover all the so-called systematic faults (e.g., a wrong parameter value). In addition, the component can still fail due to random errors, such as an externally induced bit-flip.
ARRL-3	The component inherits all the properties of the ARRL-2 level and in addition is guaranteed to reach a fail-safe or reduced operational mode upon a fault. This requires monitoring support and architectural redundancy at micro or macro level. Formally this means that the fault behaviour is predictable as well as the subsequent state after a fault occurs.
ARRL-4	The component inherits all the properties of the ARRL-3 level and can tolerate one major fault. This corresponds to requiring a fault-tolerant design. This entails that the fault behaviour is predictable and transparent to the external world. Transient faults and most common mode failures are masked out.
ARRL-5	The component inherits all the properties of the ARRL-4 level but is using heterogeneous sub-components to handle residual common mode failures.

By formalizing the ARRL levels, we make a few essential properties normative:

- The components must carry evidence that it meets the requirements, hence the use of the "Assured" qualifier. Without evidence, no verifiable assurance is possible.

- Reliability is used to indicate the quality of the component. A high reliability implies that the MTBF will be high and there is hence not a major issue in using the component.
- Resilience is used to indicate the capability of the component to continue to provide its intended functionality in the presence of faults.
- There is no mentioning of safety or security levels because they are system level properties that also include the application specific functionality.
- The ARRL criterion can be applied in a normative way, independently of the application domain.
- By this formalization we also notice that the majority of the components (software or electronic ones) on the market will only meet ARRL-1 (when tested and a test report is produced). ARRL-2 assumes the use of formal techniques and very few software meets these requirements. From ARRL-3 on, a software component has to include additional functionality that deals with error detection and isolation, and requires a software-hardware co-design. With ARRL-4 the system's architecture is enhanced by explicitly adding redundancy and whereby it is assumed that the faults are independent in each redundant channel. In software, this corresponds to the adoption of design redundancy mechanisms so as to reduce the chance of correlated failures.
- ARRL-5 further requires 3 quasi-independent software and hardware developments because ARRL-4 only covers a subset of the common mode failures. Less visible aspects are for instance common misunderstanding of requirements, translation tool errors, and time dependent faults. The latter require asynchronous operation of the components and diversity using a heterogeneous architecture.

3.2 Rules for Composition

A major advantage of the ARRL criterion is that it allows the definition of simple rules for composing safety-critical systems. We use here an approximate mapping to the different SIL definitions by taking into account the recommended architecture for reaching a certain SIL level.

A system can only reach a certain SIL level if all its components are at least of the same ARRL level. The following side-conditions apply:

- The composition rule defines a pre-condition, not a sufficient condition. Application specific layers must also meet the ARRL criterion.
- ARRL-4 components can be composed out of ARRL-3 components using redundancy. This requires an additional ARRL-4 voter.
- ARRL-3 components can be composed using ARRL-2 components (using at least two whereby the second instance acts as a monitor).
- All interfaces and interactions also need to have the same ARRL level.
- Error propagation is to be prevented. Hence a partitioning architecture (e.g. distributed, concurrent) is a must.
- ARRL-5 requires an assessment of the certification of independent development and, when applied to software components, a certified absence of correlated errors.

3.3 ARRL and SIL as Complementary Criteria

The ARRL level criterion is not a replacement for the SIL level criterion. It is complementary in the same sense that the HARA and FMEA are complementary (Fig. 2). The HARA is applied top-down whereby the system is considered in its environment including the possible interactions with a user or operator. The goal of the HARA is to find the situations whereby a hazard can result in a safety risk. The outcome is essentially a number of safety measures that must be part of the system design without necessarily prescribing how these are to be implemented.

The FMEA takes a complementary approach after an implementation architecture has been selected. FMEA aims at identifying the faults that are likely to result in errors ultimately resulting in a system failure whereby a safety risk can be encountered. Hence the HARA and FMEA meet in the middle confirming their findings. They both result in safety measures that are reflected as safety support in the system architecture and its composing entities.

Although it is still a tentative step because it will require validation in real test cases, by introducing the ARRL criterion we take a first step towards making the safety engineering process more normative and generic. The SIL is a top level requirement decomposed in normal case requirements (ARRL-1 and -2) and fault case requirements (ARRL-3, -4, and -5). From a functional point of view, all ARRL levels provide the same functionality but with different degrees of assurance and hence trustworthiness from the point of view of the user. Concretely, different ARRL do not modify the core functional requirements and specifications of the components.

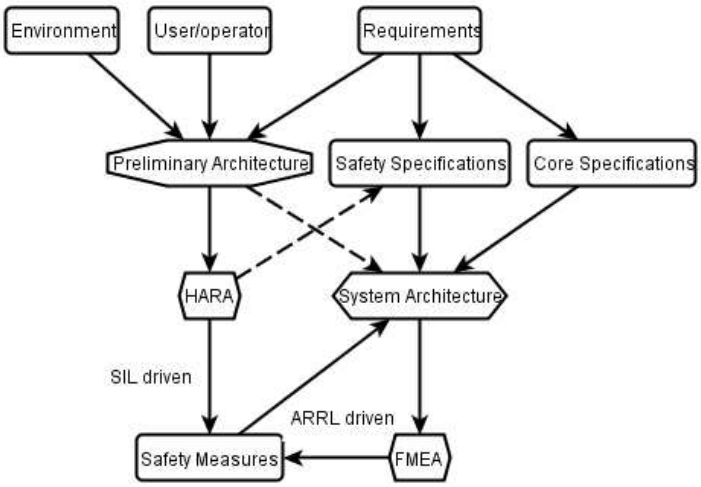

Fig. 2. Simplified view on HARA and FMEA correspondence with SIL and ARRL

The normative ARRL requirements result in additional functional specifications and corresponding functional support that assures that faults do not result in the core functional specifications to be jeopardized. The non-functional specifications will be impacted as well. For example, the additional functionality will require more resources (e.g., memory, energy, and CPU cycles) and is likely to increase the cost price of the system. However, it provides a way to reuse components with lesser efforts from one domain to another in a product family. For example a computer

module (specified for compatible environmental conditions) can be reused between different domains. The same applies to software components. This also requires that the components are more completely specified than it is now often the case.

ARRL level components carry a contract and the evidence that they will meet this contract given a specific set of fault conditions. Note that when using formal methods, each of these ARRL levels also requires more extensive formal models. The higher level ARRL models must include the fault behaviour in conjunction with the normal one, just like invariants are part of the formal models. By defining a composition rule of ARRL components to achieve a certain level of safety, we also define now safety in a quasi-domain independent way, simplifying the safety engineering process. Note however that any safety-critical system still has an application specific part that must be developed to meet the same level of ARRL to reach the required SIL.

4 Related Work

The main stream of related work corresponds to those publications that have compared safety engineering practices and standards in different domains. An overall goal of these works is to analyse the possibility of cross-domain and cross-standard safety assurance and certification. Another goal is to find cross-domain approaches.
Some publications have dealt with the comparison from an abstract perspective. A systematic literature review [11] and a survey on the state of the practice [12] determined the similarities among different application domains in relation to the artefacts used as safety evidence, techniques for structuring and for assessing evidence, and the challenges in evidence provision. Although many commonalities are indicated, the comparisons do not deal with safety standard-specific details, nor with component-related issues.

A common, unified metamodel for safety standards is proposed in [6]. This metamodel includes a concept called "Criticality Level", which corresponds to the risk and safety levels defined in safety standards (SIL, ASIL, DAL, etc.). Although this concept is generic and does not deal with improvements in the application of the levels, an ARRL-based standard could be specified with the metamodel.

Some authors have dealt with the comparison of standard-specific aspects. For example, safety qualification and certification strategies are compared for standards used in the aerospace, automation, automotive, civil aviation, industrial control, nuclear, and railway domains in [3]. The safety and assurance levels are compared and their similarities are discussed in [10].

What distinguishes this paper from the publications above is the proposal of a specific criterion for cross-domain and cross-standard reuse of components, while still meeting the SIL levels.

5 Conclusion

This paper has analysed the concept of Safety Integrity Level (SIL) and put it in a wider perspective of Quality of Service and Trustworthiness. These concepts are more generic and express the top level requirements of a system in the perspective of a prospective user. We have discussed some weaknesses in the SIL concept, mainly its probabilistic system-specific nature, whereas engineering is often based on composition using components or sub-systems.

A novel concept called ARRL (Assured Reliability and Resilience Level) has been introduced, defining a normative criterion for components and their interactions. This criterion focuses on the functional behavior in presence of faults but in a domain-agnostic way. The ARRL criterion, being normative, is a very promising approach to achieve composable safety across different domains and systems in a product family. As future work, we plan to further validate ARRL and apply it in the context of safety-critical applications.

Acknowledgments. The research leading to this paper has received funding from the FP7 programme under the grant agreement n° 289011 (OPENCOSS) and from the Research Council of Norway under the project Certus-SFI.

References

1. Anonymous: Archives of the Aircraft Crashes Record Office. . Bureau d'Archives des Accidents Aéronautiques (2013) http://baaa-acro.com/index.html
2. Anonymous: Dreamliner: Boeing 787 planes grounded on safety fears. BBC News business (17 Jan. 2013) http://www.bbc.co.uk/news/business-21054089
3. Baufreton, P., et al.: Multi-domain comparison of safety standards. In: Proc. of ERTS 2010
4. Anonymous: Railway applications - Communications, signalling and processing systems - Software for railway control and protection systems. CENELEC. Project EN 50128:2011 (2011)
5. de Florio, V., Blondia, C: On the requirements of new software development. International Journal of Business Intelligence and Data Mining, 3(3): 330-349 (2008)
6. de la Vara, J.L., Panesar-Walawege, R.K.: SafetyMet: A Metamodel for Safety Standards. Technical Report, Simula Research Laboratory (2013)
7. Goel, L.R., Tyagi, V.K.: A two unit series system with correlated failures and repairs. Microelectronics Reliability 33(14): 2165-2169 (1993)
8. Anonymous: Functional safety of electrical / electronic / programmable electronic safety-related systems (IEC 61508), IEC (2005)
9. Anonymous: International Standard Road vehicles - Functional safety, ISO/DIS 26262, ISO (2011)
10. Ledinot, E., et al.: A cross-domain comparison of software development assurance standards. In: Proc. of ERTS^2 2012, Toulouse, France (2012)
11. Nair, S., et al.: An Extended Systematic Literature Review on Evidence for Safety Compliance, Technical Report, Simula Research Laboratory (2013)
12. Nair, S., et al.: The State of the Practice on Evidence Management for Compliance with Safety Standards. Technical Report, Simula Research Laboratory (2013)
13. Anonymous: DO-178C - Software Considerations in Airborne Systems and Equipment, RTCA (2012) http://www.verifysoft.com/en_DO-178C.html
14. Anonymous: Global status report on road safety 2013: supporting a decade of action. Technical Report. World Health Organization (2013)