future internet

MDPI

Review

Resilience in the Cyberworld: Definitions, Features and Models

Elisabeth Vogel [1,*], Zoya Dyka [1], Dan Klann [1] and Peter Langendörfer [1,2]

[1] IHP—Leibniz-Institut für Innovative Mikroelektronik, 15236 Frankfurt (Oder), Germany; dyka@ihp-microelectronics.com (Z.D.); klann@ihp-microelectronics.com (D.K.); langendoerfer@ihp-microelectronics.com (P.L.)
[2] Chair of Wireless Systems, Brandenburg University of Technology Cottbus-Senftenberg, 03046 Cottbus, Germany
* Correspondence: vogel@ihp-microelectronics.com

Abstract: Resilience is a feature that is gaining more and more attention in computer science and computer engineering. However, the definition of resilience for the cyber landscape, especially embedded systems, is not yet clear. This paper discusses definitions provided by different authors, on different years and with different application areas the field of computer science/computer engineering. We identify the core statements that are more or less common to the majority of the definitions, and based on this we give a holistic definition using attributes for (cyber-) resilience. In order to pave a way towards resilience engineering, we discuss a theoretical model of the life cycle of a (cyber-) resilient system that consists of key actions presented in the literature. We adapt this model for embedded (cyber-) resilient systems.

Keywords: cyber-resilience; security; redundancy; resilience engineering

Citation: Vogel, E.; Dyka, Z.; Klann, D.; Langendörfer, P. Resilience in the Cyberworld: Definitions, Features and Models. *Future Internet* **2021**, *13*, 293. https://doi.org/10.3390/fi13110293

Academic Editors: Christos Tryfonopoulos, Skiadopoulos Spiros and Arcangelo Castiglione

Received: 15 September 2021
Accepted: 16 November 2021
Published: 19 November 2021

1. Introduction

The cyberlandscape of the 21st century is constantly growing and becoming increasingly complex, covering areas such as telemedicine, autonomous driving, etc. Our societies, as well as individuals, are highly dependent on these systems working correctly 24/7. In order to be able to cope with the increasing complexity and the unprecedented importance of cybersystems, new and innovative methods and technologies have to be applied. The concept of resilience is receiving increasing attention in this respect, which is reflected above all by the steadily growing number of publications on the topic. Figure 1 shows how the number of publications has increased since 2012. The diagram in Figure 1 shows only publications with the keyword *Cyber-Resilience*. Beneath its attention in science, the concept of resilience has already reached industry. US-American streaming provider Netflix is considered a pioneer in the application of resilience in the form of highly redundant infrastructure. However, the principles of resilience are not only found in the hardware components of Netflix. The software architecture also demonstrates the application of various methods to increase resilience. The example of Netflix shows how important resilience becomes with increasing complexity [1].

However, the term "resilience" is used in many ways in IT. In some cases, resilience is described as "extreme reliability" [2] or used as a synonym for fault tolerance [3,4]. In [5] it is described that resilience is fault tolerance with the key attribute robustness. Anderson, in [5], extended the definition of fault tolerance by the property robustness, and called the new definition resilience. In recent publications, however, resilience is defined several times as an independent term [6,7]. In publications [8] and [9], some aspects are added to the concepts already applied in [6]. In [8], the model of [6] is extended. In [9], methods used to achieve cyber-resilience are listed as possible measures against cyberattacks (keyword, IT security).

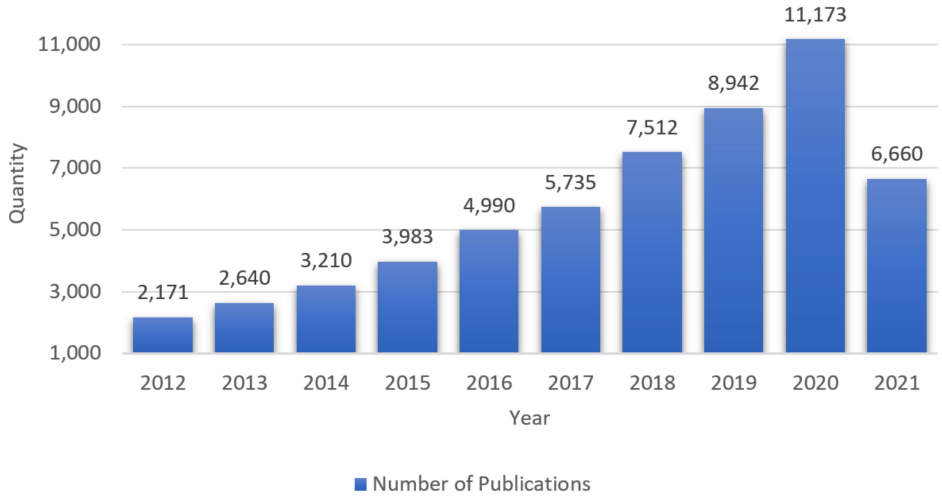

Figure 1. Number of publications with the keyword Cyber-Resilience from 2012 to 2021 (July) [10].

This clearly shows that the definition of resilience in IT as well as in other areas is certainly extensive and varied. While we admit that the term resilience is difficult to grasp, we are also convinced that as a central concept of upcoming new IT systems it needs a clear, commonly accepted definition and metrics, etc. To achieve this, we want to use this publication to provide an overview of definitions of resilience that are already known. For this purpose, only definitions from last 20 years plus one from 1976 were selected and interpreted. The goal here was to provide a reasonable synthesis of previous definitions and interpretations of the term resilience. In addition, we aim to introduce a holistic definition for cyber-resilience that benefits from the experience and knowledge of previous publications.

Throughout this publication, it becomes clear that certain key actions can be extracted that essentially constitute resilience and that can be used to create a model for resilience. Attributes can be assigned to these key actions, which in turn provide methods to implement these key actions. So, fundamental elements for realizing resilient systems are identified and discussed in this article. Based on this insight, two models with key actions are briefly presented in order to use the knowledge gained to design and describe our own model for cyber-resilience in the IT landscape. These models can be used for the later design of cyber-resilient systems.

In the following, the publications that were selected as part of this overview are named and briefly described.

Publications

For the purpose of a consistent presentation, we selected the publications to be as representative as possible. Table 1 shows the publications considered in this publication. This list of publications is of course only a very small selection; this is for a better overview. However, it also shows the different approaches of the authors when defining resilience. Further literature with similar perspectives is also noted in the appropriate places. From the publications given in Table 1, the following information on resilience was extracted (where available): definitions, attributes, models. Furthermore, we discuss our model, which describes the structure of a resilient system. With the help of this model, the development of a resilient system should be facilitated. These approaches are critically analysed and contrasted with our own holistic understanding of resilience. Finally, an example is used to show how approaches of resilience are already being implemented and in which direction the development could go in the future.

Table 1. The columns "title", "author(s)", and "year" show the papers we used as the main sources for this work, the column "further sources" indicates additional publications that use the term resilience in a similar way to the main source in the same row.

No.	Year	Title and Author(s)	Further Sources
1	1976	**A Principle for Resilient Sharing of Distributed Resources** [2] *Peter A. Alsberg; John D. Day*	
2	2007	**Release it! Design and Deploy Production-Ready Software** [11] *Michael T. Nygard*	[12]
3	2008	**From Dependability to Resilience** [13] *Jean-Claude Laprie*	[4,14,15]
4	2011	**Prologue: The scope of resilience engineering** [16] *Erik Hollnagel*	[17,18]
5	2013	**On the Constituent Attributes of Software and Organisational Resilience** [19] *Vincenzo De Florio*	[3,20]
6	2015	**Quantifying coastal system resilience for the US Army Corps of Engineers** [6] *Julie Dean Rosati; Katherine F. Touzinsky; W. Jeff Lillycrop*	[21–24]
7	2016	**What's the Difference between Reliability and Resilience?** [25] *Aaron Clark-Ginsberg*	[26,27]
8	2018	**Systems Security Engineering: Cyber Resiliency Considerations for the Engineering of Trustworthy Secure Systems (NIST Special Publication 800-160, Volume 2)** [7] *Ron Ross; Richard Graubard; Deborah J. Bodeau; Rosalie McQuaid*	[28–32]
9	2020	**Systematic Approach to Cyber Resilience Operationalization in SMEs** [8] *Juan F. Carías, Marcos R. S. Borges, Leire Labaka, Saioa Arrizabalaga, Josune Hernantes*	[33–36]
10	2020	**Foundations for Research in Cyber-Physical System Cyber Resilience using State Estimation** [9] *S. Hopkins, E. Kalaimannan, C. S. John*	[37–42]

Furthermore, we discuss our model, which describes the structure of a resilient system. With the help of this model, the development of a resilient system should be facilitated.

The rest of the paper is structured as follows. Section 2 shows different definitions according to different authors, years and application areas for resilience. In Section 3, the same authors as in Section 2 are considered again, but this time under the aspect of attributes describing resilience. Section 4 describes the model of key actions, which was already briefly mentioned in Section 2. This model of key actions will be reinterpreted in Section 4, according to our ideas.

In Section 5 we show current applications of resilience and the transferability to embedded systems. In addition, we discuss how to model and implement resilience.

2. Definitions

In the literature of recent years, there have been many definitions of resilience, some of which differ considerably. As described in [3], the content of the definitions strongly depends on the respective fields of application.

Resilience is derived from the Latin *resilire*, and can be translated as "bouncing back" or "bouncing off". In essence, the term is used to describe a particular form of resistance.

How the term resilience is used in different other disciplines (material science, engineering, psychology, ecology) is described in [43].

Additionally, in computer science, the term resilience has been defined several times from different points of view. As described in [3] for example, resilience is often used as a synonym for fault tolerance. However, recent publications show that this approach has been replaced by the view that resilience is much more than fault tolerance (see [7]).

In the following, we introduce the different definitions of resilience that we found in the literature, to provide a common understanding of resilience in general and for the

rest of this paper in particular. Definitions are taken, if available, from the publications in Table 1.

One of the first definitions was presented 1976 in [2], and describes the concept of resilience as follows:

> *"He (remark: the user) should be able to assume that the system will make a "best-effort" to continue service in the event that perfect service cannot be supported; and that the system will not fall apart when he does something he is not supposed to".*

In addition, [2] mentions attributes that constitute resilience as part of its definition. The attributes are the following: **error detection**, **reliability**, **development capability** and **protection against misuse** in the sense that the misuse of a system by individual users has only negligible effects on other users. According to Alsberg [2], these four attributes of a resilient system can be summarized as the attempt to describe extreme reliability and serviceability. In summary, a partial failure of a system should not have any effect on an individual user, so the system can be assumed to be highly reliable. Should nevertheless a partial failure or a defect occur, the best possible continuation of the services provided should be guaranteed. In extreme cases, this continuation can also be achieved by performing graceful degradation of services.

The approach of continuing a service of a system even under transient effects, permanent load or failures is also described in [11]:

> *"A resilient system keeps processing transactions, even when there are transient impulses, persistent stresses, or component failures disrupting normal processing. This is what most people mean when they just say stability. It's not just that your individual servers or applications stay up and running but rather that the user can still get work done".*

According to Nygard [11], a system must remain stable in case of tensions or stress situations or failures. As a consequence, involved (sub-) systems, or possibly also users, can still continue their work. The system must also be able to continue fulfilling at least its rudimentary functions despite any restrictions that may occur. The scope of these rudimentary functions may have been defined as part of the Risk Management, for example. Risk management also shows at what level of functional loss the entire system function according to its specifications can no longer be provided.

The paper by Laprie [13] proposes two definitions of resilience. The first definition is as follows:

> *"The persistence of service delivery that can justifiably be trusted, when facing changes".*

According to Laprie, this definition corresponds in principle to the original definition of reliability. In a second definition, Laprie offers an alternative which provides a more detailed description:

> *"The persistence of the avoidance of failures that are unacceptably frequent to severe, when facing changes".*

In [13], resilience is described as the persistence of service delivery when changes occur that have system-wide effects. These changes can be functional, environmental or technological. In addition, changes can either be planned (for example: initialized by an update), the timing of their occurrence can be unpredictable, or they can be completely unexpected. The duration of changes is also taken into account: short-term, medium-term, long-term. This refers to the duration of the impact of the change on the system or a subsystem.

In the collection of papers from 1985 [5], robustness was already mentioned in connection with resilience. About 30 years later, in [19], this connection is concretized. Resilience is defined as the trustworthiness of a software system to adapt to adverse conditions. The software system should accept and tolerate the consequences of failures, attacks and changes inside and outside the system boundaries. This is defined as an approach for robustness:

> *"Software resilience refers to the robustness of the software infrastructure and may be defined as the trustworthiness of a software system to adapt itself so as to absorb and tolerate the consequences of failures, attacks, and changes within and without the system boundaries".*

The definition of resilience was further specified in [19]. Florio [19] refers to the definition already mentioned in [3] and another definition in [44]. This definition states that resilience can be characterized as a measure of the persistence of both functional and non-functional features of a system under certain and unpredictable disturbances. After analyzing these two definitions, according to Florio, resilience is the ability to act and balance between two main behaviors:

(1) Continuous readjustment with the aim of improving the fit of the systems' environment, and compensating for both foreseeable and unforeseeable changes in the system environment.
(2) Ensure that the said changes and adjustments from 1) do not affect the identity of the system. This means that its specific and distinctive functional and non-functional features should not be affected.

Ref. [6] deals with the management of water resources, and was written from the perspective of the USACE (US Army Corps of Engineers). According to Rosati [6], resilience is a cycle consisting of anticipation, resistance, recovery and adaptation. Anticipation is the starting point of the cycle, while adaptation marks the end. The cycle is started by the occurrence of an event that affects the system in some way. This event is called a disruption. Specifically, Rosati defines resilience (in this case coastal resilience) as follows:

> *"(Coastal) resilience is defined as the ability of a system to prepare, resist, recover, and adapt to disturbances in order to achieve successful functioning through time".*

A disturbance occurs here as an effect of a hazard on the infrastructure, system, etc. A hazard is an environmental or adverse anthropogenic condition.

The article by Clark-Ginsberg published in 2016 [25] defines the ability of a system to reduce the extent and the duration of disruptions as resilience. Disruptive events are not always predictable, but when they occur they are supposed to lead to a learning and adaptation effect of the system. Adaptation is crucial when it comes to realizing resilience against cyberaccidents, since the cyberlandscape is developing very rapidly. Clark-Ginsberg says in his article that errors must be detected and understood. It must be possible for the system to adapt to the errors or the error situation and a fast recovery must be guaranteed. The system must recover quickly after the occurrence of an error. If this is not possible, the error and the resulting faulty system environment must be dealt with appropriately.

The U.S. National Institute of Standards and Technology (NIST) [7] coins the term cyber-resilience to clearly distinguish its approach from the general definitions of resilience. Cyber-resilience is the following property:

> *"Cyber Resilience is defined in this publication as the ability to anticipate, withstand, recover from, and adapt to adverse conditions, stresses, attacks, or compromises on systems that include cyber resources".*

According to NIST, the definition of cyber-resilience refers specifically to all entities that contain cyber-resources. A cyber-resource is an information resource that creates, stores, processes, manages, transmits, or disposes information in electronic form and that can be accessed over a network or by network methods. The definition of NIST can therefore be applied to a system, a mechanism, a component or a system element, a common service, an infrastructure or a system of systems, etc.

Publications [8] and [9] do not present completely new definitions for cyber-resilience. These publications refer to already-known sources such as NIST [7] for a definition. Publication [8] by Carías extends the previously mentioned circuit (Rosati [6]) to include the ability of a system to detect threats. Carías calls this new cycle the cyber-resilience life cycle.

This approach is described in detail in Section 4. Publication [9] by Hopkins considers cyber-resilience as an effective countermeasure against cyberattacks or cyberthreats. In [45], the link between security and cyber-resilience is also discussed.

The publications selected here show that the type and scope of the definitions of resilience depend very much on the respective (informatics) application area. However, some key actions can be filtered out, which appear at least partially in all the publications considered here: Anticipating, resisting, recovering, detecting and adapting (to threats of any kind). In some publications, such as [6,7,46] these key actions are even mentioned explicitly. Thus, we filtered these five key actions out of the definitions. Table 2 shows which key action is mentioned in which of the publications considered here. In addition, Table 2 also clearly indicates that the definition of resilience has been becoming more and more complex over the last decades; the number of key action mentions per publication has increased from two to all five in the most recent publications. Each of the five key actions can be assigned different attributes and behaviors. They are described in the following section.

Table 2. Sources mentioning key actions.

No.	Publication	Key action
1	Alsberg, 1976	
2	Nygard, 2007	
3	Laprie, 2008	
4	Hollnagel, 2011	
5	Florio, 2013	
6	Rosati, 2015	
7	Clark-Ginsberg, 2016	
8	NIST, 2018	
9	Carías, 2020	
10	Hopkins, 2020	

Section 4 specifically shows how, for example, [6] builds a model from these key actions to achieve an illustration for cyber-resilience. Building on the model of [6] and some others, this publication has designed its own model of key actions for the cyberlandscape.

3. Key Actions and Attributes

Section 2 introduced the key actions anticipation, resistance, recovery, detecting and adaptation. These five key actions can be defined as follows. Comparable definitions are listed in [6,8,9,33], for example:

- **Anticipation:** Anticipation is a process that enables the system to prepare for a disruption or an attack that may occur.
- **Resistance:** Resistance is the ability to withstand the effects of a disruption or an attack and maintain a certain level of functionality.
- **Recovery:** The system must be able to recover the lost functionality.
- **Adaptation:** The system must be able to put itself in a state by responding more efficiently to the disturbance in the future. This essentially means that the system will lose less functionality in the future and the recovery time will also be less.
- **Detection:** Detection means that a system can detect a disturbance or an attack in order to initiate appropriate countermeasures.

Each key action comprises several attributes (In the publications presented here, the terms attribute, feature and measure were used synonymously. For the sake of clarity, only the term attribute will be used in this article, representing Feature and Measure). These attributes can be derived directly from the definitions or were explicitly mentioned in the publications. The attached descriptions of the attributes can be found in this way in [11,47], for example:

- **Robustness:** Still function reliably under adverse conditions.
- **Reliability:** Continuity of service.
- **Availability:** Readiness for usage.
- **Evolvability:** Ability to accommodate changes.
- **Adaptability:** Ability to anticipate to changes.
- **Security:** Crime Prevention.
- **Safety:** Accident prevention.
- **Assessability:** Check for correctness of data (plausibility check).
- **Integrity:** Nonoccurrence of incorrect system alterations.

Ref. [2] describes the four main attributes of a resilient service. First, a resilient service must be able to detect and correct errors. Further, the resilient service must be so robust and reliable that a user expects the service not to fail. If the service is capable of always detecting n errors and recovering from those errors, the $n + 1$ error is not catastrophic. This only applies under the condition that the system offers perfect detection and recovery of n errors. The resilient service is therefore able to anticipate the $(n + 1)th$ error in such a way that its negative consequences for the service can be minimized. This corresponds to a simple definition of evolvability. As a fourth key attribute, Alsberg [2] cites the ability of a resilient service to tolerate abuse by a single user in such a way that this abuse has negligible impact on the other users of the service. Alsberg does not specify misuse, but if a malicious and intentional action is assumed, then this misuse protection corresponds to the security feature of availability. Alsberg summarizes the following features: robustness, reliability, evolvability and security.

Ref. [11] claims that robustness under all conditions is the most important property of a resilient system. According to Nygard [11], this robustness is directly related to the reliability of a resilient system. This connection is obvious, since a system that is not stable cannot be reliable either.

This understanding of robustness and reliability is also illustrated in [13]. Assessability is also an important property, because a resilient system must be able to validate the correctness or plausibility of sensor data, for example. Laprie [13] also mentions diversity as another important basic property. Diversity can be understood here as a basic idea of redundancy, because according to Laprie, diversity in a system (of hardware components, for example) should prevent the occurrence of single point of failure.

In [16], it is also described that reliability is a key feature of resilience. The ability to detect a fault before it occurs is also essential. However, this only refers to faults that can be anticipated on the basis of existing information. A resilient system must be able to minimize the negative effects of a disturbance by anticipating it. This is achieved by constantly updating information about the disturbances that have already occurred and been treated. This process can be understood as the ability to evolve.

According to [19], the following attributes are essential for a resilient system: reliability, evolvability and integrity. Reliability and evolvability are related to resilience, as described in the previous definitions. Integrity, according to Florio [19], means that a resilient system does not lose its intention after adaptation or application of changes regarding a failure. This refers mainly to its functional and nonfunctional characteristics.

In the publications [6,7,25], the following abilities: anticipate, resist, recover and adapt, are directly mentioned as the four basic attributes, or, as in NIST, the four basic goals of resilience. Publications [8] and [9] extend this approach to include the ability to detect a threat. Detection is important for the self-assessment of a system, because if a threat cannot be detected, the system cannot initiate appropriate countermeasures

Discussion of the Relationships between Key Actions and Attributes

The attributes are directly related to the key actions. This relationship becomes important when it comes to the concrete design or practical modeling of cyber-resilient systems. In order to design a system to be cyber-resilient or to make an existing system cyber-resilient, it must first be examined how, for example, the ability to resist can be achieved or improved. At this point, attributes must be considered. For example, to make a system resistant, methods can be used that increase robustness. Methods of reliability and security are also helpful here. It is to be considered here that the use of several different methods can lead to dependencies, which must be considered. For example, a method that makes a system more adaptable could at the same time reduce its security. Table 3 shows the mapping of the mentioned attributes to the key actions according to our understanding.

Table 3. Mapping of the mentioned attributes to the key actions.

	Anticipation	Resistance	Recovery	Adaptation	Detection
Robustness		X	X		
Reliability		X	X		
Adaptability		X	X	X	
Evolvability				X	
Security	X	X			X
Safety	X	X			X
Assessability					X
Integrity	X	X			

The various key actions have interdependencies, which need to be taken into account when building a theoretical model of cyber-resilience. In the following section, such models [6,8,9] are briefly presented.

4. Model of Key Actions

The key actions of resilient systems were introduced in Section 2. In this section their dynamic relationships are shown and explained.

The key actions are mentioned completely or partially in many publications, but Rosati in [6] establishes a direct connection between them in the form of a cycle as shown in Figure 2. Rosati's paper is not about implementing an IT system, but about the ability of different departments, components and participants to respond when a disaster occurs. This responsiveness must be constantly improved in the sense of Rosati's publication in order to keep (permanent) damage and loss of life as low as possible. The concept presented by Rosati is intended to implement a system-wide approach that will support the challenges of managing the United States' water resource infrastructure. However, this abstract concept can be transferred to IT systems.

The cycle shown in Figure 2 is started when a disturbance occurs, and is considered successful once it has been completed for a disturbance. A learning effect is considered to have occurred when there is a measurable improvement if the same fault occurs again in the future.

A disturbance is considered to be an adversity that has negative effects on the system. Negative effects are consequences that interfere with the intention of the system, i.e., in a certain way with its task. In the worst case, the compromised system can no longer fulfill its tasks. Disturbances can be caused by malicious, nonmalicious, anthropogenic (influenced or caused by humans), nonanthropogenic, internal and external influences.

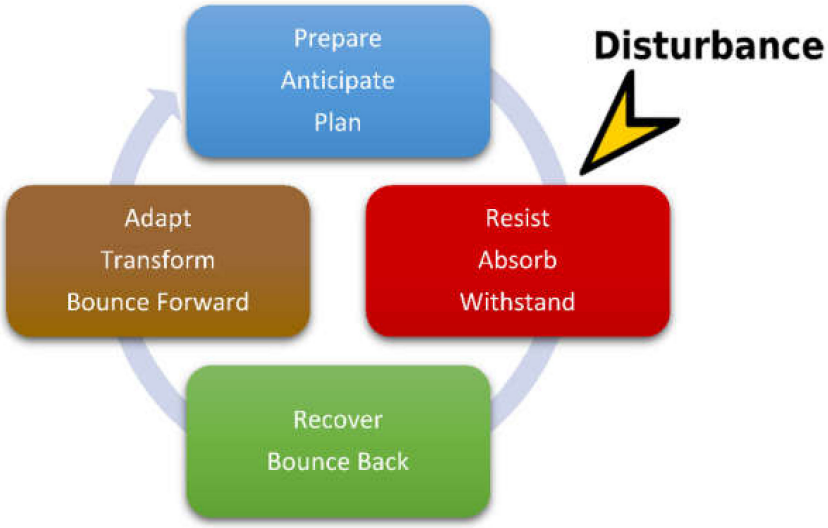

Figure 2. Presentation of key actions in a cycle (image based on [6]).

The four key actions shown in Figure 2 and the resulting cycle are to be understood according to Rosati [6] as follows:

1. **Prepare/Anticipate/Plan:** This key action includes a natural process, or under certain circumstances, an anthropogenic activity, with the aim of preparing the system for a disturbance.
2. **Resist/Absorb/Withstand:** This is the ability to withstand a disturbance while maintaining a certain level of functionality.
3. **Recover/Bounce Back:** The lost functionality must be restored. If it is not possible to maintain functionality, the system shall be able to return to its original state.
4. **Adapt/Transform/Bounce Forward:** This ability to adapt involves putting a system into a state that is better able to withstand or recover from disruption. Ideally, this adaptation leads to reduced loss of functionality and a shorter recovery time. Figure 3 shows the process of increasing resilience schematically. However, the process of adaptation only occurs when the cycle has been completed and applies only to this type of disturbance.

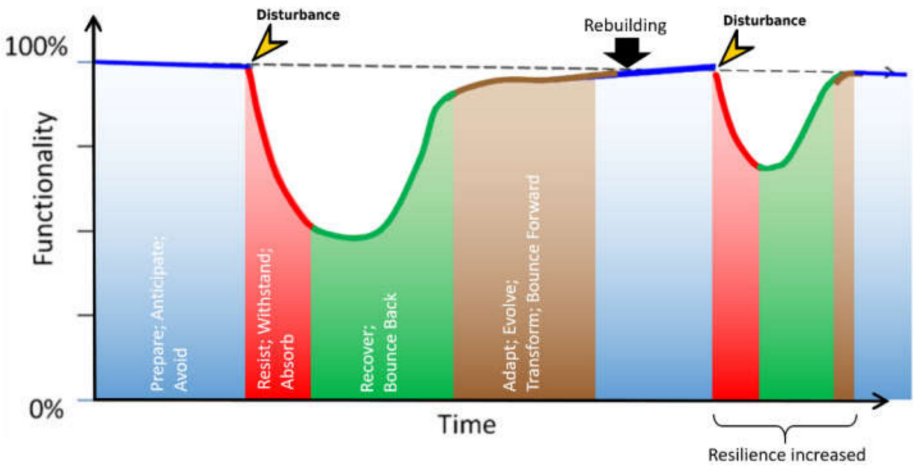

Figure 3. Schematic representation of the learning effect. A system is influenced by a disturbance and loses functionality. This loss must be resisted. Through the process of restoration and adaptation, a similar disruption will trigger a less severe loss of functionality in the future and the time to restore the system is shortened (image based on: [6]).

In Carías [8], the principle of key actions was extended to include the ability to detect a threat. According to Carías [8], a system must first be able to detect a threat in order to respond appropriately to it in the next step. Figure 4 shows the cyber-resilience lifecycle according to Carías.

Figure 4. Cyber-resilience lifecycle [8].

5. Discussion

In this discussion we present our idea of extending the concept of key actions to IT systems and critical infrastructure systems.

In contrast to the models using four or five key actions as described in the literature (see Figures 2 and 4), we propose the use of the following five key actions and/or attributes: anticipation, error analysis, new (learned) methods of/for resilience, robustness and resistance, recovery and adaptation. It is noticeable that our model includes a combination of key actions and attributes. The reason for this is that methods of reliability, availability, security and so on (key action: resistance) must be permanently active from our point of view.

In addition, we consider a continuous "resilience monitoring", where in the literature this is discrete, i.e., assessments are made at specific points in time. The process of increasing resilience can be interpreted as the capacity for evolution. The system "learns" how to better deal with a disturbance that has already occurred at least once. This process is called evolvability.

The term "evolvability" originally comes from evolutionary biology and describes the ability of a living organism to bring about a change in its characteristics (attributes) by changing its genes with the aim of improving its (survival) abilities.

Two approaches are therefore important for evolvability, which build on each other:

1. Modification of genes
2. Changes in attributes resulting from 1.

The relation between genes and attributes can be transferred to a resilient system as follows: genes contain, among other things, the basic information for the evolution of the attributes of a living organism. The key actions represent the "genes" of resilience and the resilience itself has a multitude of attributes (reliability, safety, integrity, etc.). By modifying or improving the key actions (according to Figure 2) the properties of a resilient system can be improved.

The model of the four key actions from [6] is very well-suited for the purpose described in [6].

As shown in Figure 4, Carías extends Rosati's model of key actions [6] to include the ability to detect a threat. From our point of view, however, the sole detection of a threat is not sufficient. For the IT area, we believe that modifications of the model are necessary. First, another essential key action must be added: **error analysis**. Error analysis includes

both the detection and the understanding of the detected error. The reason for this is that the system must "understand" what the error is in order to react optimally or at least appropriately within the scope of its possibilities.

Furthermore, from our point of view, the key action resistance must be considered differently for a (distributed) IT system. An IT system must be able to continuously resist current and future disturbances. A disturbance does not necessarily have to be an event that immediately has negative consequences for the system. A disturbance can also be an attack with the aim of stealing information (e.g., private keys). Such actions, often carried out as side channel attacks [48,49] do not cause any direct, immediately visible damage. However, if the extraction of the private key is successful, the system is considered broken from a security perspective and can no longer be described as resilient.

This eliminates the key action resistance from the cycle. In our model, resistance is divided into permanent and newly learned methods of resistance. This key action is continuously active, whereby "newly learned" methods of resistance are added to the permanent methods of resistance after the disturbance has been eliminated. Figure 5 shows our model of the key actions [45].

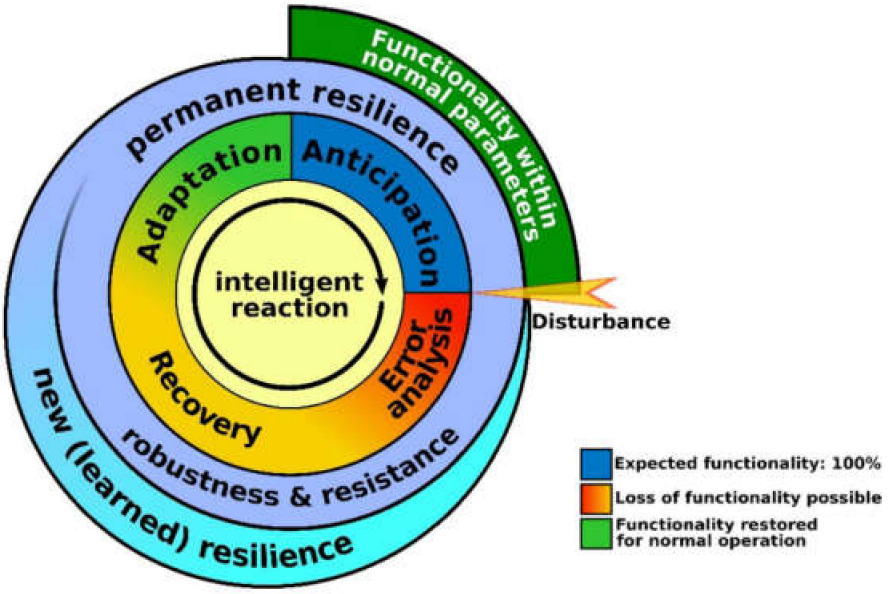

Figure 5. Schematic representation of the key actions anticipation, resilience (permanent methods and new (learned) methods), recovery and adaptation, and the added key action error analysis (source: [44].)

The components of our model are described below. Please keep in mind that an IT system is under constant change, i.e., its state concerning the key actions is also constantly changing. The boundaries between the key actions are not strict. There is hardly a rapid transition between the states. Thus, the individual methods of, e.g., error analysis or recovery, must also be considered. They interact with each other and partly also build on each other.

1. Anticipation

Anticipation, or the ability to anticipate, is generally understood as the capability to predict future conditions, actions or events, taking into account only information already available. Anticipation is executed while the functionality of the system is within normal and expected parameters.

The system should be able to anticipate possible disturbances from already known and/or collected information. This "previous knowledge" empowers the system to minimize the negative consequences of an anticipated disturbance in advance, or even to

prevent the disturbance completely. Whether or not it is possible to prevent a disturbance naturally depends on the type of disturbance.

2. Error analysis

The error analysis basically covers four points: Error detection, error localization, error cause and error type determination. This key action is added to the four formerly used ones, especially when disturbances occur which could not be anticipated in this form because no information was available (yet). In this way, error analysis correlates in a special way with anticipation. If a disturbance passes through the cycle of key actions several times, this disturbance can be better and better anticipated later on, which supports the process of evolvability considerably and shortens the time span for error analysis. This can be traced back to the stored information about the disturbance from previous runs. How well a disturbance can be anticipated depends on the type and complexity of the disturbance.

For error analysis, it is essential that the disturbance has been detected, understood, localized, and possibly even predicted by the system. Additionally, the cause of the disturbance can be helpful. However, the cause often cannot be identified or eliminated.

3. New (learned) methods of/for resilience

This key action includes methods to resist, which are not permanently active. There is a pool of methods that are known to the system but inactive. If a disturbance occurs that is unknown to the system, i.e., could not be anticipated, additional methods to resist become active. If this is the case, the required method is added to the permanent resilience methods (see point 4) after the disturbance has been dealt with.

A corresponding algorithm decides, on the basis of various parameters (type of disturbance, current operating environment, dangerousness of the disturbance, probability of the reoccurrence of the disturbance . . .), whether and how long a method is added to the permanent resilience methods. It also decides when a method can become inactive again, e.g., to save memory. Such an algorithm can be realized with methods of artificial intelligence.

4. Robustness and resistance

Robustness and resistance is the comprehensive key action that is always active. In general, this key action includes the ability to resist the negative consequences of a known disturbance. When a disturbance occurs, a loss of functionality must be expected. This loss of functionality as a negative consequence of a disturbance must be counteracted by methods of permanent resilience. This means, for example, the prevention of data loss or the forwarding of faulty data.

5. Recovery

During the recovery process, the disturbance that has occurred must be remedied as far as possible. It should be noted that the correction of a disturbance is not a process that necessarily only has to take place during recovery. Rather, the correction of a disturbance and the elimination of the negative effects of the disturbance is a continuous process that can also be started during the execution of resistance methods.

However, the process of eliminating a disturbance must be definitely completed with the completion of the recovery. Any lost functionality will, if possible, be reintegrated into the system according to the possibilities and mechanisms used for recovery. Irreparably damaged hardware, for example, can of course no longer be used, but a message could be sent to a maintenance team. This maintenance team can replace the hardware. It should be possible to replace hardware while the system is running.

6. Adaptation

According to Figure 3, Rosati [6] indicates that during the adaptation phase, the system is still in a state in which full functionality has not yet been restored. We believe that for effective adaptation, the system should be in a state in which it is fully functional.

Adaptation essentially means that the system puts itself by self-modification into a new state, in that it can react more efficiently to this type of disturbance in the future. This means less loss of functionality and/or a shorter recovery time.

However, the process of adaptation is by no means trivial. The system has to take into account previous adaptations to other disturbances. These must not be significantly negatively affected by the new phase of adaptation. Furthermore, the adaptation of the system should be checked beforehand. This check can be made, for example, in a backup of the system.

According to this modification of the model of key actions, the schematic representation of the learning effect must be adapted. Figure 6, shows the new schematic representation.

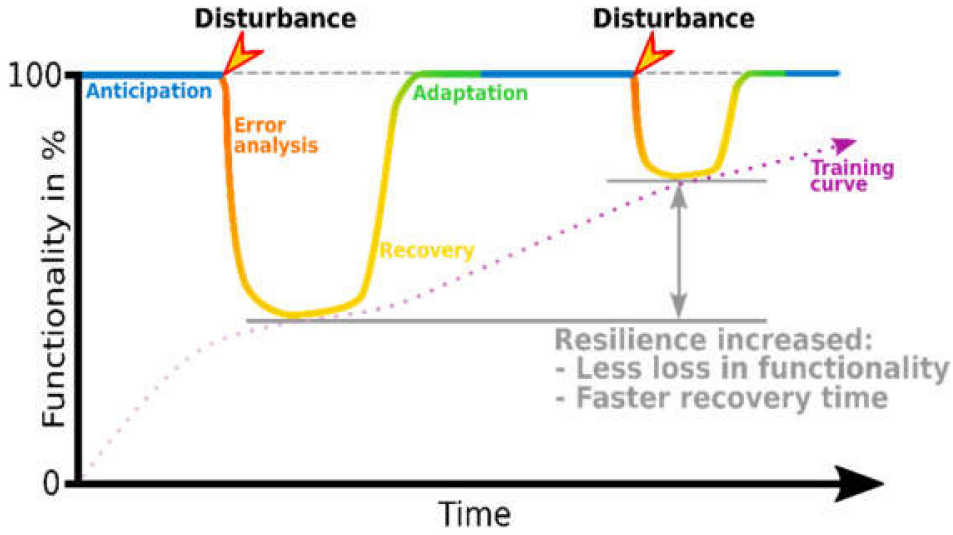

Figure 6. Schematic representation of the learning effect with the added key action error analysis. The two key actions of resistance are to be regarded as decoupled and cover the other four key actions of anticipation, error analysis, recovery and adaptation completely.

Based on the key actions and their relationships to each other, a resilient system can be designed. For this purpose, in addition to the consideration of the key actions, information from the following areas, among others, must be taken into account:

- Mechanisms for error detection
- Mechanisms for attack detection (e.g., IDS (Intrusion Detection System))
- Mechanisms for error correction
- Mechanisms for fault localization
- Types of errors
- Mechanisms for pattern recognition

These mechanisms are to be applied according to the requirements of the system and the available options. Particular attention is required in the area of error detection. As a rule, errors are only detected after they have caused (negative) consequences or accidents. Thus, it is obvious that the cycle of key actions consists of partially interlocking key actions that are directly or indirectly interdependent. For example, at the moment when an error has a negative impact on the system, the system must be able to resist this impact.

Thus, in order to achieve resilience, CPS (oS) (Cyber-Physical System (of Systems)) need to be capable to handle different adverse conditions, of which some might be expected while others are not.

CPS (oS) must be robust to specified disturbances. This includes specified manipulation and attacks. Specified disturbances are disturbances that could be expected during the development phase. This robustness can be realized, e.g., with special materials, which

guarantee the robustness of the design in the specified working area (the expected working conditions). These can be, for example, wires made of metal that have a higher melting point and can therefore meet the set requirements.

In addition, this robustness can be achieved by using redundancy. Redundancy is used according to the specified working conditions to ensure fault tolerance.

CPS (oS) must also be resistant to unspecified disturbances (at least partially). Unspecified disturbances are disturbances that had to be expected but for which no reaction scenario was defined by the system. It is essential that unspecified disturbances can be detected, minimized, predicted or even avoided at an early stage if they occur repeatedly. Unspecified disturbances can be triggered by the following causes, for example:

- Excessive deviation of the physical parameters of the environment from the specified working range.
- Too frequent (even not-strong) deviations of the physical parameters of the environment from the specified working range.
- Too short a reaction time.
- (dynamic) Changes in the specified working or analysis range (e.g., increased operating voltage).
- Individualized work or analysis areas.

In short, we define resilience for CPS (oS) as follows:

> *"A CPS (oS) is resilient if it has the ability to react to specified and unspecified disturbances in a way that preserves its function and reacts quickly. This reaction includes the early detection, minimization, prediction or even avoidance of disturbances. In addition it needs to have the capability to anticipate future challenges and to prepare itself for those".*

According to our definition, we discuss in the following four different types of systems, i.e., primitive system without error handling (1), fault-tolerant system (2), resilient system (3) and total-resilient system (4) shown in Figure 7.

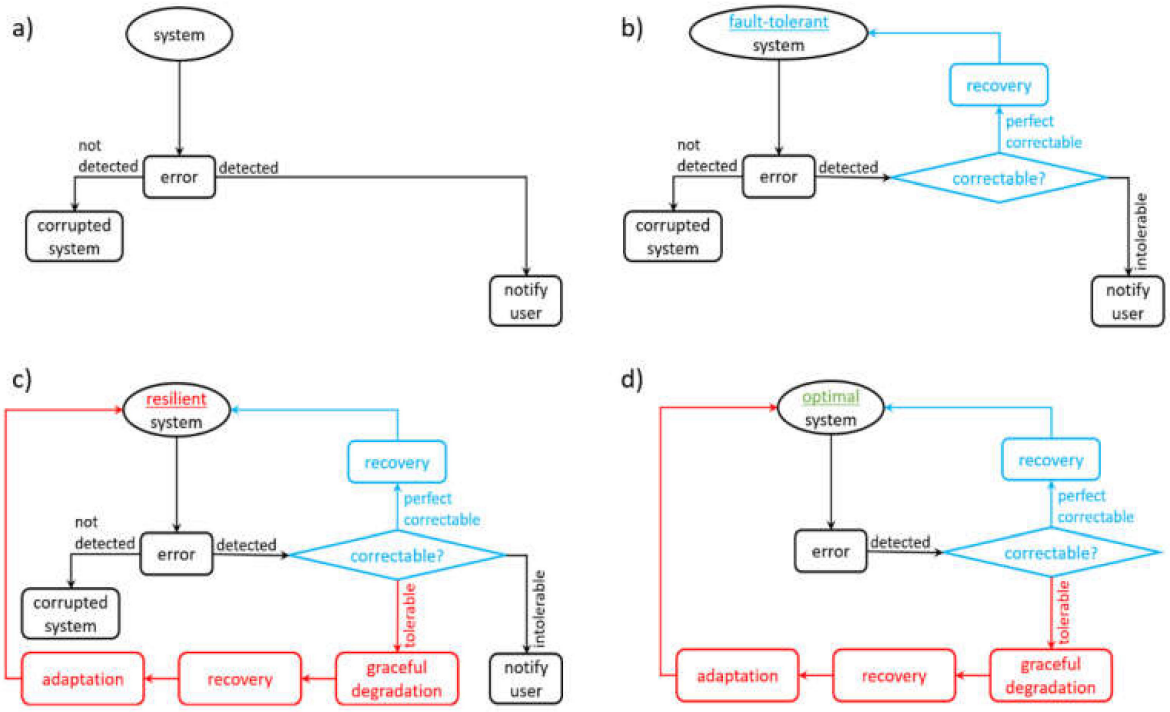

Figure 7. Schematic representation of four different systems: (**a**) primitive system without error handling, (**b**) fault-tolerant system, (**c**) resilient system, (**d**) optimal system.

A system such as the one shown in Figure 7a without any form of error detection, should not be in use today, especially not in the area of critical infrastructure. This system is not able to deal with a fault, understand it or resist its negative effects. As an example, a memory without error correction can be mentioned here. Figure 7b shows a fault-tolerant system. These systems are currently the ones most commonly used. If this system detects an error, this error can often be corrected. If the error cannot be corrected, the user is informed. If an error is not detected, e.g., because it has no direct effect on the functionality of the system, it is a corrupt system that can, among other things, supply incorrect data. As an example, a memory with error correction can be mentioned here.

Figure 7c shows our idea of a resilient system. This idea of a resilient system was developed using our model of key actions. The probability that an error was not detected (corrupt system) or that an error was detected but cannot be handled (inform users) is given but should be very low. As an example, a network with adaptive routing can be mentioned here. Figure 7d shows an optimal system, i.e., a system that can cope with all adverse circumstances. Such a system acts autonomously, learns independently, solves complex tasks, can react to unforeseen events and does not require any additional maintenance or action by a human. Such an optimal system cannot exist comprehensively within the current state of the art. It is in our view, the ultimate target system when designing resilient systems.

In the current development process for highly reliable systems in critical infrastructure, it makes sense to develop increasingly resilient systems and to make fault-tolerant systems resilient. However, resilience is an extremely dynamic concept, as the many different definitions show. Ultimately, however, resilience is not only the ability to deal with errors, but also the realization that errors will definitely happen. It is irrelevant whether these errors are symptoms of a malicious attack, environmental influences or wear and tear. In order to effectively implement resilience, a holistic understanding of the system design and the threat situation is necessary. Additionally, such a system architecture must offer possibilities for expansion (during normal operation), since resilience is subject to constant change due to its dynamics.

6. Conclusions and Future Work

Cyber-resilience encompasses more than security and reliability. Cyber-resilience also deals with the ability of a system to make autonomous decisions according to the situation in which the system is located. These autonomous decisions can be modeled using artificial intelligence methods. However, these methods of artificial intelligence must also satisfy the conditions of security and reliability.

Currently, Netflix is the most well-known example of a resilient system. Netflix's infrastructure and applications use a high degree of redundancy to implement the idea of cyber-resilience in the best possible way. In addition to redundancy, Netflix uses other mechanisms to ensure that high availability is maintained. These include extensive empirical checks of the resilience (the checks are performed live and during normal operation), rapid isolation of errors, the ability to quickly perform fallback, rollback and failover, and constant logging and monitoring of all activities in the system. All these mechanisms enable Netflix to guarantee almost continuous availability to users and to react to unexpected (negative) events [1].

Redundancy is a very powerful tool to achieve cyber-resilience, but it is not universally applicable. The level of redundancy used by Netflix to ensure cyber-resilience is unthinkable for embedded systems, for example. Redundancy requires a lot of physical space, especially in the area of hardware, but embedded systems cannot be extended at will. This means that one has to work with the given form factor and cost limitations.

At the same time, however, it must also be determined what cyber-resilience means for an embedded system. The objectives are the same as those of Netflix, but the resources are limited. The embedded system must also be capable of self-observation (logging and monitoring), and it must be able to predict and detect (negative) changes/events in order

to react. The first big challenge is the (correct and complete) detection of a negative change, because if a system is not able to do so, it cannot take appropriate countermeasures.

If negative events were detected correctly, the system must react to these events. The way the system reacts to such an event is another major challenge. It is not sufficient to say that a system must always react to error X with countermeasure Y. Embedded systems are highly complex, and components are sometimes highly interdependent. The system must react in a way that the negative event can be countered, and at the same time own functionalities are not or only temporarily impaired. Additionally, (sensor) data must not be corrupted or lost. In addition, various negative influences can occur almost simultaneously. The different countermeasures must not hinder or even prevent each other under any circumstances.

The third major challenge is the optimal selection of mechanisms that can be used preventively against negative events. Redundancy would be one such mechanism, but it is only of limited use in embedded systems. A further mechanism would be the isolation of different components of an embedded system, so that negative events from which errors/disruptions arise are limited to one component. In this way, cascading to other components can be prevented. Of course, a complete isolation is not possible, but there are mechanisms that are summarized under the term "loose coupling", that help to keep the interrelationships as small as possible.

So, there is a very wide range of different methods that can be used to achieve cyber-resilience. Especially for embedded systems, it has to be planned exactly which methods should be used. The reasons for this are, among others, the limited space, the limited storage and computing capacity (also with regard to the use of software solutions for cyber-resilience) as well as the available financial means. In addition, the location and the degree of criticality of the system (critical infrastructure) play an important role.

The great challenge of cyber-resilience is the planning and development of systems that fulfill selected aspects of security and reliability. In addition, a system must be able to make intelligent decisions in order to defend itself efficiently against negative effects.

Future Work

One approach would be to develop a design kit for cyber-resilient systems. To develop such a design kit, it is first necessary to collect and classify different methods for achieving cyber-resilience. A classification only makes sense if the methods have been proven to have a positive effect on a system. Theoretical methods must first be tested.

The classification of the methods could then be as follows: effort of implementation, type and extent of effect on the system, possibilities of implementation, etc. The next step is to examine what dependencies exist between these methods. This refers to how these methods influence each other within a system. This allows negative dependencies to be taken into account. This analysis of the dependencies can be implemented, e.g., with the help of artificial intelligence.

The idea would be a semiautomatic design kit that combines different methods for cyber-resilience and can select and combine them with the help of artificial intelligence in a way that is suitable for the system. The result is a theoretical model that can be used for the practical development of a cyber-resilient system.

Author Contributions: Conceptualization, E.V. and P.L.; methodology, E.V., Z.D. and P.L.; validation, Z.D., D.K. and P.L.; formal analysis, E.V. and P.L.; writing—original draft preparation, E.V.; writing—review and editing, Z.D., D.K. and P.L.; visualization, E.V. and Z.D.; supervision, P.L. All authors have read and agreed to the published version of the manuscript.

Funding: The publication of this article was funded by the Open Access Fund of the Leibniz Association.

Data Availability Statement: Not applicable, the study does not report any data.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tseitlin, A. Resiliency through Failure: Netflix's Approach to Extreme Availability in the Cloud; 2013. Available online: https://qconnewyork.com/ny2013/node/281.html (accessed on 15 November 2021).
2. Alsberg, P.A.; Day, J.D. A Principle for Resilient Sharing of Distributed Resources. In Proceedings of the 2nd International Conference on Software Engineering October, San Francisco, CA, USA, 13–15 October 1976; pp. 562–570.
3. Meyer, J.F. Defining and evaluating resilience: A performability perspective. In Proceedings of the International workshop on performability modeling of computer and communication systems (PMCCS), Eger, Hungary, 17–18 September 2009; Available online: http://ftp.eecs.umich.edu/people/jfm/PMCCS-9_Slides.pdf (accessed on 15 November 2021).
4. Avizienis, A.; Laprie, J.-C.; Randell, B.; Landwehr, C. Basic concepts and taxonomy of dependable and secure computing. *IEEE Trans. Dependable Secur. Comput.* **2004**, *1*, 11–33. [CrossRef]
5. *Resilient Computing Systems*; Anderson, T. (Ed.) Wiley: New York, NY, USA, 1985; ISBN 0471845183.
6. Rosati, J.D.; Touzinsky, K.F.; Lillycrop, W.J. Quantifying coastal system resilience for the US Army Corps of Engineers. *Environ. Syst. Decis.* **2015**, *35*, 196–208. [CrossRef]
7. Ron, R.; Richard, G.; Deborah, B.; Rosalie, M. Draft SP 800-160 Vol. 2, Systems Security Engineering: Cyber Resiliency Considerations for the Engineering of Trustworthy Secure Systems. 2018. Available online: https://insidecybersecurity.com/sites/insidecybersecurity.com/files/documents/2018/mar/cs03202018_NIST_Systems_Security.pdf (accessed on 16 April 2020).
8. Carias, J.F.; Borges, M.R.S.; Labaka, L.; Arrizabalaga, S.; Hernantes, J. Systematic Approach to Cyber Resilience Operationalization in SMEs. *IEEE Access* **2020**, *8*, 174200–174221. [CrossRef]
9. Hopkins, S.; Kalaimannan, E.; John, C.S. Foundations for Research in Cyber-Physical System Cyber Resilience using State Estimation. In Proceedings of the 2020 SoutheastCon, Raleigh, NC, USA, 28–29 March 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–2, ISBN 978-1-7281-6861-6.
10. Web of Science. Available online: https://clarivate.com/webofsciencegroup/solutions/web-of-science/ (accessed on 20 July 2021).
11. Nygard, M.T. *Release It! Design and Deploy Production-Ready Software*, 2nd ed. 2018. Available online: https://www.oreilly.com/library/view/release-it-2nd/9781680504552/ (accessed on 15 November 2021). ISBN 9781680502398.
12. Farraj, A.; Hammad, E.; Kundur, D. A Cyber-Physical Control Framework for Transient Stability in Smart Grids. *IEEE Trans. Smart Grid* **2018**, *9*, 1205–1215. [CrossRef]
13. Jean-Claude, L. *From Dependability to Resilience*; DSN: Anchorage, AK, USA, 2008; p. 8. Available online: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.331.8948&rep=rep1&type=pdf (accessed on 15 November 2021).
14. Hamel, G.; Välikangas, L. The quest for resilience. *Harv. Bus. Rev.* **2004**, *62*, 355–358.
15. Gray, J.N. Why do computers stop and what can be done about it? In Proceedings of the 5th Symp. on Reliability in Distributed Software 1986, Los Angeles, CA, USA, 13–15 January 1986; pp. 3–12. Available online: http://bnrg.eecs.berkeley.edu/~randy/Courses/CS294.F07/10.1.pdf (accessed on 15 November 2021).
16. Hollnagel, E. Prologue: The scope of resilience engineering. In *Resilience Engineering in Practice: A Guidebook*; Hollnagel, E., Pariès, J., Woods, D., Wreathall, J., Eds.; Aldershot: Hampshire, UK, 2011; pp. 29–39.
17. Hollnagel, E. *Safety-II in Practice: Developing the Resilience Potentials*; Routledge: London, UK; New York, NY, USA, 2018; ISBN 1138708925.
18. Hollnagel, E.; Leonhardt, J.; Macchi, L.; Kirwan, B. White Paper on Resilience Engineering (Eurocontrol). Available online: https://www.eurocontrol.int/sites/default/files/2019-07/white-paper-resilience-2009.pdf (accessed on 12 December 2019).
19. De Florio, V. On the Constituent Attributes of Software and Organizational Resilience. *Interdiscip. Sci. Rev.* **2013**, *38*, 122–148. [CrossRef]
20. De Florio, V. Robust-and-evolvable resilient software systems. In *ASAS '11*; Association for Computing Machinery: New York, NY, USA, 2011; p. 10. ISBN 9781450308533.
21. Herrera, M.; Abraham, E.; Stoianov, I. A Graph-Theoretic Framework for Assessing the Resilience of Sectorised Water Distribution Networks. *Water Resour. Manag.* **2016**, *30*, 1685–1699. [CrossRef]
22. Bakkensen, L.; Fox-Lent, C.; Read, L.; Linkov, I. Validating Resilience and Vulnerability Indices in the Context of Natural Disasters: Validating Resilience and Vulnerability Indices. *Risk Anal.* **2016**, *37*, 982–1004. [CrossRef] [PubMed]
23. Cutter, S.L.; Burton, C.G.; Emrich, C.T. Disaster Resilience Indicators for Benchmarking Baseline Conditions. *J. Homel. Secur. Emerg. Manag.* 2010, 7. [CrossRef]
24. Cutter, S.L.; Ash, K.D.; Emrich, C.T. The geographies of community disaster resilience. *Glob. Environ. Chang.* **2014**, *29*, 65–77. [CrossRef]
25. Clark-Ginsberg, A. What's the Difference between Reliability and Resilience? 2016. Available online: https://www.researchgate.net/profile/Aaron-Clark-Ginsberg/publication/320456274_What's_the_Difference_between_Reliability_and_Resilience/links/59e651230f7e9b13aca3c2ba/Whats-the-Difference-between-Reliability-and-Resilience.pdf (accessed on 15 November 2021). [CrossRef]
26. Alexander, D.E. Resilience and disaster risk reduction: An etymological journey. *Nat. Hazards Earth Syst. Sci.* **2013**, *13*, 2707–2716. [CrossRef]
27. Mayunga, J.S. Understanding and applying the concept of community disaster resilience: A capital-based approach. *Ummer Acad. Soc. Vulnerability Resil. Build.* **2007**, *1*, 1–16.

28. Mitra, S.; Brelsford, K.; Sanda, P.N. Cross-layer resilience challenges: Metrics and optimization. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 8–12 March 2010*; IEEE: Piscataway, NJ, USA, 2010; pp. 1029–1034. ISBN 978-3-9810801-6-2.
29. Bodeau, D.; Graubart, R. Cyber Resiliency Design Principles: Selective Use Throughout the Lifecycle and in Conjunction with Related Disciplines. 2017. Available online: https://www.mitre.org/sites/default/files/publications/PR%2017-0103%20Cyber%20Resiliency%20Design%20Principles%20MTR17001.pdf (accessed on 15 November 2021).
30. Bodeau, D.; Graubart, R. Cyber Resilience Metrics: Key Observations 2016. Available online: https://www.mitre.org/sites/default/files/publications/pr-16-0779-cyber-resilience-metrics-key-observations.pdf (accessed on 15 November 2021).
31. Hukerikar, S.; Engelmann, C. Resilience Design Patterns-A Structured Approach to Resilience at Extreme Scale (version 1.0). 2016. Available online: https://arxiv.org/abs/1708.07422 (accessed on 15 November 2021).
32. Carias, J.F.; Labaka, L.; Sarriegi, J.M.; Hernantes, J. An Approach to the Modeling of Cyber Resilience Management. In Proceedings of the 2018 Global Internet of Things Summit (GIoTS), Bilbao, Spain, 4–7 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–6, ISBN 978-1-5386-6451-3.
33. World Economic Forum. *A Framework for Assessing Cyber Resilience*; World Economic Forum: Geneva, Switzerland, 2016.
34. Ligo, A.K.; Kott, A.; Linkov, I. How to Measure Cyber-Resilience of a System With Autonomous Agents: Approaches and Challenges. *IEEE Eng. Manag. Rev.* **2021**, *49*, 89–97. [CrossRef]
35. Haque, M.A.; Shetty, S.; Krishnappa, B. ICS-CRAT: A Cyber Resilience Assessment Tool for Industrial Control Systems. In Proceedings of the 2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), Washington, DC, USA, 27–29 May 2009; IEEE: Piscataway, NJ, USA, 2019; pp. 273–281, ISBN 978-1-7281-0006-7.
36. Kott, A.; Linkov, I. To Improve Cyber Resilience, Measure It. 2021. Available online: https://arxiv.org/pdf/2102.09455 (accessed on 14 November 2021).
37. Kotenko, I.; Saenko, I.; Lauta, O. Analytical modeling and assessment of cyber resilience on the base of stochastic networks conversion. In Proceedings of 2018 10th International Workshop on Resilient Networks Design and Modeling (RNDM), Longyearbyen, Norway, 27–29 August 2018; Heegaard, P.E., Helvik, B.E., Rak, J., Eds.; IEEE: Piscataway, NJ, USA, 2018; pp. 1–8, ISBN 978-1-5386-7030-9.
38. Ulrich, J.; McJunkin, T.; Rieger, C.; Runyon, M. Scalable, Physical Effects Measurable Microgrid for Cyber Resilience Analysis (SPEMMCRA). In Proceedings of the 2020 Resilience Week (RWS), Salt Lake City, ID, USA, 19–23 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 194–201, ISBN 978-1-7281-8693-1.
39. Zhao, P.; Gu, C.; Ding, Y.; Liu, H.; Bian, Y.; Li, S. Cyber-Resilience Enhancement and Protection for Uneconomic Power Dispatch Under Cyber-Attacks. *IEEE Trans. Power Deliv.* **2021**, *36*, 2253–2263. [CrossRef]
40. Dai, J.; Xu, Y.; Wang, Y.; Nguyen, T.-L.; Dasgupta, S. A Cyber-Resilience Enhancement Method for Network Controlled Microgrid against Denial of Service Attack. In Proceedings of IECON 202-46th Annual Conference of the IEEE Industrial Electronics Society, Singapore, 18–21 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 3511–3516, ISBN 978-1-7281-5414-5.
41. Hopkins, S.; Kalaimannan, E.; John, C.S. Cyber Resilience using State Estimation Updates Based on Cyber Attack Matrix Classification. In Proceedings of the 2020 IEEE Kansas Power and Energy Conference (KPEC), Manhattan, KS, USA, 13–14 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6, ISBN 978-1-7281-5391-9.
42. Hossain-McKenzie, S.; Lai, C.; Chavez, A.; Vugrin, E. Performance-Based Cyber Resilience Metrics: An Applied Demonstration Toward Moving Target Defense. In Proceedings of the IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society, Omni Shoreham Hotel, Washington, DC, USA, 20–23 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 766–773, ISBN 978-1-5090-6684-1.
43. Dyka, Z.; Vogel, E.; Kabin, I.; Aftowicz, M.; Klann, D.; Langendorfer, P. Resilience more than the Sum of Security and Dependability: Cognition is what makes the Difference. In *Proceedings of the 2019 8th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 10–14 June 2019*; Stojanovic, R., Ed.; IEEE: Piscataway, NJ, UAS, 2019; pp. 1–3. ISBN 978-1-7281-1739-3.
44. Jen, E. Stable or robust? What's the difference? *Complexity* **2003**, *8*, 12–18. [CrossRef]
45. Dyka, Z.; Vogel, E.; Kabin, I.; Klann, D.; Shamilyan, O.; Langendörfer, P. No Resilience without Security. In 2020 9th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 8–11 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–5. Available online: https://ieeexplore.ieee.org/abstract/document/9134179 (accessed on 14 November 2021).
46. Deborah, B.; Richard, G.; Jeffrey, P.; Rosalie, M. Cyber Resiliency Engineering Framework. 2011. Available online: https://www.mitre.org/sites/default/files/pdf/11_4436.pdf (accessed on 16 April 2020).
47. Castano, V.; Schagmayaev, I.; Schagaev, I. *Resilient Computer System Design*; Springer: Cham, Switzerland, 2015; ISBN 9783319150680.
48. Kabin, I.; Dyka, Z.; Kreiser, D.; Langendoerfer, P. Horizontal Address-Bit DEMA against ECDSA. In *9th IFIP International Conference on New Technologies, Paris, France, 26–28 February 2018*; Mobility & Security: Piscataway, NJ, USA, 2018; ISBN 978-1-5386-3662-6.
49. Kabin, I.; Dyka, Z.; Kreiser, D.; Langendoerfer, P. *Horizontal Address-Bit DPA against Montgomery kP Implementation*; IEEE: Piscataway, NJ, USA; ReConFig '17: New York, NY, USA, 2017; ISBN 978-1-5386-3797-5.