SEQIS QualityNews

Analyse, Test und Management

Ausgabe H2/2019

Resilienz

Durch Krisen wachsen

Titel: „Meeresküste", Künstlerin: Claudia Toth, Technik: Acryl auf Keilrahmen

Analysis. Test. Management. Better IT Results.

Lesen Sie in dieser Ausgabe:

Ihre Meinung ist gefragt!

Nach den QualityNews ist bekanntlich vor den QualityNews! Schon bald arbeiten wir wieder auf Hochtouren an der nächsten, spannenden Ausgabe. Lesen Sie nur das, was Sie wirklich interessiert! Sagen Sie uns, welche Themen Sie spannend finden.

Kontaktieren Sie uns: marketing@SEQIS.com
Join us: twitter.com/SwTestIsCool

Wir freuen uns auf Ihre Vorschläge und Wünsche!

Über SEQIS QualityNews:
Dieses Magazin richtet sich an Gleichgesinnte aus den Bereichen Software Test, IT Analyse und Projektmanagement im IT Umfeld. Die SEQIS Experten berichten über ihre Erfahrungen zu aktuellen Themen in der Branche. Die Leser des Magazins gestalten die Ausgaben mit: Schreiben Sie uns Ihre Meinung im SEQIS Blog (www.SEQIS.com/de/blog-index) oder als Leserbrief.
Wenn Sie dieses Magazin abbestellen möchten senden Sie bitte ein Mail an marketing@SEQIS.com.

Impressum:
Information und Offenlegung gem. §5 E-Commerce-Gesetz und §25 Mediengesetz

Herausgeber: SEQIS GmbH,
Neusiedler Straße 36, A-2340 Mödling
Tel: +43 2236 320 320 0
Fax: +43 2236 320 320 350
info@SEQIS.com, www.SEQIS.com
Gericht: Bezirksgericht Mödling
Firmenbuchnummer: 204918a
Umsatzsteuer-ID: ATU51140607
Geschäftsführung: Mag. (FH) Alexander Vukovic, Mag. (FH) Alexander Weichselberger, DI Reinhard Salomon

Druck: druck.at Druck- und Handels-gesellschaft mbH, 2544 Leobersdorf
Erscheinungsweise: 2x pro Jahr


Mag. (FH) Alexander Vukovic


Mag. (FH) Alexander Weichselberger


DI Reinhard Salomon

Editorial

Sehr geehrte Leserin,
sehr geehrter Leser,

wir freuen uns, Ihnen die zweite Ausgabe der SEQIS QualityNews im Jahr 2019 zu präsentieren.

Vielen Dank für das positive Feedback zu unseren letzten Ausgaben zu den Themenbereichen AgilePM und Toolchain. Wir hoffen, wir konnten Ihnen damit interessanten Content zur Verfügung stellen und Sie stellenweise auch gut unterhalten. Über weitere Anregungen, Themenwünsche und Feedback Ihrerseits freuen wir uns.

Auch in dieser Ausgabe finden Sie neben den branchenbezogenen Artikeln auch nicht-technische Bereiche :

Im Heft finden Sie einige Kunstwerke der Lebenshilfe Niederösterreich der Werkstätten Baden und Mödling. Denn nicht nur unsere Spezialisten, sondern auch die Klienten der Lebenshilfe leben für ihre(n) Beruf(ung).

In dieser Ausgabe dreht sich alles um das Thema Resilienz. Auf einen sehr einfachen Punkt gebracht bedeutet das, Systeme zu bauen, die nicht nur nicht ausfallen, sondern immer ihren (definierten) Dienst tun. Recovery-Konzepte, USVs, skalierbare Systeme sind jedem von uns bekannt, hier kann man sicherlich das eine oder andere Requirement definieren oder auch beim Test ordentliche Stress-Szenarien realisieren um zu prüfen.

Auf den folgenden Seiten geben Ihnen unsere Experten einen Einblick in die vielseitigen Aspekte der Resilienz im Bereich IT.

Wir wünschen viel Lesevergnügen mit unseren SEQIS QualityNews!

Ihre SEQIS Geschäftsleitung

Neulich im Netz: Dinge die wir nicht brauchen (oder doch)?

von *Hansjörg Münster*

Von der Öffentlichkeit relativ unbemerkt wurde bereits in 2008 in den internationalen Schriftzeichen Standards ISO -10646 und Unicode 5.1 ein neues Zeichen eingeführt. Im Jahre 2017 hat der Rat für deutsche Rechtschreibung die Einführung dieses neuen Zeichen auch offiziell beschlossen: Ein Zeichen, das in keinem deutschen Wort vorkommt: Das große „ß".

Ich selbst habe das Schreiben und Lesen noch im 2. Drittel des vorigen Jahrhunderts erlernt – lange bevor man an eine Rechtschreibreform dachte. Damals kam das „ß" noch in viel mehr Worten vor. Verstanden habe ich damals als Volksschüler nicht, warum man zwischen „s", „ss" und „ß" unterscheiden muss. Ich habe es halt so (auswendig) gelernt und akzeptiert. Als dann die Rechtschreibreform kam, habe ich gehofft, dass das Ende des „ß" gekommen ist. Ich wurde eines Besseren belehrt, habe umgelernt und verstehe die Logik dahinter noch weniger. Aus „daß" wurde „dass", aber aus „Maßnahme" wurde nicht „Massnahme". Zumindest ist mir kein Wort bekannt, bei dem das „ß" neu eingeführt wurde.
Dass es auch ohne „ß" geht zeigen uns die Schweizer: In der Schweizer Variante von „Deutsch" gibt es das „ß" nur in Eigennamen.

Stellt sich aber doch die Frage nach der Motivation für die Einführung eines „ß". Gebraucht werde das große „ß" - so liest man – einerseits für amtliche Dokumente, Ausweise und Pässe: Da sei es vorgeschrieben, Eigennamen in Großbuchstaben zu schreiben. Andererseits brauche Designer und Typografen diesen Buchstaben um noch schönere Druckwerke (wie z.B.: Werbung) zu machen.

Schön und gut - diese Entscheidung war 2017, warum dazu 2019 einen Artikel dazu? Weil vor kurzem die erste Tastatur auf den Markt kam, deren Tasten nach dem Layout DIN-2137 T2 beschriftet sind: Cherry Stream 3.0 T2. Auf diese Tastatur sind alle europäischen Sonderzeichen aufgedruckt, selbst isländische Zeichen finden sich. Allerdings unterscheidet sich diese Tastatur beim Schreiben nicht von allen anderen: Zum Tippen solcher Sonderzeichen (auch beim großen „ß") müssen fingerakrobatische Kombinationen von <Alt Gr>, <Umschalt> etc. gedrückt werden.

Alles in Allem stellt sich mir die Frage, ob es nicht die billigere und einfachere Variante gewesen wäre, die Schweizer Regel im gesamten deutschsprachigen Raum einzuführen und das große „ß" aus unserem Sprachgebrauch und unseren Dokumenten zu verbannen. Zumal der Rat für deutsche Rechtschreibung weiterhin die Verwendung von doppel „S" als Ersatz für das große „ß" erlaubt. Irgendwie erinnert mich das an einen Schildbürgerstreich…

Jetzt gilt es noch ein Rätsel lösen: wie können Sie es auf Ihrer Tastatur eingeben:
Windows: <ALT GR> + <UMSCHALT> + „ß" oder <ALT" + „7838"
Für MacOS gibt es noch keine Möglichkeit (wie auch unter iOS und Android). Ein Workaround unter MacOS ist die **Definition** einer automatischen Textersetzung (Systemeinstellungen > Tastatur > Text). Das Zeichen selbst muss man sich vor mittels Zwischenablage aus dem Netz kopieren (z.B. von http://typografie.info/versaleszett/).

Zum Schluss bleibt nur mehr das neue Zeichen vorzustellen: Das große „ß": ß (Erkennen Sie den Unterschied?)

P.S.: Natürlich gibt es eine Regel für das „ß": Wenn der Vokal vor dem „s"-Laut lang ist, schreibt man wie bisher „ß"■

Weblinks:
https://www.rechtschreibrat.com/DOX/rfdr_PM_2017-06-29_Aktualisierung_Regelwerk.pdf

https://de.wikipedia.org/wiki/Gro%C3%9Fes_%C3%9F

http://typografie.info/versaleszett/

https://de.wikipedia.org/wiki/DIN_2137

https://de.wikipedia.org/wiki/Tastaturbelegung#Deutschland_und_%C3%96sterreich

https://cherry.de/PDF/DE_CHERRY_STREAM_3_0.pdf

https://page-online.de/typografie/grosse-news-das-versal-ss-ist-da/

Hansjörg Münster ist Principal Consultant und Teamlead bei SEQIS.

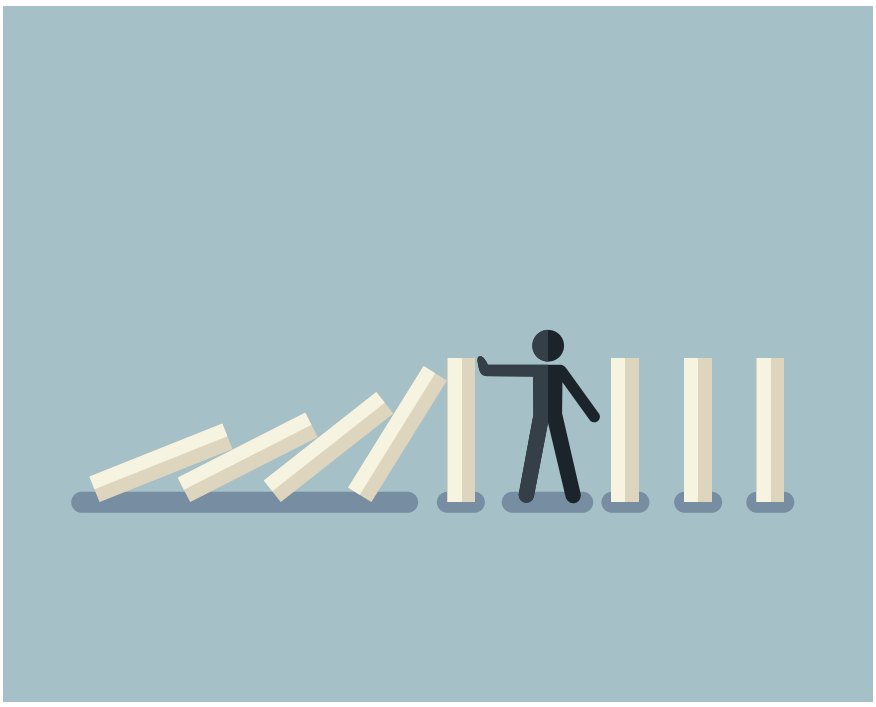
Als Allrounder deckt er ein breites Spektrum an Aufgaben ab. Die Schwerpunkte seiner Tätigkeit liegen in den Bereichen Test Management, Testautomation und Lasttest.

Ganz oben auf der Prioritätenliste des IT Profis steht, einen Nutzen und Mehrwert in der Qualitätssicherung seiner IT Projekte zu generieren.

Resilienz - ein Rückblick und ein Ausblick

von *Philip Stockerer*

Am 26. Juni 1991 waren große Teile von Washington DC, Maryland und West Virginia durch einen massiven Ausfall des öffentlichen Telefonsystems betroffen. Zwölf Millionen Amerikaner konnten keine Spitäler erreichen oder sich im Kindergarten nach ihren Kindern erkundigen. Schon damals hat man bei einer Pressekonferenz darauf hingewiesen, dass durch die immer raffinierter werdenden IT-Systeme und immer mehr voneinander abhängigen Netzwerksystemen die Wahrscheinlichkeit für Totalausfälle steigt. Es war ein Weckruf für ein Land, dessen zentrale Infrastruktur massiv von diesen Systemen abhängig war. Netzwerke und IT-Systeme wurden immer größer und ein Ausfall hatte immer größere Auswirkungen.



Quelle: Adobe Stock, Urheber: Elokua

Ein einfacher, kleiner Bug im Code verursachte diesen massiven Ausfall im Jahr 1991. Denken wir an Computersysteme und IT Infrastruktur in der heutigen Zeit, möchte man sich gar nicht ausmalen, welch verheerende Wirkung ein kleiner Bug, eine winzige Abweichung auf unser alltägliches Leben hätte. Die Konsequenzen sind jedenfalls höher denn je.

Aus diesem Grund habe auch ich mir das Thema dieser QualityNews Ausgabe zur Brust genommen und mich intensiv mit dem Thema Resilienz auseinandergesetzt. Bei meinen Recherchen bin ich auf immer mehr Definitionen und Vorgehensweisen gestoßen, oft war keine klare Linie zu erkennen und richtige Ratschläge, wie Resilienz gelebt werden kann waren rar.

Beginnen wir also mit dem Wesentlichen: Was ist Resilienz und woher kommt die Idee?

Resilienz bezeichnet die Fähigkeit von Systemen, bei Störungen oder Ausfällen nicht vollständig zu versagen, sondern die Dienstleistung aufrechtzuerhalten. Es ist also die Fähigkeit eines Systems, trotz massiver externer oder interner Störungen den Status Quo beizubehalten - also zu funktionieren. Resilienz ist im Bereich des Betriebskontinuitätsmanagements angesiedelt und dadurch auch ganz eng verwoben mit dem Risikomanagement, der IT-Notfallplanung und der Informationssicherheit.

Die Idee ist militärischen Ursprungs, welcher in der chinesischen Literatur belegt wird. Um 500 vor Christus sprach Sun Tzu schon über die fortdauernde Planung, Umsetzung und den erfolgreichen Abschluss eigener Pläne trotz Feindeinwirkung. Militärtheoretiker wie Clausewitz (1780 bis 1831) führten die Idee weiter und mit der industriellen Revolution wurde die Vorgehensweise auf das betriebliche Geschehen übertragen. Die Idee ist also keinesfalls neu. Die klügsten Köpfe der Kriegsführung haben früh erkannt, dass Resilienz für Erfolg unabdingbar ist.

Wie komme ich nun zu Resilienz im Betrieb?

Zuallererst gilt es, realistische Katastrophenszenarien zu identifizieren. Im Bereich der IT wäre dies zum Beispiel ein kompletter Systemausfall. Davon abgesehen spielen viele andere Faktoren eine tragende Rolle im Betrieb. Stellen Sie sich zum Beispiel vor, am Standort Ihrer Datenbank fällt der Strom für mehrere Stunden aus, bei Straßenbauarbeiten vor Ihrem Büro wird das Glasfaserkabel gekappt oder Ihr Personal fällt durch eine Epidemie massenhaft aus und Sie haben niemanden, der Ihre Systeme betreuen kann.

Sie müssen außerdem herausfinden, welche Prozesse in solchen Situationen unbedingt aufrechterhalten werden müssen und welche Use-Cases Ihrer Anwendung unter keinen Umständen fehlerhaft oder ohne Funktion sein dürfen. In einer Bankenapplikation wäre dies zum Beispiel der Zahlungsverkehr und ggf. die Synchronisation mit dem Mainframe, damit kein Cent verloren geht und keine Buchung doppelt durchgeführt wird. Bei selbstfahrenden Autos muss die Spur und Geschwindigkeit trotz Absturz der Computersysteme beibehalten werden und sämtliche Berechnungen zur Kollisionsverhinderung dürfen auch nicht fehlen. Ergänzend müsste sinnvollerweise auch das Fahrzeug gestoppt werden.

Sind die initialen Analysen abgeschlossen, gilt es, am wichtigsten Gut Ihres Unternehmens zu arbeiten: Ihren Mitarbeitern und deren Mindset.

Der wichtigste Faktor für Resilienz in Ihrem Unternehmen ist eine Unternehmenskultur, in der Mitarbeiter schnell auf Veränderungen und Ausfälle reagieren können und wirklich verstehen wie sämtliche Systeme zusammenarbeiten und funktionieren. Der beste Weg diese Kultur aufzubauen ist kontinuierliches Training und eine Veränderung im Mindset. Statt darauf zu pochen, Ausfälle und Fehler zu verhindern und jeden Ausfall als Misserfolg zu werten müssen wir sie als wertvolle Übung und Lehre ansehen. Ja, Ausfälle passieren. Es ist keine Frage ob, sondern wann der nächste Systemausfall eintritt. Dies gilt es zu akzeptieren. Durch jeden Systemausfall lernt man wichtige Lektionen für den weiteren Betrieb und ist nach entsprechender Optimierung besser auf den nächsten Ausfall vorbereitet.

Wie also nun auf den Ernstfall vorbereiten? Wie trainiere ich meine Mitarbeiter? Ganz einfach: Zerstören

Titel: „Schmetterling", Künstler: Katharina Stockreiter, Technik: Acryl auf Leinwand

Sie Ihr System regelmäßig. Klingt verrückt - ist aber effektiv. Und der Proof of Concept? Alle großen IT Unternehmen. Google nutzt "DiRT" (Disaster Recovery Testing) – dabei werden eine Woche lang alle Katastrophenszenarien verprobt. Netflix nutzt "Chaos Monkey", ein Tool, welches zufällig Instanzen auf Amazon Web Services (AWS) im laufenden Betrieb herunterfährt oder crashed. Und Amazon selbst dürfte eine der ersten Firmen gewesen sein, die dieses Konzept mit "Game Day" umgesetzt hat. "Game Day" zerstört empfindliche Komponenten der Systemlandschaft. Alle diese Programme haben ein Ziel: Durch Erleben und Erfahren des worst-case Szenarios, das System und dessen Verhalten kennenzulernen, zukünftig auf den Ernstfall vorbereitet zu sein und zu wissen, wie man am besten und schnellsten die notwendige Stabilität wiederherstellen kann.

Software unterstützend nutzen

Neben Ihren Mitarbeitern und dem Mindset müssen Sie gegebenenfalls auch Ihren Technology Stack auf Vordermann bringen. Sie können sowohl die Reaktionszeit Ihrer Mitarbeiter als auch die Möglichkeiten einen Katastrophenfall zu erkennen durch speziell dafür entwickelte Software wesentlich verbessern. Musste Ihr Betrieb früher noch mühsam Log-Files nach einer Exception durchforsten, kann dies per Software mittlerweile automatisch analysiert werden und durch entsprechend eingerichtetes Alerting Ihre Mitarbeiter über Fehlverhalten in Kenntnis setzen. Mit der richtigen Visualisierungssoftware haben Ihre Mitarbeiter sämtliche KPIs wie Responsetimes, Requestcount und HTTP Status Codes ständig im Blick, können eine Veränderung zeitnah erkennen und mit den richtigen Maßnahmen reagieren.

Spinnen wir den Gedanken der Softwareunterstützung noch weiter kann man sich sogar vorstellen den Prozess der Wiederherstellung zu automatisieren. Mit der richtig trainierten künstlichen Intelligenz werden Gegenmaßnahmen

automatisch eingeleitet und in einer virtualisierten Umgebung mehr Ressourcen zur Verfügung gestellt. Die Alerts erreichen Ihre Mitarbeiter nur noch, wenn die automatischen Gegenmaßnahmen nicht greifen und Ihren Mitarbeitern bleibt mehr Zeit für das Daily Business. Denke ich noch weiter in die Zukunft sehe ich KI, welche in der Lage ist, Katastrophenfälle präzise vorauszusagen und noch bevor ein solcher Fall eintreten könnte automatisch präventive Maßnahmen dagegen zu ergreifen und umzusetzen. Ja, sogar die Dokumentation könnte automatisch erfolgen und Ihre Mitarbeiter erhalten nur noch eine Zusammenfassung per Mail. Prediction durch KI wird allerdings nicht ohne massiven Entwicklungsaufwand umsetzbar sein und ist aus heutiger Sicht noch Zukunftsmusik.

Blech einplanen und für sich nutzen

Neben den Softwarelösungen zur Unterstützung des Monitorings und Ihres DevOps Teams können Sie Ihre Infrastruktur schon in der Planung auf Resilienz auslegen. Beispiele hierfür wären ein verteiltes System, welches neben der besseren Skalierbarkeit auch eine erhöhte Ausfallsicherheit bietet. Die Last, welche vom System bewältigt werden muss, kann durch verteilte Systeme im Regelfall auch besser weggesteckt und über mehrere Prozessoren und Rechner verteilt werden.

Natürlich stellen auch komplett redundant ausgelegte Infrastrukturen eine valide Option dar, sofern die Kosten-Nutzen-Rechnung dafür spricht. Ein Zwei- oder gar Drei - Standorte Konzept kann Ihr System vor vielen Katastrophenfällen schützen. Fällt auf Standort 1 zum Beispiel der Strom aus, stemmen Standort 2 & 3 die Last problemlos weiter und halten Ihr System am Leben. Eine defakto nicht vorhandene “Mean time to recovery” (MTTR) - das ist die mittlere Reparaturzeit nach einem Ausfall eines Systems - ist der Lohn. Wenn die Software und Infrastruktur perfekt zusammenspielen ist ein Wechsel des ausführenden Standorts ohne User-Impact machbar und man kommt der Vision eines zu 100% verfügbaren Systems einen großen Schritt näher. Durch die erhöhte Komplexität können allerdings mehr Fehlerquellen eingeführt werden und die Anzahl der Ausfälle kurzfristig steigen.

Egal in welchem Bereich - Resilienz spielt eine riesen Rolle. Ausfallsicherheit ist in der heutigen Zeit das A und O in jeder Branche und in jedem Unternehmen, unabhängig davon ob ich nun technische oder soziale Resilienz erreichen möchte, muss ich zuerst am Mindset meiner Mitarbeiter und der Unternehmenskultur arbeiten. Ungeachtet der Größe Ihres Unternehmens ist, wenn Sie noch nicht über Resilienz, Ausfallsicherheit oder Betriebskontinuitätsmanagement nachgedacht haben, jetzt der beste Zeitpunkt dafür. Denn Sie suchen sich den Zeitpunkt des nächsten Katastrophenfalles nicht aus, der Zeitpunkt wird Sie erwählen. Sie können lediglich darauf vorbereitet sein. ■

Philip Stockerer ist Consultant für IT-Analyse, Software Test und Projekt Management.

Als professioneller Projektbegleiter und Berater in verschiedensten Projektsituationen ist er der richtige Ansprechpartner für sämtliche Themen rund um Testprozessmanagement, Testautomation und Last- und Performancetests.

In agilen Projektteams fühlt er sich besonders wohl und unterstützt das Team funktionsübergreifend bei allen anfallenden Aufgaben.

Resilienz: Stabilität im Grenzbereich

von *Klemens Loschy*

Quelle: Fotolia

Montag Vormittag: Typischerweise der stärkste Tag der Woche (wenn es um die Belastung eines Systems geht). Oder es ist Monatsanfang/ Monatsende (auch das bedeutet typischerweise mehr Last als normal).
Vielleicht fallen diese beiden Ereignisse sogar zusammen (was die Belastung zusätzlich erhöht).
Um das etwas zu verdeutlichen: Ich rede nicht von ein paar % Lastanstieg, sondern eher um 50% bis hin zu einem Vielfachen der „normalen Durchschnittslast".

Wir Menschen haben die unterschiedlichsten Angewohnheiten, wenn es um die Benutzung von Software geht. Da gibt es z.B. noch das ZiB22/ZiB24 Phänomen: nach dem Ende dieser Sendungen werden oft noch einmal die letzten Lastspitzen in z.B. eCommerce oder Bankenapplikationen gemessen, bevor in der Nacht die Belastung auf ein Minimum sinkt. Das steht natürlich in keinem Verhältnis mehr zur „Montag Morgen am Monatsanfang"-Last, verdeutlicht aber die unregelmäßige und teils schwer vorhersagbare Belastung von Systemen.

Wozu aber das Ganze?
Unsere Systeme müssen funktionieren (stabil und performant), auch oder gerade in den beschriebenen Ausnahmesituationen: und genau das verstehe ich unter Resilienz – Stabilität im Grenzbereich!
Im Schönwetterflug zu funktionieren ist heutzutage keine Kunst mehr, „Blech" ist im Normalfall günstig, eine Überdimensionierung der Systeme daher einfach. Damit können die meisten Belastungsspitzen der meisten Systeme sehr einfach erschlagen werden. Nun haben wir es in er Praxis aber oft nicht mit „den meisten Systemen", sondern mit hochkomplexen, ineinander verzahnten, von verschiedenen Lieferanten gebauten und gewarteten multi-Tier Anwendungen zu tun - so wie Sie wahrscheinlich auch! „Blech" allein dazustellen steigert dabei nur die Temperatur und den Stromverbrauch im Rechenzentrum, Leistungsgewinne und Stabilität bekommt man in solchen Systemen nicht mehr so einfach. Expertise und die Möglichkeit diese Ausnahmesituationen zu simulieren und die daraus gewonnen Erkenntnisse wieder in den Systemaufbau zurückfließen zu lassen, können nachhaltig die Resilienz (und so ganz nebenbei auch die Performance) steigern.

Technische Wunderwerke!?
Um ein System theoretisch resilient zu machen, gibt es viele technische Hilfsmittel, die in den allermeisten Fällen auch schon verwendet werden: Hardware Loadbalancing, mehrere Reverse Proxies vor mehreren Applikationsservern (im Parallelbetrieb oder aktiv/passiv). Datenbanken, die als Cluster aufgesetzt sind und die Daten in Realtime auf mehrere Maschinen persistieren, bis natürlich runter zu ausfallsicherer Storage, die transparent die Daten auf mehrere physische Platten duplizieren. Jetzt multiplizieren wir das ganze mal zwei und verteilen das Konstrukt auf zwei verschiedene Standorte – wir wollen ja schließlich gegen Stromausfälle oder andere Katastrophen gewappnet sein. Das Ergebnis ist, dass wir ein an sich überschaubares System zu einem komplexen

Giganten aufgeblasen haben. Oft endet aber hier das Streben nach Resilienz: wir haben ja alles doppelt und dreifach, kann ja gar nix mehr passieren... Hand hoch: wer hat alle diese theoretischen technischen Wunderwerke praxisnahe getestet, bevor sie live genommen wurden? Wer kann sagen, wie das System reagiert und mit welchen Auswirkungen der Endbenutzer zu rechnen hat, wenn auf einmal einer der beiden Standorte wegbricht? Erholt sich ein System nach punktueller Überbelastung oder endet das in einem „shutdown -r now"? Und, wir alle kennen Murphys Law („Anything that can go wrong will go wrong."), wenn das genau an einem „Montag Morgen am Monatsanfang" passiert? Und wer prüft diese Ausnahmesituation auf regelmäßiger Basis oder analysiert die produktive Lastentwicklung? Denn einerseits werden Software- und die Hardwarekonzepte regelmäßig aktualisiert, müssten also auch regelmäßig auf deren korrektes Verhalten hin getestet werden. Und andererseits ändert sich das spezifische Benutzerverhalten von „sich aus" – und darauf muss man auch ein Auge haben.

Simulation der Realität und daraus lernen!

Meine Empfehlung wird jetzt kaum noch überraschen: Ohne (regelmäßige) Tests all dieser Konzepte haben wir einfach nur viel Geld für „ein gutes Gefühl" investiert. Holen wir uns doch besser die Gewissheit und testen die Umsetzung, und zwar so realitätsnahe wie nur irgend möglich. Beantworten wir die obigen Fragen (und noch viele mehr) und gehen wir einen Schritt weiter als nur „ein gutes Gefühl" zu haben. Last- und Performance Tests helfen dabei die Realität zu simulieren, und währenddessen „ziehen wir den Stecker" und lernen – aber in einem gesicherten Umfeld und nicht das erste Mal – wenn die Produktivumgebung am Glühen ist. Und dann optimieren wir! Wir passen die Tomcat Konfiguration an, ändern JVM Settings, erhöhen die maximal möglichen Threads oder die Datenbank Connections und verringern vielleicht auch noch die Timeouts und prüfen alle Änderungen gezielt durch Last- und Performance Tests. Machen wir einen Stresstest und versuchen wir die besten Einstellungen zu finden, damit bei Überbelastung ein Großteil der Benutzer weiterhin (stabil und performant) servisiert werden kann und nur der Teil der Benutzer negativ betroffen ist, der tatsächlich zu viel ist.

Der Tropfen, der das Fass zum Überlaufen bringt!

Viele Systeme reagieren bei Überbelastung indeterministisch: oftmals schaukeln sich die Einzelkomponenten dann derart gegenseitig auf, dass auch nur ein wenig Überbelastung zu einem plötzlichem Totalausfall führen kann. Ein komplexes System auf diese Situationen abzustimmen ist fast schon ein Projekt für sich. Einfacher, und in der Praxis durchaus verbreitet, ist das Prinzip der liebevoll genannten „Spaßbremse": eine Komponente nimmt die Rolle des Gatekeepers ein und schützt das eigentliche System gezielt vor Überbelastung. Welche Metriken und Grenzwerte dafür herangezogen muss natürlich auch erstmal mit ermittelt werden (z.B. technische wie Requests pro Sekunde oder fachliche wie z.B. Suchanfragen pro Sekunde). Dabei können Last- und Performance Tests durch gezielte Simulation die notwendigen Daten liefern. Die „Spaßbremse" misst also die definierten KPIs und beginnt bei Überschreitung der definierten Grenzwerte Anfragen einfach zu blocken. Richtig eingestellt hat das den Effekt, dass mein eigentliches System vor Überbelastung gezielt geschützt wird und immer nur ein geringer Teil der Benutzer bzw. der Anfragen abgelehnt werden. Alle anderen Benutzer sind davon nicht betroffen und können weiterhin stabil und performant das System weiterverwenden.

Ich persönlich habe erst dann „ein gutes Gefühl", wenn ich mit Gewissheit die Auswirkungen solcher Szenarien kenne. Und ja, das bedeutet im Normalfall einen nicht unerheblichen Aufwand (Zeit und Kosten) im Vorfeld! Aber: Jeder, der an einem Wochenende oder Feiertag schon einmal mit einem, durch fehlende Resilienz verschuldetem, Ausfall konfrontiert war, wird mir bestätigen, dass sich der Aufwand im Vorfeld für alle lohnt! ■

Klemens Loschy ist Principal Consultant, Teamlead und Spezialist in den Bereichen IT-Analyse, Software Test und Projekt Management.

Er kann auf jahrelange Erfahrung in den Bereichen Testautomation, Last-Tests und Performance Engineering, funktionale Tests, Testen in agilen Teams, Anwendungsentwicklung von Testsoftware sowie Beratung und Unterstützung in zahlreichen Projekten unterschiedlichster Branchen zurückblicken.

Resilienz - worauf sollten wir als Agile Wizards achten?

von *Alexander Vukovic*

Resilienz könnte man auch als Widerstandsfähigkeit eines Systems bezeichnen. Wie reagiert ein System auf unvorhersehbare Ereignisse? Kann es den Betrieb aufrecht erhalten? Falls nicht, kann der Betrieb nach dem Ernstfall weitergeführt werden?

Diese Frage beschäftigt im Moment hauptsächlich die CISOs (Chief Information Security Officers) und die Operationteams, die versuchen müssen, die Produktionsarchitektur möglichst ausfallsicher und skalierbar anzulegen. Was ist nun für Agile Wizards, also Mitglieder eines agilen Teams, die cross functional aufgestellt sind und sowohl entwickeln als auch testen, zu beachten?

Bereits bei der Analyse muss Resilienz berücksichtigt werden, Resilienz by Design ist viel leichter umsetzbar als sie nachträglich über andere Kunstgriffe herzustellen. Natürlich spielt Resilienz auch beim Coding und Testing der Software eine Rolle. Es ist sehr schwierig sich gegen unbekannte Gefahren zu rüsten. Einige können auf Softwareseite gar nicht adressiert werden, wie z.B. ein Verlust jeglicher Elektronik durch einen Magnetsturm oder einen EMP-Angriff (Electro Magnetic Pulse). Andere typische Szenarios sind jedoch auch softwareseitig korrekt adressierbar.

Ein Beispiel: eine Webapplikation erlaubt es einen komplexen Geschäftsprozess durchzuführen, dieser hat eine typische Durchlaufzeit von 30 Minuten. Was sollte nun der Product Owner oder auch die Agile Wizards im Team in Bezug auf Resilienz bedenken? Nun ein Geschäftsprozess der durchschnittlich 30 Minuten durchläuft ist einem erhöhten Risiko einer Unterbrechung ausgesetzt. Dieses Risiko ist zu bewerten und einzuschätzen. Gemäß Kano ist die Erwartungshaltung des Kunden, dass die Applikation mit einer Unterbrechung korrekt umgehen kann, immer ein Basismerkmal. Er würde dies niemals als explizite Anforderung formulieren. Umso wichtiger ist, dass im Zuge der Analyse dies vom Product Owner und dem Team bedacht wird.

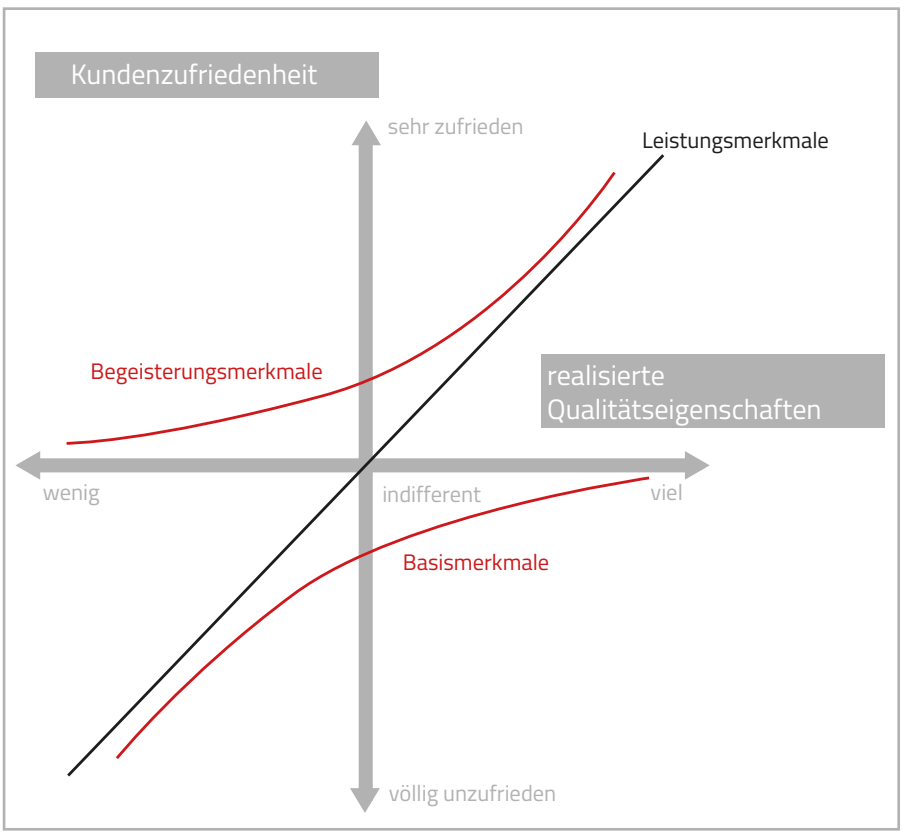


Abbildung: Kano

Im konkreten Beispiel könnte ein Akzeptanzkritierium festgelegt werden, das z.B. lautet: „Wird der Prozess unterbrochen, oder kommt es zu einem unvorhergesehenen Ausfall, so muss danach der Geschäftsprozess wieder aufgenommen werden können. Sollte dies nicht möglich sein, dann ist der Ursprungszustand vor dem Start des Geschäftsprozesses wiederherzustellen".

Wie könnte dies nun implementiert werden? Der Einfachheit halber gehen wir davon aus, dass der Server mehrere Services implementiert hat, die alle in die gleiche Oracle Instanz persistieren. Der Agile Wizard, der den Code schreibt, muss diese Anforderung nun umsetzen. Wie könnte er dies konkret machen?

Falsch wäre es, die einzelnen Schritte des Geschäftsprozesses als einzeln gekapselte Transaktionen in die Datenbank zu schreiben. Dabei entsteht das Risiko, dass einzelne Transaktionen erfolgreich geschrieben wurden, andere nicht. Es wäre notwendig in der Prozesskette fachlich konsistente Wiederaufsetzpunkte zu definieren. Alle Daten, die

Titel: „Sängerin", Künstler: Max Minichmayr, Technik: Pointilismus

seit dem letzten Wiederaufsetzpunkt geschrieben wurden, müssten im Desasterfall rückgängig gemacht werden. Dies ist im Normalfall nur mit der Nutzung des sogenannten Two Phase Commits möglich. Das bedeutet, dass die Einzeltransaktionen weiterhin durch einzelne Transaktionen mit Commit oder Rollback abgeschlossen werden. Darüber hinaus wird aber nach jedem Aufsetzpunkt eine übergeordnete Transaktionsklammer gestartet und am Ende als nächster Wiederaufsetzpunkt committed. Kommt es zu einem Ausfall wird die äußere Transaktionsklammer rückgängig gemacht („rollback"), obwohl einzelne Transaktionen darin bereits erfolgreich committed wurden.

Klarerweise muss sich das agile Team auch fragen, wie diese Anforderung zu testen ist. Das ist ebenfalls eine spezielle Herausforderung, da hier die Art des Ausfalls antizipiert werden muss. Die geeignetste Vorgehensweise ist in der Regel die Durchführung eines sogenannten Katastrophentests.
Dabei wirkt auf ein im Schönwetterflug befindliches System (der Geschäftsprozess wird mit fachlich und technisch korrekten Eingaben durchgeführt) eine externe Katastrophe ein, die das System beeinträchtigt. Das kann z.B. ein Netzwerkausfall oder ein Stromausfall sein. Nun könnte man argumentieren, dass dies durch Maßnahmen auf Operationsseite nicht passieren kann (z.B. unterbrechungsfreie Stromversorgung, redundante Netzwerksysteme, usw.). Aber um wirklich auf Nummer sicher zu gehen, muss es die Software richtig machen. Es kann ja auch zu einem softwareseitigen Absturz kommen, der genau den gleichen Effekt hat.

Ein Katastrophentest des o.g. Systems könnte nun so aussehen:
Prozessstart > Wiederaufsetzpunkt 1 > Wiederaufsetzpunkt 2 > Schritt 3/5 wird gerade gespeichert ….
Eintritt der Katastrophe ….
Erwartetes Ergebnis: Prozess kann auf Wiederaufsetzpunkt 2 wieder aufgenommen werden, die Daten entsprechen konsistent dem Wiederaufsetzpunkt 2.

In einem unserer Projektaufgabenstellungen wurde der Datenbankserver im Moment des Schreibens einfach stromlos gemacht oder das Netzwerk zwischen Applikationsserver und Datenbankserver unterbrochen. Alternativ kann auch der Datenbankserver auf Prozessebene gekillt werden. In jedem Fall muss das System nach Beseitigung der Katastrophenursache (Strom, Netzwerk, DB-Service) wieder korrekt hochkommen und konsistente Daten anbieten. Wurden Teile gespeichert und Teile nicht, kann es zu heftigen Folgefehlern führen, das wäre dann das Gegenteil von Resilienz.
Zusammengefasst ist es wichtig, Resilienz bereits im Zuge der Analyse zu bedenken und bewusst zu berücksichtigen. Bei der Entwicklung der Applikation müssen entsprechende Techniken verwendet werden, die die Widerstandsfähigkeit der Software erhöhen und der Test muss dies auch nachweisen.

Haben Sie die Problemstellung Bewusstsein für Resilienz in Ihrem Team zu verankern? Wir unterstützen Sie gerne mit Workshops, Coaching oder Mitarbeit auf Zeit.■

Alexander Vukovic ist SEQIS Gründer und Chief Evangelist.

Er ist erster Ansprechpartner für alle agilen, testmethodischen und testtechnischen Anfragen. In der Praxis arbeitet er als Agile Quality Coach, Berater, Interims-Testmanager, CI-Experte und Lasttester. Mehr als 25 Jahre Beratertätigkeit führten ihn während seiner zahlreichen Projekte in die unterschiedlichsten Branchen und Länder.

Sein persönliches Motto „Es gibt keine Probleme, sondern nur nicht gefunden Lösungen" spiegelt sich in jedem Projekt wider.

Resilience Engineering - die vier Stufen resilienter Systeme

von *Christina Eder*

Definition

Ursprünglich kommt der Begriff Resilienz aus der Holzwirtschaft, um zu erklären, warum manche Holztypen schwere Ladungen aushalten können, ohne zu brechen und andere nicht . Vorrangig bekannt ist er heutzutage aus der Psychologie: Resilienz (von lateinisch resilire „zurückspringen", „abprallen") oder psychische Widerstandsfähigkeit ist die Fähigkeit, Krisen zu bewältigen und sie durch Rückgriff auf persönliche und sozial vermittelte Ressourcen als Anlass für Entwicklungen zu nutzen.
Das Gegenteil von Resilienz ist Verwundbarkeit (Vulnerabilität).
Im technischen Bereich beschreibt Resilienz „die Fähigkeit von technischen Systemen, bei Störungen bzw. Teil-Ausfällen nicht vollständig zu versagen, sondern wesentliche Systemdienstleistungen aufrecht zu erhalten."

Bedeutung im Alltag

Benjamin Scharte und Klaus Thoma stellen in einem Aufsatz im eBook „Resilienz – Ingenieurwissenschaftliche Perspektive" 2016 folgende These auf: „Komplexe Systeme wie zum Beispiel kritische Infrastrukturen, „resilient" zu gestalten, ist eine der größten Herausforderungen, denen sich unsere Gesellschaft gegenübersieht." Daran hat sich bis heute nicht viel verändert. Denn je mehr wir darauf angewiesen sind, dass komplexe technische Systeme funktionieren, desto wichtiger wird deren Resilienz. „Komplexe technische Systeme" finden sich heute überall, man denke nur an das „Internet of things". Immer mehr Dinge, die wir alltäglich benutzen, sind mittlerweile digitalisiert. Von unseren Uhren, über Autos, medizinische Geräte, Flugzeuge, unsere Heizungen, Haushaltsgeräte, und und und. Dies betrifft uns allerdings nicht nur als Einzelpersonen, sondern auch die Gesellschaft im Großen und Ganzen, da auch der Löwenanteil unserer Infrastruktur komplex vernetzt ist. Man denke an den Verkehr, an Kommunikationssysteme, Krankenhäuser, etc.
Bei Resilienz (oder präziser gesagt: beim Potenzial für resiliente Performance) geht es jedoch nicht darum, Fehler und Ausfälle zu vermeiden (dies wäre klassisches Safety Management) – Resilienz ist nicht das Gegenteil von mangelnder Sicherheit – sondern um die Suche nach Wegen und Möglichkeiten, Systeme so zu gestalten, dass sie unter unterschiedlichen Bedingungen und Einflüssen überleben und funktionieren. Es ist unvermeidbar, dass Fehler passieren, sowohl bei Menschen als auch bei Systemen. Wichtig ist, entsprechend damit umzugehen.
Mittlerweile hat sich ein eigener Berufszweig herauskristallisiert, der sich mit dieser Thematik befasst: Resilience Engineering (Es wird sogar schon ein Diplom für Resilience Engineering angeboten, siehe https://www.academy.fraunhofer.de/de/weiterbildung/energie-nachhaltigkeit/resilience-engineering.html).

Resilience Engineering

Stefan Hiermaier und Benjamin Scharte beschreiben es im Buch „Ausfallsichere Systeme" so:
Resilience Engineering ist der „Erhalt kritischer Funktionen, das Sicherstellen eines eleganten Abschmelzens – falls die kritische Funktionalität aufgrund der Schwere eines Störereignisses nicht aufrechterhalten werden kann – und die Unterstützung schneller Erholung von komplexen Systemen. Dazu werden generische Fähigkeiten sowie anpassbare und maßgeschneiderte technische Lösungen benötigt, die das System im Fall von kritischen Problemen und unerwarteten oder sogar noch nie dagewesenen Ereignissen schützen." Doch wie kann man sich das in der Praxis vorstellen?

Fähigkeiten resilienter Systeme

Beim Resilience Engineering betrachtet man die Funktionsweise der gesamten Organisation, die resilient agieren soll. Hierfür gibt es vier Fähigkeiten, die genauer betrachtet werden, um die Gesamtheit zu verstehen :

1. *How it responds* (wie es antwortet): Das System ist fähig, auf geplante und ungeplante Veränderungen oder Störungen zu reagieren.
2. *How it monitors* (wie es beobachtet): Das System weiß, wonach es sucht und was es beobachtet und reagiert darauf. Das bezieht sich sowohl auf das

Abb.1: Fähigkeiten resilienter Systeme

System selbst als auch auf die Umwelt.

3. *How it learns* (wie es lernt): Das System hat die Fähigkeit, aus Erfahrungen zu lernen und sich weiterzuentwickeln.
4. *How it anticipates* (wie es antizipiert): Das System beobachtet einerseits Entwicklungen der Umwelt und schätzt andererseits die Reaktionen der Umwelt auf Veränderungen an sich selbst ab.

Daraus entstehen vier Typen von Organisationen:

Systeme der ersten Art:
Systeme der ersten Art sind in der Lage, auf unerwartete Ereignisse so zu reagieren, dass ihr Fortbestehen gewährleistet ist. Hierbei geht es vor allem um die Fähigkeiten zu antworten und zu beobachten. Durch die Beobachtung erkennt das System, dass eine Handlung notwendig ist. Durch das Antworten ist es möglich, eine Reaktion zu implementieren. Wichtig ist, dass die beiden Fähigkeiten Hand in Hand gehen, da das System sonst nur reaktiv arbeitet, was in der heutigen Zeit meist nicht ausreichend ist.

Systeme der zweiten Art
Systeme der zweiten Art sind gewissermaßen lernfähig. Die Fähigkeit zu Antworten ist elementar – die Fähigkeit, Antworten nach Erfahrungen zu modifizieren, ist enorm hilfreich und macht das System zweiter Art deutlich stabiler als ein System erster Art. Es werden zwei unterschiedliche Typen von Lernen definiert: auf der einen Seite wird die Art des Lernens beschrieben, die absichert, dass das System die passenden Signale, Symbole und Zeichen beobachtet. Auf der anderen Seite geht es darum, die richtigen Reaktionen zum richtigen Zeitpunkt abzusetzen.

Systeme der dritten Art
Systeme der dritten Art agieren proaktiv. Sie scannen ihre Umwelt und passen sich an, bevor eine Störung passiert. Dazu muss das System in der Lage sein, Entwicklungen zu antizipieren oder vorauszusehen. Gleichzeitig bedeutet dies auch, ein gewisses Risiko einzugehen, weil man in die Vorlage geht und bei Nichteintreffen womöglich Ressourcen verschwendet hat. Ist ein System dazu in der Lage, alle vier Fähigkeiten einzusetzen, kann man es als resilient bezeichnen.

Systeme der vierten Art
Darüber hinaus gibt es noch die Systeme vierter Art, die im Bereich Resilience an der Spitze stehen. Diese sind nicht nur in der Lage einzuschätzen, was zwischen System und Umwelt passiert, sondern können auch noch einen Schritt weiter „denken" und Konsequenzen errechnen, wie die Umwelt auf Änderungen des Systems reagieren wird.

Fazit

Resilience Engineering beschreibt keine bestimmten Proportionen, wie diese vier Fähigkeiten vorhanden sein oder miteinander agieren sollen. Es gibt heute (noch) keinen allgemeingültigen Standard und möglicherweise wird es den auch künftig nicht geben, da die Notwendigkeit der Ausprägungen der Fähigkeiten von System zu System variiert.
Es scheint heute jedoch unerlässlich, komplexe technische Systeme zumindest einer Analyse zu unterziehen und diese vier Fähigkeiten einzuschätzen und in die Planung und Entwicklung miteinzubeziehen, damit sie je nach Umfeld und Notwendigkeit der Ausfallsicherheit weiterentwickelt werden (können). Üblicherweise wird bei der Planung und Entwicklung die meiste Zeit in die Response-Funktionen investiert, an zweiter Stelle steht häufig die Lernfähigkeit. Beobachtungsgabe und Antizipation kommen oftmals zu kurz. Hier lohnt es sich unter Berücksichtigung der Umwelt allerdings einen genauen Blick drauf zu werfen und sich die Frage zu stellen, ob die Prioritäten nach wie vor richtig gesetzt sind.■

Quellen und weiterführende Informationen:

http://erikhollnagel.com/ideas/resilience-engineering.html

http://erikhollnagel.com/ideas/resilience%20assessment%20grid.html

http://erikhollnagel.com/onewebmedia/RAG%20Outline%20V2.pdf

Wikipedia: Resilienz

MMag. Christina Eder ist Marketing Managerin bei SEQIS. Sie ist erste Ansprechpartnerin für Presse- und Marketinginformationen.

Von Drucksortengestaltung über Videodreh und -schnitt, klassischer Pressearbeit, Betreuung der SEQIS Onlinekanäle bis hin zur Organisation von Veranstaltungen übernimmt sie Marketing- und Kommunikationsagenden.

Besonders wichtig ist ihr die stetige Entwicklung von Medien, Tools und Konzepten.

Die nächsten Termine im Überblick:

14.11.2019
SEQIS „10 things“
Expertentreff:
„Testing“,
Tech Gate, Wien

Die Anmeldung ist ab sofort möglich!

Weitere Infos:
www.SEQIS.com

September	
1 So	
2 Mo	
3 Di	
4 Mi	
5 Do	
6 Fr	
7 Sa	
8 So	
9 Mo	
10 Di	
11 Mi	
12 Do	
13 Fr	
14 Sa	
15 So	
16 Mo	
17 Di	
18 Mi	
19 Do	SEQIS „10 things“
20 Fr	
21 Sa	
22 So	
23 Mo	Herbstanfang
24 Di	
25 Mi	
26 Do	
27 Fr	
28 Sa	
29 So	
30 Mo	

Oktober	
1 Di	
2 Mi	
3 Do	
4 Fr	
5 Sa	
6 So	
7 Mo	
8 Di	
9 Mi	
10 Do	
11 Fr	
12 Sa	
13 So	
14 Mo	
15 Di	
16 Mi	
17 Do	
18 Fr	
19 Sa	
20 So	
21 Mo	
22 Di	
23 Mi	
24 Do	
25 Fr	
26 Sa	Nationalfeiertag
27 So	
28 Mo	
29 Di	
30 Mi	
31 Do	

Haben Sie schon Ihre nächste Weiterbildung geplant?

www.SEQIS.com

November		
1	Fr	Allerheiligen
2	Sa	
3	So	
4	Mo	
5	Di	
6	Mi	
7	Do	
8	Fr	ANMELDESCHLUSS 10 things
9	Sa	
10	So	
11	Mo	
12	Di	
13	Mi	
14	Do	10 things SEQIS „10 things“
15	Fr	
16	Sa	
17	So	
18	Mo	
19	Di	
20	Mi	
21	Do	
22	Fr	
23	Sa	
24	So	
25	Mo	
26	Di	
27	Mi	
28	Do	
29	Fr	
30	Sa	

Über die SEQIS Expertentreffs „10 things I wished they'd told me!“

An Informationen mangelt es meist nicht – im Gegenteil, derer gibt es oft mehr als genug. Wichtiger denn je ist es, an die entscheidenden Informationen zu gelangen. Als Dienstleistungsunternehmen sind wir uns unserer Rolle als Informant bewusst und sprechen die an Software Test und IT Analyse Interessierten mit der Veranstaltungsreihe „10 things I wished they'd told me!“ konkret an.

Für all jene, die Software entwickeln, nutzen, beschaffen, in einem Betrieb für die Softwarequalitätssicherung zuständig oder als Requirements Engineer/Business Analyst tätig sind, haben wir eine passende Plattform geschaffen!

Bei unseren Expertentreffs erhalten Sie die Möglichkeit branchenbezogene Erfahrungen auszutauschen und wertvolle Tipps von den Profis abzustauben. Die Vortragenden bringen aktuelle Test- und IT Analyse-Themen auf jeweils 10 knackige Punkte und teilen mit Ihnen ihre Erfahrungen aus zahlreichen großen und komplexen IT Projekten.

Save-the-Date zu den „10 things“ 2019

Auch im Jahr 2019 laden wir Sie wieder ein, unsere vier kostenlosen Expertentreffs zu aktuellen IT Trendthemen zu besuchen.

Agile Transformation: 10 Erfolgsfaktoren aus der Praxis
Donnerstag, 14. März 2019 Done

Business Analyse: Kick-Down gleich von Start weg
Donnerstag, 06. Juni 2019 Done

Wie anpassungsfähig ist Ihre IT? Ein Einstieg in die Resilienz
Donnerstag, 19. September 2019 Done

Testen in Software Lifecycle Virtualization - die Zukunft des Testings?
Donnerstag, 14. November 2019

Melden Sie sich gleich an und sichern Sie sich Ihren Platz:
www.SEQIS.com/de/events-index

Mit 14 Techniken zur Cyber-Resilienz

von *Josef Falk*

Definition

Resilienz im allgemeinen Sinn ist die Widerstandsfähigkeit eines Systems gegenüber Störungen. Der Begriff ist auf alle Arten von Systemen anwendbar. In der Psychologie bezeichnet Resilienz die Fähigkeit einer Person, Krisen zu bewältigen und sie durch Rückgriff auf persönliche und sozial vermittelte Ressourcen als Anlass zu Entwicklungen zu nutzen. Der Begriff wird aber auch auf technische Systeme angewendet. In diesem Fall beschreibt er die Fähigkeit eines Systems, bei Störungen bzw. Teil-Ausfällen nicht vollständig zu versagen, sondern wesentliche Systemdienstleistungen aufrechtzuerhalten.

Wenn es um die Widerstandsfähigkeit von IT-Systemen geht, spricht man auch von Cyber-Resilienz. Die Publikation "Systems Security Engineering: Cyber Resiliency Considerations for the Engineering of Trustworthy Secure Systems" des National Institute of Standards and Technology des US Department of Commerce beschreibt ein Framework für die Herstellung von Cyber-Resilienz in IT-Systemen. Dieses Framework soll im Folgenden zusammenfassend dargestellt werden.

Die Aufgaben

Cyber-Resilienz in einem System erfüllt folgende Aufgaben:

- *Vorhersehen*: Durch das Sammeln und Auswerten von Informationen werden Angriffe und schädliche Zustände vor deren tatsächlichem Auftreten vorweggenommen.
- *Standhalten*: Ist ein Schadensfall eingetreten, soll das System diesem standhalten und in der Lage sein, seine Funktionalität zumindest in seinen Grundzügen weiter auszuführen.
- *Wiederherstellen*: Nach dem und während des Schadensfalles soll die volle Funktionsfähigkeit schnellstmöglich wieder hergestellt werden können.
- *Anpassen*: Das System soll seine Funktionalität an zu erwartende Bedrohungsbilder anpassen können.

Die Ziele

Um die genannten Aufgaben erfüllen zu können, werden folgende Ziele formuliert, die ein resilientes System erreichen soll:

- *Vermeiden*: ein möglicher Angriff bzw. ein schädlicher Zustand soll von vornherein verhindert werden.
- *Vorbereiten*: das System und die Organisation sollen auf einen Angriff bzw. einen schädlichen Zustand vorbereitet sein.
- *Fortsetzen*: die wesentlichen Funktionen können auch während eines Angriffs oder schädlichen Zustandes ausgeführt werden.
- *Beschränken*: der Schaden aus einem Angriff oder schädlichen Zustand soll minimiert werden.
- *Wiederherstellen*: die volle Funktionsfähigkeit soll nach einem Angriff oder schädlichen Zustand ehestmöglich wiederhergestellt werden.
- *Verstehen*: Pflege einer Dokumentation über die Funktionalität und die Systemzusammenhänge im Hinblick auf einen Angriff oder schädlichen Zustand.
- *Transformieren*: flexible Veränderung der Funktionalität, sodass das Risiko von Angriffen und schädlichen Zuständen minimiert wird.
- *Umgestalten*: Verändern der Architektur des Systems, um mit Angriffen und schädlichen Zuständen besser umgehen zu können.

Alle diese Punkte zielen darauf ab, ein System in die Lage zu versetzen, seine Aufgaben unter allen Umständen erfüllen zu können.

Cyber-Resilienz als Gestaltungsaufgabe

Ein System ist dann widerstandsfähig gegen eine Vielzahl von Gefahren, wenn diese Widerstandsfähigkeit von Beginn an bei der Gestaltung des Systems berücksichtigt wurde. Nachträgliche Maßnahmen können diese Widerstandsfähigkeit nur teilweise sicherstellen. Deshalb ist es Aufgabe der IT-Analyse, entsprechende Resilienz-Anforderungen von Anbeginn an in den zu gestaltenden IT-Systemen zu berücksichtigen.
Die Forderung nach Resilienz gehört zu den so genannten „nicht-funktionalen Anforderungen".
Diese beschreiben, anders als die funktionalen Anforderungen, nicht, welche Funktionen ein System ausführen können muss, sondern, auf welche Art und Weise es das tut.
Andere nicht-funktionale Anforderungen sind Performance, Wartbarkeit oder Erweiterbarkeit.
Da viele der erforderlichen Maßnahmen für nicht-funktionale Anforderungen, insbesondere auch für die Resilienz, technischen Charakter haben, ist die IT-Analyse auf die Zusammenarbeit mit technisch orientierten Berufsgruppen (z. B. Architekten) angewiesen.

14 Techniken für Cyber-Resilienz

Um die obengenannten Ziele erfüllen zu können und resiliente Systeme zu gestalten, nennt die Publikation "System Security Engineering, Cyber Resiliency Considerations for the

Copyright: Fotolia

Engineering of Trustworthy Secure Systems" des "National Institute of Standards and Technology" (NIST)3 14 Techniken, wie Cyber-Resilienz sichergestellt werden kann. Diese sind im Folgenden dargestellt und erläutert.

1. Flexible Reaktion

Zweck: Es wird die Möglichkeit geschaffen, auf zeitnahe und geeignete Art und Weise auf schädliche Bedingungen, Belastungen und Angriffe zu reagieren.
Beispiele: dynamisches Abschalten der Zugriffsmöglichkeiten; automatische Abschaltung des Systems; Load Balancing.

2. Analytische Überwachung

Zweck: Durch das Sammeln und die Auswertung von Betriebsdaten wird der Betrieb in die Lage versetzt, Gefährdungen frühzeitig zu entdecken und potenziellen oder schon eingetretenen Schaden festzustellen.
Beispiele: Verknüpfung von Daten aus verschiedenen Tools; Einsatz von Continuous Diagnostics and Mitigation (CDM) oder anderer Werkzeuge, die ein System auf Verletzbarkeit prüfen; Einsatz von Intrusion Detection Systemen (IDS).

3. Koordinierter Schutz

Zweck: Durch die Kombination mehrere Maßnahmen wird ein Angreifer gezwungen, mehrere Sicherungen zu überwinden, was die Schwierigkeit und die Kosten für den Gefährder erhöht und es wahrscheinlicher macht, dass ein Angriff rechtzeitig entdeckt werden kann.
Beispiele: Kombination von netzwerk- und host-basierten Angriffer-kennungssystemen; Einsatz von Defense-In-Depth-Konzepten.

4. Täuschung

Zweck: Einem potenziellen Angreifer werden falsche Tatsachen über das System vorgespiegelt bzw. es werden wesentliche Informationen verborgen, sodass es für ihn schwierig ist, die geeignete Vorgehensweise zu wählen.
Beispiele: Einsatz von Verschlüsselung in verschiedenen Stellen des Systems: Verschleierung des Datenverkehrs durch „Onion-Routing"; Desinformation durch Fragen in öffentlichen Foren, die auf falschen Informationen beruhen; Veränderung des Programmcodes, sodass der Quellcode nicht durch Reverse Engineering gewonnen werden kann (Obfuscation); Schaffung falscher Anmeldedaten.

5. Vielfalt

Zweck: Die Unterschiedlichkeit verschiedener Systemkomponenten führt dazu, dass nicht das gesamte System durch eine einzige Art von Fehler oder Angriff außer Gefecht gesetzt werden kann. Ein Angreifer muss unterschiedliche Strategien für unterschiedliche Ziele entwickeln, was dessen Zeitaufwand und Kosten erhöht.
Beispiele: Verwendung unterschiedlicher Betriebssysteme; Auslieferung mehrerer Protokollstandards; N-Versions-Programmierung; Verwendung unterschiedlicher Telekommunikations-Services; Beschäftigung unterschiedlicher Zulieferer.

6. Dynamische Positionierung

Zweck: Funktionalität und Systemkomponenten werden verteilt und dynamisch repositioniert. Durch die Verteilung wird die Möglichkeit geschaffen, die negativen Folgen von Nicht-Angriffs-Ereignissen (Feuer, Überflutung, usw.) zu minimieren.
Beispiele: Wechsel der Prozessor-Lokationen (z. B. Wechsel auf eine virtuelle Maschine auf einer anderen physischen Komponente); Wechsel der Speicherorte; Fragmentierung und Partitionierung von verteilten Datenbanken.

7. Dynamische Repräsentation

Zweck: Die (kritische und nicht-kritische) Funktionalität ist in Hinblick auf mögliche Bedrohungen dokumentiert, sodass im Ernstfall ein genaues Bild vorliegt.
Beispiele: Verfolgung vorhergesagter oder bevorstehender Naturkatastrophen; dynamisches Speichern von Vorfalls- und Bedrohungsdaten; Pflege einer laufenden Dokumentation über Funktionalitäten und Beziehungen zwischen Ressourcen und deren Status in Bezug auf Bedrohungen.

8. Nicht-Persistenz

Zweck: Ressourcen und Daten werden nur in dem Maße abgespeichert und zur Verfügung gestellt, wie sie unbedingt benötigt werden. Nicht vorhandene Komponenten und Daten können nicht angegriffen werden.
Beispiele: Löschen von Daten nach der Verarbeitung; Verlagern von Daten in Offline-Speicher; Beendigung von nicht mehr benötigten Sessions; Netzwerkverbindungen trennen, die nicht benötigt werden.

9. Restriktive Vergabe von Rechten

Zweck: Jeder Benutzer erhält nur die Rechte, die er für die Erfüllung seiner Aufgabe braucht. Dadurch wird das System nicht nur vor unbeabsichtigten Änderungen geschützt, sondern ein möglicher Angreifer wird auch in seinen Möglichkeiten eingeschränkt.
Beispiele: Vergabe der geringstmöglichen Rechte an Benutzer; zeit-

basierte Zugriffs-Beschränkungen; rollenbasierte- (RBAC) und attributbasierte (ABAC) Berechtigungskonzepte.

10. 10 Ausrichtung an der Funktionalität

Zweck: Die System-Ressourcen werden nach den Business-Funktionen ausgerichtet. Die Verbindungen zwischen unternehmenskritischen und nicht-kritischen Funktionen werden minimiert, sodass ein Ausfall der nicht-kritischen Funktionen keine Auswirkungen auf die unternehmenskritischen hat. Dadurch wird auch die Angriffsfläche für Bedrohungen minimiert. Ein Angriff auf eine nicht-kritische Funktion stellt keine oder nur eine geringe Bedrohung dar.
Beispiele: Whitelisting zur Verhinderung der Installation von nicht benötigten Applikationen; Verhinderung, dass privilegierte Konten für nicht privilegierte Funktionen verwendet werden; Outsourcing nicht-essenzieller Services; Konfiguration des Systems, so dass nur die wesentlichen Funktionen ausgeführt werden können.

11. Redundanz

Zweck: Es werden mehrere geschützte Instanzen kritischer Ressourcen bereitgehalten. Dadurch werden die Folgen des Verlusts an Information und Funktionalität klein gehalten. Die Wiederherstellungskosten nach den Folgen eines schädlichen Ereignisses werden minimiert.
Beispiele: Sicherung früherer Konfigurationseinstellungen; Vorhaltung von Ersatzteilen; Führen von Schatten-Datenbanken; Haltung eines oder mehrerer alternativer physischer Speicherorte; Führen eines redundanten Sekundärsystems.

12. Segmentierung

Zweck: Systemteile werden auf Basis von Kritikalität und Vertrauenswürdigkeit voneinander abgegrenzt. Dadurch sollen feindliche Aktivitäten, aber auch Nicht-Angriffs-Schädigungen (Feuer, Überflutung), auf einzelne Systemteile begrenzt werden. Die möglicherweise von Malware
betroffenen Systemteile werden von den anderen abgegrenzt.
Beispiele: Trennung von Komponenten auf der Basis von Geschäftsfunktionalität; Isolierung sicherheitsrelevanter Funktionalität von nicht-sicherheitsrelevanter; Verwendung von System-Partitionierung; Verwendung von Prozess-Isolierung.

13. Nachgewiesene Integrität

Zweck: Durch verschiedene Maßnahmen wird ermittelt, ob kritische Systemelemente korrumpiert wurden. So sollen Versuche eines Angreifers, schadhafte Daten, Software oder Hardware ins System einzubringen, aufgedeckt werden.
Beispiele: Verwendung automatisierter Tools für die Prüfung der Datenqualität; Verwendung von Sicherheitssiegeln; Integritätschecks von externen Systemen; fälschungssichere Schutzmaßnahmen.

14. Unvorhersehbarkeit

Zweck: Durch zufällige und unvorhersehbare Änderungen am System wird die Ungewissheit eines Angreifers über getroffene Schutzmaßnahmen erhöht.
Beispiele: Erforderliche Neu-Authentifizierung in zufälligen Intervallen; Ausführung von Routinetätigkeiten zu unterschiedlichen Tageszeiten; wechselnde Rollen und Zuständigkeiten.

Fazit

Ein System, das auch bei Bedrohungen von außen in der Lage ist, seine Grundfunktionen auszuführen, wird ein resilientes System genannt. Diese Resilienz entsteht nicht von alleine, sondern sie muss bewusst gestaltet werden. Die Publikation "System Security Engineering, Cyber Resiliency Considerations for the Engineering of Trustworthy Secure Systems" des "National Institute of Standards and Technology" (NIST)3 nennt 14 Techniken, deren Anwendung zu einem resilienten IT-System führen.■

Quellen und weiterführende Informationen:

https://de.wikipedia.org/wiki/Resilienz_(Psychologie)
https://de.wikipedia.org/wiki/Resilienz_(Ingenieurwissenschaften)
https://csrc.nist.gov/publications/detail/sp/800-160/vol-2/draft
[ROSS2018], Ron Ross, System Security Engineering, Cyper Resiliency Considerations for the Engineering of Trustworthy Secure Systems, National Institute of Standards and Technology, S. 8

Mag. Josef Falk ist IT Analytiker.

Seit dem Abschluss seines Studiums der Betriebswirtschaftslehre in Wien gestaltet er Lösungen in den unterschiedlichsten Fachbereichen – und ist dabei Mittler zwischen Fachbereich und IT-Entwicklung.

Besonderes Augenmerk legt er bei der Analyse auf den Innovationsgrad. Neben seiner Projekttätigkeit befasst er sich mit der Entwicklung der Business Analyse und ist aktuell Mitglied des Vorstandes des Austria Chapter des IIBA (International Institute of Business Analysis).

Resilienz in der IT durch Verwendung redundanter Systeme

von *Martin Brandhuber*

Begibt man sich auf die Suche nach dem Begriff der Resilienz (lat. Resilire, „zurückspringen, abprallen"), wird einem schnell klar, dass diese Fähigkeit, nach einem bedeutenden Ereignis wieder in den Ursprungszustand zurückzukehren, quasi jeder und jedem von uns bereits in irgendeiner Form bekannt ist. Er findet sich in der Energiewirtschaft (Ausfallsicherheit in der Energieversorgung), in der Psychologie (psychische Widerstandsfähigkeit), in Ökosystemen (z.B. Regeneration von Wäldern nach einem Brand) - oder eben auch in der IT. Die Fähigkeit, einen Dienst oder einen Service nach einem unangenehmen Zwischenfall möglichst rasch oder sogar dem Benutzer gegenüber unbemerkt wiederherzustellen kann sich in manchen Branchen als überlebenswichtig darstellen – man stelle sich nur etwa die Konsequenzen vor, wenn die AustroControl, welche den Flugverkehr über Österreich regelt, aufgrund eines massiven Hackerangriffs für Minuten oder gar Stunden außer Gefecht gesetzt werden würde. Die größte bislang bekannte DDos-Attacke (Distributed Denial of Service) gegen ein System fand am 28. Februar 2018 statt. Die populäre Entwicklerplattform GitHub wurde damals mit einem plötzlichen Datenansturm von 1,35 Terabytes/s bombardiert, was einer Menge von 126,9 Millionen übertragenen Datenpaketen pro Sekunde entspricht. Github wurde von diesem Angriff zwar nicht vollständig „auf dem falschen Fuß" erwischt, da bereits ein DDoS Protection Service im Hintergrund verfügbar war, die Magnitude des Angriffes sorgte aber dennoch für eine ca. 20-minütige Unterbrechung der Services.

Als Auslöser für einen Zusammenbruch eines Systems kann oft ein

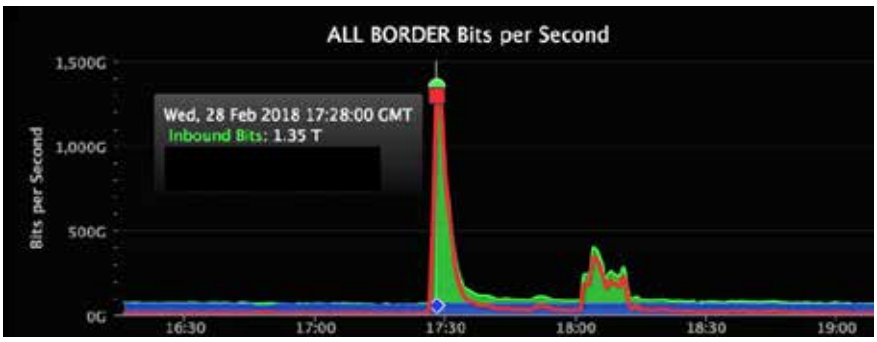


Abb. 1: Übertragene Bits pro Sekunde beim Angriff auf GitHub (© A10networks.com)

„Single Point of Failure" identifiziert werden – also ein Device in einer Netzwerkinfrastruktur oder eine Komponente, deren Ausfall eine Unterbrechung des Systems verursacht. Hat dieses System die Fähigkeit, im Falle eines Ausfalls einer Teilkomponente trotzdem weiter operativ zu bleiben, spricht man von Fehlertoleranz. Je nach Einsatzgebiet bieten sich verschiedene Ansätze an, um eine hohe Fehlertoleranz zu erreichen, etwa die Einrichtung einer unterbrechungsfreien Stromversorgung (USV), oder die Verwendung bzw. Zusammenführung mehrerer physischer Festplatten zu einem logischen Laufwerk, welches eine höhere Ausfallsicherheit bietet. Das Akronym RAID (Redundant Array of Independent Disks) ist diesbezüglich bereits seit 1987 ein Begriff, wenn es um die Ausfallsicherheit von Systemen geht, und in diesem Akronym findet sich bereits das Wörtchen „redundant". Redundanz bedeutet in diesem Fall die Verwendung von mehreren identen Systemen, welche trotz eines Ausfalls einen Dienst oder eine Funktionalität sicherstellen können. RAID betrifft aber nur physische Laufwerke, also Festplatten, zumeist sind aber komplette Webserver Ziel eines Angriffs - wie sieht es mit der Errichtung redundanter Serversysteme aus?

Eine Möglichkeit eines redundanten Serversystems sieht die parallele Verwendung von identen Servern vor. Dabei laufen die Anfragen über einen Switch an einen Load Balancer, welcher die Anfragen etwa nach wählbaren Algorithmen wie Round Robin oder Least Load First zwischen den Servern verteilt. Dabei wird berücksichtigt, welche Server verfügbar sind, und die Last demensprechend zwischen den Servern verteilt. Vorteile dieses

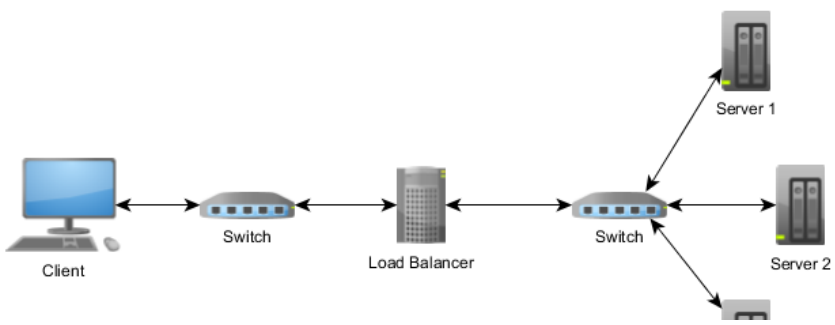


Abb. 2: Beispiel eines redundanten Serversystems

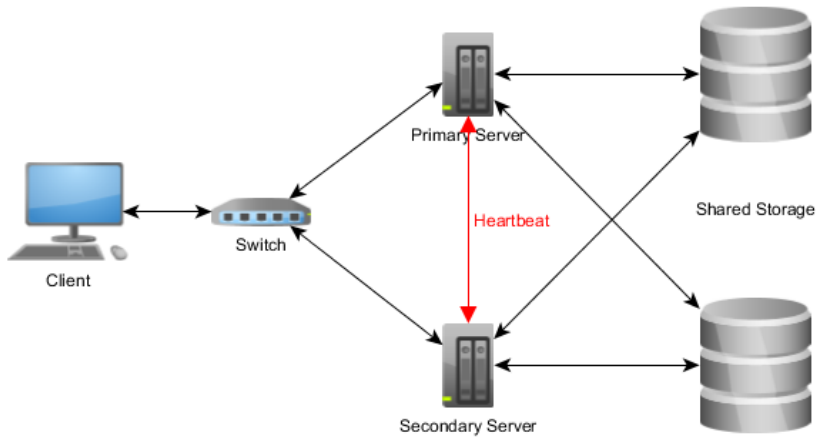


Abb. 3: Beispiel für Clustering

Systems sind eine hohe Ausfallsicherheit und gute Skalierung, da dem bestehenden System jederzeit beliebig viele idente Server hinzugefügt werden können.

Ein Cluster ähnelt dem zuvor beschriebenen redundanten System nur zum Teil. Der grundlegende Unterschied besteht darin, dass im Falle eines Clusters die zugehörigen Server miteinander kommunizieren, mittels des sogenannten „Heartbeat" (Herzschlag). Sie agieren zwar als voneinander unabhängige Knoten, benachrichtigen sich aber gegenseitig, ob sie verfügbar sind und die ihnen zugewiesenen Services noch ausführen können. Beim Clustering haben alle verbundenen Server Zugriff auf dieselben Datenbanken oder Programme, wodurch vermieden wird, dass beim Ausfall eines Servers auch dessen Daten verloren gehen - ein verfügbarer Server übernimmt einfach die Aufgaben seines ausgefallenen Kollegen.

Welche Art von redundantem Serversystem ist nun die „richtige"? Aufgrund der Vielzahl von möglichen Clusterarten (Hochverfügbarkeitscluster, Load-Balancing Cluster, High Performance Computing Cluster, Virtualization Cluster, ...) gibt es keine einfache Antwort auf diese Frage. Die Unternehmensgröße bzw. die Verfügbarkeit von Ressourcen, das vorgesehene Aufgabengebiet des Clusters und die Menge der zu bearbeitenden Daten sind nur einige Faktoren, welche die Wahl des geeigneten Systems beeinflussen werden. Eine umfassende Analyse und die Entwicklung von realistisch zu erwartenden Risikoszenarien bieten den ersten Schritt in eine redundante IT-Umgebung, welche Resilienz nicht nur propagiert sondern auch lebt - in einer Welt, in der Systeme zumindest doppelt oder gar mehrfach vorhanden sein müssen, um dem Benutzer ausfallsichere, störungsfreie Services bieten zu können.■

Martin Brandhuber, MSc. ist Consultant.

„Kundenzufriedenheit und Professionalität in der Planung und Durchführung des Testprozesses hat für mich höchste Priorität.

Testen ist für mich ein nicht wegzudenkender Bestandteil der Softwareentwicklung. Da die Anforderungen an den Vernetzungsgrad von Systemen laufend zunehmen, ist die frühe EInbindung von Testprozessen in der Softwareentwicklung unabdingbar. Nur so kann die Entwicklung kostengünstiger, qualitativ hochwertiger Software garantiert werden."


10 things
I wished they'd told me!
Sie haben unsere bisherigen Veranstaltungen verpasst?
Für Sie haben wir auf unserer Website alle Vorträge in chronologischer Reihenfolge zusammengefasst.
Sie finden dort auch alle Vortragsunterlagen zum Download:
www.SEQIS.com
Im November gibt's den nächsten spannenden Termin zum Thema „Testen in Software Lifecycle Virtualization".
Alle Details finden Sie im Kalender auf den Seiten 14-15 oder unter www.SEQIS.com!

Digitalisierung in Zeiten von VUCA: Mit Resilienz-Engineering fundiert Überleben sichern

von *Alexander Weichselberger*

Mit der zunehmenden Digitalisierung steigt die Abhängigkeit von IT Systemen. Gleichzeitig steigt das Risiko, dass diese IT Funktionen nicht bereitstehen, zumindest im gleichen Ausmaß. Dazu kommt, dass wir uns in einer VUCA Welt befinden – dh. alles volatiler, unbeständiger, komplexer und mehrdeutiger ist. Antizipiert man aus diesen Entwicklungen die potentiellen Risiken, Krisen und Katastrophen, dann ist Schluss mit Trial-and-Error und „Wird schon nix passieren!". Mit Resilienz-Engineering fundiert Überleben sichern.

Es pfeifen die sprichwörtlichen Spatzen vom Dach: Keine Frage, Digitalisierung ist heute die Antwort auf die Frage, wie wir künftig leben werden. Im privaten (Facebook, WhatsApp, Amazon, usw.) wie auch im unternehmerischen Umfeld (CRM, ERP, Cloud Dienste, Mobile Apps, usw.) ist die Digitalisierung, also der Einsatz von Informations- und Kommunikationstechnik um Abläufe effizienter zu machen und die Wirtschaftlichkeit zu steigern, der Ansatz für die Zukunft. Es gibt kaum noch Bereiche, die ohne Digitalisierungsmöglichkeiten und -perspektiven über weitere Wachstums- und Entwicklungsperspektiven nachdenken. Dabei sind Politik und alteingesessene Unternehmen – insbesondere im nicht-amerikanischen Umfeld – tendenziell überfordert („Wie das wirklich angehen?"), aber das ist eine andere Geschichte.

Ist Digitalisierung riskant?

Seit dem Millenniumswechsel haben wir IT architektonisch viele Neuerungen eingeführt, die unsere IT Systeme mehr und mehr exponieren. Wie in nebenstehender Tabelle zusammengestellt sind Vernetzung, Integration, Devices die miteinander sprechen und Datensammlungen die Quelle für eine Vielzahl von Risiken.

IT Pattern bzw. Architektur	Risiko
Mehr Outsourcing von IT Infrastruktur in die Cloud	***Down Time Risiko*** Mehr Komponenten und mehr an Abhängigkeiten von der Verfügbarkeit aller an der Wertschöpfung beteiligter Systeme erhöht die Anforderungen an Redundanzen und Flexibilität im gesamten Verbund. [1], [2]
Mehr Internet Connectivity industrieller Devices	***The Internet of (Insecure) Things*** IOT Funktionalität befindet sich nicht nur in Fahrzeugen. Nahezu in allen Lebensbereichen kann man IOT Devices sehen: Fernseher, Spielplattformen, Öfen, Waschmaschinen, Kühlschränken, Glühbirnen, Kinderspielzeug,... . Oft sind diese Geräte nicht nur am Netz, sondern auch noch mit einer Vielzahl von Sensoren ausgestattet: Kameras, Temperatursensoren, Geo Informationen, dynamische (Bewegungs)Sensoren. All diese Funktionen öffnen potentiellen Angreifern einen Vielzahl von Eingängen in unser privates und unternehmerisches Umfeld. [3]
Mehr Einsatz privater Devices im Unternehmensnetzwerk	***Bring Your Own Virus*** Mobile Versionen von Malware können Smartphones infizieren und das Unternehmensnetz auf mehreren Wegen exponieren. [4], [5], [6]
Mehr Online-Einsatz persönlicher Daten	***Social Hacking erleichtert*** Entgegen der von der EU verordneten Datenschutzgrundverordnung (DSGVO) werden immer mehr und mehr personenbezogene Daten online verarbeitet (Probleme im Kontext: Technischer Schutz, Minimumprinzip, usw.). Zusätzlich kommen zum potentiellen Datenverlust hier noch im besonderen Maße Sicherheitsgefährdungen dazu: Hat der Social Hacker viele personenbezogene Daten, ist ein Angriff deutlich einfacher. Der Mensch ist und bleibt einfach die Schwachstelle [7]

Ist also Digitalisierung riskant? Meiner Meinung nach: Ja. Es kommen einfach neue transparente bzw. vernetzte Möglichkeiten ins Spiel, die wir z.B. in einer „offline, old fashioned Silo Lösung" nicht hätten. Zum Guten,

Ist also Digitalisierung riskant? Meiner Meinung nach: Ja. Es kommen einfach neue transparente bzw. vernetzte Möglichkeiten ins Spiel, die wir z.B. in einer „offline, old fashioned Silo Lösung" nicht hätten. Zum Guten, aber auch zum Bösen.

VUCA – Problem und Lösung

Kennen Sie „bullshit-bingo"? Im Kern geht es darum, eine vorgegebene Sammlung von Buzzwords in z.B. einer Präsentation oder Fachartikel zu identifizieren. Hat man alle Wörter auf der Sammlung abgehakt, hat man gewonnen und darf sich durch ein lautstarkes „Bullshit" melden. Stellen wir mal eine Sammlung solcher Buzzwords zusammen; ich hätte hier: „Digitalisierung", „Scrum", „Kanban", „agiles Vorgehen", „Industry 4.0", „Disruption" und „Demokratisierung von Führung". Kaum ein aktueller Business Talk kommt ohne diese Begriffe aus. Hier geht es allerdings nicht nur um Buzzwords, sondern um die Basis für grundlegende Veränderungen, die durch die technischen Möglichkeiten gestützt bzw. auch gepushed werden.

Diese Wandlungsfähigkeit wird durch das Akronym VUCA „Volatility", „Uncertainty", „Complexity" und „Ambiguity" gut beschrieben. Und ein Haupttreiber dafür ist die Digitalisierung.

Ursprünglich ist VUCA ein Ergebnis aus einer Analyse aus dem militärischen Umfeld: Die Kontrahenten in den letzten Kriegen in Afghanistan und Irak, und dem damit einhergehenden Terrorismus, haben sich nicht mehr am Schlachtfeld gegenübergestellt, sondern haben in der Auseinandersetzung auf andere Strategien gesetzt. Dadurch mussten Gegenmaßnahmen entwickelt werden, wie mit dieser Art von Auseinandersetzung damit umgegangen werden kann.

> ***Resilienz***
>
> *... ist allgemein gesprochen ein dynamischer Anpassungsprozess. Es bedeutet, sich erfolgreich an Herausforderungen anzupassen und diese zu meistern. Dabei darf man keinen Schaden nehmen und sollte – bestenfalls – aufgrund dieser Erfahrungen wachsen und reifen.*

Zusammengefasst: Das Umfeld ist VUCA – und auch das wird sich wohl steigern. Die technische Exposition wird durch die fundamentalen Entscheidungen der Digitalisierung noch wesentlich verschärft. Dagegenhalten kann man mit einer Vision, Verstehen und Klarheit auf der Basis von Ausschnitten der Gesamtsituation sowie agilem Vorgehen. Und mit Resilienz.

Was ist nun Resilienz – und wie kann das helfen?

Die Basisüberlegungen zur Resilienz kommen aus der Humanmedizin und beschäftigen sich mit der Leistungs-

Problem	Militärische Analyse und „-->" Maßnahmen	Gegenstrategie
Volatilität („volatility")	Einzelne, schnelle kriegerische Aktionen --> Dezentralsierung: eigene Entscheidungszyklen müssen schneller sein als die gegnerischen Entscheidungsprozesse	***Vision („vision")***
Unsicherheit („uncertainty")	häufige Unvorhersehbarkeit von Angriffen --> Handeln auch ohne vollständigen Überblick über die Lage	***Verstehen („understanding")***
Komplexität („complexity")	Kommandostrukturen und Koordinationsprinzipien sind nicht erkennbar. Das gegnerische Netzwerk besitzt mit vielen Kommunikationsbeziehungen --> Implementierung selbststeuernder Teams, ex-ante vereinbarte Prioritäten	***Klarheit („clarity")***
Mehrdeutigkeit („ambiguity")	Mehrdeutig: Es kann sich um kriegerische oder friedliche Akteure handeln --> Schwache Informationen dezentrale Entscheidungen werden besser vernetzt.	***Agilität („agility")***

[8] Frei nach Ulrich Lenz, „Coaching im Kontext der VUCA-Welt: Der Umbruch steht bevor"

fähigkeit von Menschen in Krisenfällen.

Die Mechanismen zwischen Menschen und IT Systemen sind diesbezüglich vergleichbar. Es stellen sich generell die gleichen Fragen:

- Was ist die Ausgangsposition vor der Krise – und welche Abwehrmechanismen sind bereits vorhanden/etabliert?
- Wie tiefgreifend wirkt die Krise?
- Wie rasch kann man diese Krise überwinden, wie schnell läuft die Wiederherstellung der vollen Leistung?
- Gibt es eine Steigerung gegenüber der vorhergehenden Leistungsfähigkeit?

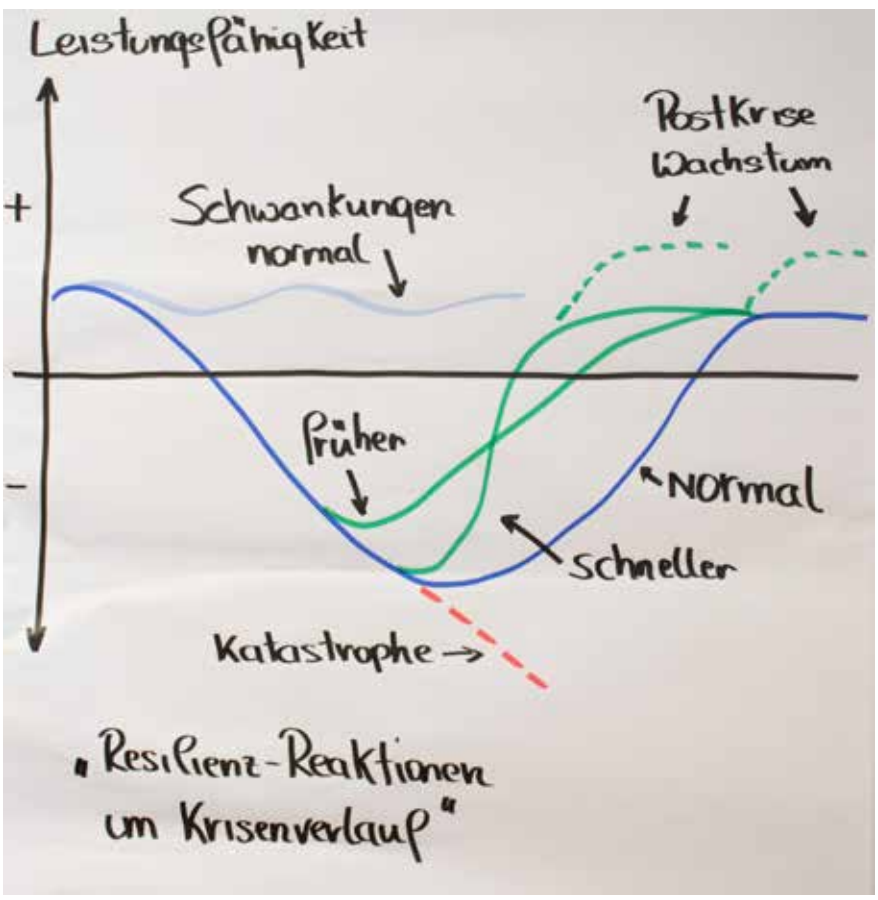
Relevant sind mehrere Faktoren, die eine Beurteilung beeinflussen: Zwar kann man eine Krise generell immer nur im Nachhinein betrachten, aber es ist relevant, ob man die Situation vor, in oder nach der Krise betrachtet. Darüber hinaus ist es auch schwieriger, wenn mehrere Krisen parallel auf das System (oder den Menschen) wirken. Wesentlich ist auch, wie das Umfeld (= soziales Umfeld, Unternehmen) sich hinsichtlich gelebter Führung und Wertehaltung real verhält – können / dürfen / müssen wir aus Fehlern lernen, oder ist eine Verdeckung von Fehlern taktisch. Einen großen Einfluss hat natürlich auch die Krisendauer: Besteht diese kurzfristig, oder dauerhaft?

Krise – oder doch Katastrophe?

Resilienzbetrachtungen beziehen sich letztendlich immer auf Krisen. Eine Krise („entscheidende Wendung") ist – lt. Duden.de – definiert als eine schwierige Situation, die Höhe und Wendepunkte einer gefährlichen Entwicklung darstellt. Da es sich um einen Wendepunkt handelt, kann diese Krise immer erst ex-post, dh. wenn sie abgewandt oder beendet wurden, als solche festgestellt werden.
Nimmt die Entwicklung einen dauerhaften negativen Verlauf, so spricht man von einer Katastrophe („Niedergang").
Allen Krisen gleich stehen folgende Herausforderungen gegenüber: Welche Basismerkmale sind etabliert, die der potentiellen Krisen entgegenhalten? Wie Krisen frühzeitig erkennen? Welche Standards, wie Notfallpläne, Checklisten und Interventionscenter (= Hilfe von außen), sind etabliert, um Krisen rasch zu beenden? Welche Standards sind vorgesehen, um aus Krisen gestärkt / verbessert einen höheren Level an Leistungsbereitstellung zu erreichen?

Resilienz-Engineering

Im Kern geht's darum, im Standard der Leistungsfähigkeit zu bleiben („Vorhersehen", „Standhalten") und, wenn Krisen notwendig sind (Krise = entscheidende Wendung), gilt es die



VUCA	Ansatz	Ziel	Beispiele für technische Umsetzung
Vision *...Ziel und angestrebter Zustand zur Erfüllung der Mission*	Kill Kill-Path	Maximale Hürden für Hacker und Fehler; Vermeiden, dass Schäden entstehen können, weil Angriffspunkte zu offensichtlich sind oder Fehlhandlung nicht aktiv vermieden werden	IDS, IPS; mehrstufige Securityzonen; 4- Augen-Prinzip; programmierte Konfiguration und automatisierte Systemkonfiguration
	Minimumprinzip bei Datenspeicherung und Connectivity	Potentielle Ausbeute und Angriffsmöglichkeiten bewusst gering halten	Datenoperationen im Arbeitsspeicher, Verbindungen nur temporär
Understanding *...frühzeitig verstehen, was los ist*	Vorhersagen und automatisierte Empfehlungen	Prognosemodelle für KPIs, nahe Echtzeitanalyse Rasche Identifikation einer Krise	Betriebsüberwachung / Monitoring, Performance Profiling, IDS, IPS
	Integritätsnachweise	Ermittlung, ob kritische Systemelemente korrumpiert wurden	Automatisierter Test der Datenqualität; Integritätschecks externer Systeme
Clarity *...intern klares Verständnis, was zu tun ist; nach extern: Verschleierung und Verwirrung*	Don't let me think	Klare Vorgaben, Anweisungen und Checklisten für Development, Deployment und Operations	Entwickler- und Betriebshandbücher, statische und dynamische (Code) Analyse
	Dynamische Konfiguration, volatile Betriebsabläufe, aktive Falschinformationen (Antipatterns)	Ziel wird schwierige greifbar	Verschlüsselung Daten und -verkehr; Desinformation der Communities System wird scheinbar zufällig und unvorhersehbar verändert
Agility *...aktiv Krisen meistern können*	Flexibilität	Rasch und geeignet auf Krisen reagieren	Überlastungsschutz, Load Balancing
	Segmentierung und Redundanzen	Einschränkung von Schäden auf einzelne Systemteile; Vorhaltung ausreichender Ressourcen in den jeweiligen Szenarien	System-Partitioning, Prozess-Isolierung auf unterschiedliche Systemteile; distributed datacenter

Leistungsfähigkeit früher und schneller wiederherzustellen („Wiederherstellen") und dabei im Idealfall mit einem Lerneffekt die Leistungsbereitstellung noch zu steigern („verbessern").

Nun, wie am Besten angehen?

Es hängt wie so oft am richtigen Mix – hier mal ein Ausschnitt aus einem möglichen Lösungs-VUCA („vision", „understanding", „clarity" & „agility"):

Diese Zusammenstellung ist natürlich nur ein Auszug hinsichtlich möglicher und sinnvoller Aktivitäten. U.a. wurden nur Teile des „Systems Security Engineering: Cyber Resiliency Considerations for the Engineering of Trustworthy Secure Systems" gelistet, andere essentielle Bereiche hinsichtlich Kriseninterventionscenter, interne und externe Mitarbeiter als Akteure im jeweiligen Systemkontext usw. gar nicht erwähnt.

Im Wesentlichen müssen wir die richtigen Maßnahmen richtig priorisieren und reihen, instrumentalisieren und deren Zusammenspiel testen. Öfters auch im Produktionssystem einzelne Krisen bewusst provozieren um den Status Quo des Systems, auch über die Zeit, immer wieder kennenzulernen und uns Platz einzuräumen aus bestehenden Schwächen zu lernen und Resilienz zu entwickeln.■

Quellenverzeichnis:

[1] 16.8.2013 – Ausfall eines einzigen Unternehmens: Google. Und damit rd. 40% des globalen Internet Verkehrs

- Tim Worstall, "Analyzing Friday's Google Outage", https://www.forbes.com/sites/timworstall/2013/08/19/analysing-fridays-google-outage/#72fbcd0f6ede , 14.7.2019, 09:17
- Simon Tabor, "Google's downtime caused a 40% drop in global traffic", https://engineering.gosquared.com/googles-downtime-40-drop-in-traffic, 14.7.2019, 08:15

[2] Oktober 2013 – Ausfall von Microsoft's Azure Plattform – zwei mal in zwölf Monaten

- https://www.theregister.co.uk/2013/10/30/windows_azure_global_fail/, 12.7.2019, 20:05

[3] 2013 auf der Defcon Hacker's conference – Nachweis, wie leicht die Lenkung oder Bremsen eines Toyota Prius übernommen werden kann.

- https://www.computerworld.com/article/2484616/researchers-reveal-methods-behind-car-hack-at-defcon.html, 1.7.2019, 12:05

[4] Ende 2012 – Ciscos mobile BYOD strategy: "Mobile at Cisco is now BYOD, period." Brett Belding, senior manager Cisco IT Mobility Service.

- https://blogs.cisco.com/security/standing-up-to-threats-the-cisco-2013-annual-security-report-and-security-intelligence-operations?dtid=osscdc000283 , 26.6.2019, 15:05

[5] Die mobile Version von FinSpy/FinFisher loggt für Angreifer eingehende und ausgehende Anrufe.

- https://blogs.cisco.com/security/byod-many-call-it-bring-your-own-malware-byom, 26.6.2019, 15:00

[6] Mobile Malware kann auch in Form von SMS Nachrichten kommen: Der User wird über die Zustellung von DHL Paketen informiert, klickt auf den Tracker Link... .

- https://blogs.cisco.com/security/byod-many-call-it-bring-your-own-malware-byom, 26.6.2019, 15:00

[7] Social Engineering – der Mitarbeiter als Angriffsziel

- Quelle https://www.wko.at/Content.Node/blogs/it-safe/Social-Engineering.html, 17.3.2019, 12:03

[8] Coaching im Kontext der VUCA-Welt: Der Umbruch steht bevor, Ulrich Lenz, © Springer Fachmedien Wiesbaden GmbH; J. Heller (Hrsg.), Resilienz für die VUCA Welt, https://doi.org/10.1007/978-3-658 -21044-1_4

Mag. (FH) Alexander Weichselberger hat seine Einsatzschwerpunkte in den Bereichen Systemanalyse, Software Test, Koordination und Management von exponierten Großprojekten und kann auf jahrelange Erfahrung zurückblicken.

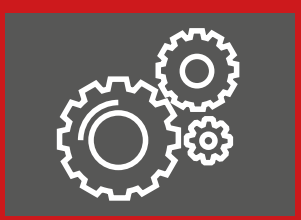
Dieses Wissen gibt er gerne in Form von Coachings, Methodentrainings und Fachvorträgen weiter.

SEQIS ist der führende österreichische Anbieter in den Spezialbereichen
IT Analyse, Software Test und Projektmanagement.
Beratung, Verstärkung und Ausbildung:
Ihr Partner für hochwertige IT-Qualitätssicherung.

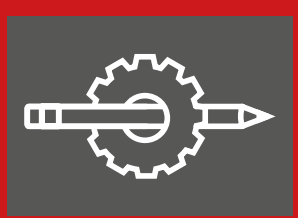
IT ANALYSE
Notwendige Änderungen analysieren und IT-gerecht aufbereiten

CODING
Guten Code schreiben und schlechten Code überarbeiten

TESTING
Probleme durch methodischen Soll-Ist Vergleich erkennen



RELEASE & OPERATE
Reibungsloser Go Live und Betrieb der IT-Lösungen



DEVOPS
Neuerungen abgestimmt mit Entwicklung und Betrieb live setzen

METHODOLOGY & TOOLS
Vorgehensweisen optimieren und auf die richtigen Tools setzen

TRAINING & WORKSHOPS
Mitarbeiter Know-how stärken - standardisiert oder maßgeschneidert

PROJEKT-MANAGEMENT
Verantwortung übernehmen, zielorientiert und pragmatisch agieren

Analysis. Test. Management. Better IT Results.
www.SEQIS.com