Seminar
Berühmt berüchtigte Softwarefehler

USS Yorktown

René Lotz
Matrikel-Nummer: 200210224
eMail: renelotz@uni-koblenz.de
Seminarleiter: Bernhard Beckert

Sommersemester 2003

Inhaltsverzeichnis

Zusammenfassung

Diese Arbeit ist ein Teil des Seminars 'Berühmt - berüchtigte Software - Fehler' , das im Sommersemester 2003 an der Universität Koblenz-Landau unter der Leitung von Herrn Dr. Bernhard Beckert veranstaltet wurde. Ziel des Seminars war es berühmte Softwarefehler der Vergangenheit zu analysieren und zu überprüfen, wie man solche Fehler vermeiden kann. Im Rahmen dieses Seminars ist auch diese Arbeit über die USS Yorktown, ein Kriegsschiff der US NAVY, entstanden. Aufgrund eines Softwarefehlers fiel 1997 whärend eines Routine–Manövers das gesamte Antriebssystem der Yorktown aus und sie war für mehrere Stunden manövrierunfähig.

Im Folgenden wird beschrieben, wie es zu dem Fehler kam, wer den Fehler hätte verhindern können und wie man solche Fehler im Allgemeinen vermeiden kann. Des weiteren wird auch eine Diskussion darüber geführt, wie man Computertechnik in Zukunft sinnvoll an Bord eines Kriegsschiffes einsetzen kann.

1 Einleitung: Das „Smart Ship Concept“

Mit dem sogenannten „Smart Ship Concept“ versucht die US Navy die Anzahl der Matrosen auf den Kriegsschiffen deutlich zu vermindern, um Kosten zu sparen. Zu diesem Zweck werden in den Schiffen moderne Computer installiert und vernetzt, die die Matrosen in ihrer Arbeit unterstützen sollen. Somit können die Kosten für die Arbeiten, die die neuen Systeme leisten, veringert werden. Diese Systeme werden hauptsächlich in Bereichen

- der Wartung,
- der Überwachung und
- der Steuerung

eingesetzt und vereinfachen dadurch die Arbeit der Matrosen in diesen Bereichen enorm. Dies führt dazu, dass weniger Personal für solche Aufgaben benötigt wird und senkt somit auch die Kosten. Außerdem bringen solche neuen Steuerungsarchitekturen auch weitere Vorteile mit sich, wie z.B. einen besseren Überblick über den Gesamtzustand des Schiffes, vereinfachte Steuerung und damit verbundene Vorteile in einem möglichen Kampfeinsatz.
Es werden im Rahmen dieses Konzeptes neue Schiffe nach diesen Vorgaben gebaut, aber auch ältere Kriegsschiffe mit Computertechnik nachgerüstet. Eines dieser umgebauten Schiffe ist die USS Yorktown.

2 Allgemeine Informationen über die USS Yorktown

Die USS Yorktown ist ein Kriegsschiff der US NAVY vom Typ „Guided Missile Cruiser“. Sie wurde am 17. Januar 1983 zum ersten mal zu Wasser gelassen und Mitte der Neunziger im Rahmen des „Smart Ship Concepts“ umgebaut. Die Haupt–Aufgaben der USS Yorktown ist die Unterstützung und der Schutz von Flugzeugträgern.

2.1 Technische Daten des Schiffes

Um eine Vorstellung von den Dimensionen und der Ausstattung der USS Yorktown zu bekommen, dienen die folgenden technischen Daten
(siehe auch [6] [1]):

- Abmessungen: 16,8m breit, 173m lang
- Wasserverdrängung bei voller Ladung: 9600 Tonnen
- Antrieb: 4 Motoren
- Geschwindigkeit: 30 Knoten
- Besatzung: 24 Offiziere und 340 Matrosen
- Kosten: ca. 1 Milliarde US $
- auf der Yorktown ist ein Hubschrauberplatz.
- Bewaffnung:
  - 2 Raktenwerfer für Standardraketen und ASROC
  - Torpedos (Mk 46)
  - "Harpoon" Raketenwerfer (Lenkraketen)
  - 2 Geschütze (Mk45)

Die USS Yorktown ist also zwar kein Flugzeugträger oder Zerstörer, aber dennoch ein relativ großes Schiff mit mehr als 300 Mann Besatzung.

2.2 Umbau im Rahmen des „Smart Ship Concepts"

1996 wurde die USS Yorktown zu einem sogenannten „Smart Ship" umgebaut. D.h. man hat in dem Schiff 27 Terminals installiert, die je aus einem Rechner mit „Dual Pentium Pro 200MHz" Prozessoren bestanden. Diese Terminals waren durch ein schnelles Glasfasernetzwerk miteinander verbunden, so dass das Schiff in einem möglichen Notfall von irgendeinem der 27 Terminals aus gesteuert werden kann. Des weiteren kann man davon ausgehen, dass das System einen großen Hauptspeicher verwendet (zumindestens einen Adressbereich) [1].
Als Betriebssystem wurde Microsoft Windows NT 4.0 verwendet. (vgl. [9])

[1] Dies kann man aus den Beschreibungen des Softwarefehlers folgern

2.3 Aufgabenbereiche der Computertechnik und deren Vorteile

Die Aufgabenbereiche der neuen Systeme waren Funktionen der Überwachung und Steuerung, die die Haupt–Aufgaben des Brückenpersonals sind. Unter anderem war das System auch für die Antriebssteuerung zuständig. Durch spezielle Sensorabfragen konnte auch die Schadenskontrolle stark vereinfacht werden und die Wartung wurde auch einfacher, da man aufgrund von Sensorinformationen Fehler direkt lokalisieren und beheben konnte.
Die Vorteile, die durch die neuen Systeme entstanden sind waren enorm:

- kleinere Besatzung
  Aufgrund der Vereinfachung der Arbeiten in den Bereichen, die von den Computer–Systemen unterstützt werden, konnte man ca. 10% der Besatzung auf der Yorktown einsparen.
  z.B. wurden die Brückenfunktionen so stark vereinfacht, dass nach dem Umbau nur 3 Offiziere die gleiche Arbeit leisten konnten, wie vor dem Umbau 13 Offiziere.

- geringere Kosten
  Nicht zuletzt aufgrund der Einsparung beim Personal konnten die laufenden Kosten des Schiffes um ca. 2,8 Millionen US$ gesenkt werden.

3 Der Softwarefehler

3.1 Was ist passiert? (Kontext des Softwarefehlers)

Am 21. September 1997 fuhr die USS Yorktown am Kap Charles vor Norfolk ein Routine-Übungsmanöver. Aufgrund eines Softwarefehlers ist das gesamte Antriebssystem ausgefallen und die Yorktown war manövrierunfähig. Einige Quellen berichten, dass die Yorktown auch nach knapp drei Stunden nicht wieder fahrbereit gemacht werden konnte und somit musste sie in den nächsten Hafen geschleppt werden. Die meisten Quellen und auch die offiziellen Berichte der Navy sagen aber aus, dass die Yorktown nach ca. 2 Stunden 45 Minuten den Antrieb soweit repariert hatte, dass sie aus eigener Kraft den nächsten Hafen, die Naval Base in Norfolk, Va. erreichen konnte. Dort wurde sie dann drei Tage repariert.
Wäre ein solcher Fehler während eines echten Kampfeinsatzes aufgetreten, hätten die Folgen fatal sein können. Also stellt sich die Frage, wie es eigentlich auf einem Kriegsschiff zu solch einem großen Fehler kommen kann. Dazu müssen wir das Szenario des Fehlers ein wenig genauer betrachten.

3.2 Wie ist es passiert? (Szenario des Softwarefehlers)

Alles begann damit, dass ein Überwachungsprogramm ein Ventil als geöffnet anzeigte, obwohl es geschlossen war. Um einen solchen Fehler zu beheben war es auf der Yorktown üblich, dass ein Offizier einige Werte in der Datenbank des Systems direkt ändert, um den fehlerhaften Sensor / Eintrag zu umgehen bzw. zu korrigieren. Die Änderungen werden schriftlich festgehalten, um sie ggf. wieder rückgängig zu machen. Dieses direkte Ändern der Werte war allerdings von der Software so nicht vorgesehen.
Im Rahmen dieser Versuche ändert der Offizier einen bestimmten Eintrag auf den Wert „0“, was vorher noch nie getan wurde.
Die Software prüfte den geänderten Eintrag nicht und verwendete den Eintrag als Divisor in einer Division. Daraufhin kam es angeblich sogar zu mehrfachen „Divide by zero“–Errors , welche von der Software nicht korrekt abgefangen worden sind. Aus diesem Grund füllte sich der temporäre Speicher des Systems und als dieser voll wurde, sind einfach Daten im angrenzenden Speicherbereich überschrieben worden. In diesem Fall der Hauptspeicher des Antriebssystems.
In Folge dessen fiel das Antriebssystem komplett aus und das Netzwerk des Schiffes brach zusammen.

4 Schuldfrage

Nun stellt sich die Frage, wer die Schuld für diesen Unfall übernehmen muss. Es gibt verschiedene Meinungen:

- „menschliches Versagen“ (des Offiziers)
- Betriebssystem (MS Windows NT4.0)
- Applikation des Schiffes

Im Folgenden werden diese unterschiedlichen Meinungen dargestellt und näher erörtert.

4.1 menschliches Versagen

Offizielle Vertreter der NAVY haben den Vorfall als „menschliches Versagen“ dargestellt. (siehe [9]) Dies wird dadurch begründet, dass der diensthabende Offizier die Werte direkt in der Datenbank geändert hat, obwohl dies nicht in der Software vorgesehen ist. Außerdem ist die Behebung von Fehlern durch „Trial–and–Error“–Prinzip auch nicht tolerierbar.

Allerdings gibt es auch einiges, was gegen „menschliches Versagen“ spricht. Der Offizier konnte offensichtlich ohne größere Probleme die Werte direkt in der Datenbank ändern und somit sollte die Software alle Werte, die in solch einer Art geändert werden können vor der Verwendung auf Korrektheit überprüfen. Dies kann nicht die Aufgabe des Offiziers sein, da er nicht wissen kann, wie die Einträge verwendet werden.
Gerade auf Kriegsschiffen können während Kampfeinsätzen extreme Stresssituationen auftreten. Die Systeme müssen daher so konstruiert sein, dass jegliche Bedienfehler eines Offiziers vor dem eintreten eines Fehlers abgefangen werden.
Kritiker behaupten auch, dass, wenn man nur weit genug zurück geht es sich immer um menschliches Versagen handelt, egal um welchen Fehler es sich auch handelt (im Bezug auf Software–Systeme). Da die Fehler dann bei der Entwicklung der Software entstanden sind, welche auch von Menschen durchgeführt wird.

Aus diesen Gründen kann man die Schuld unmöglich dem Offizier geben und man kann daher den Vorfall auch nicht als „menschliches Versagen“ bezeichnen.

4.2 Betriebssystem (MS Windows NT4.0)

Einige Kritiker geben dem Betriebssystem der USS Yorktown die Schuld an dem Vorfall. Sie sind der Meinung, dass Windows NT den Fehler hätte abfangen müssen, bevor er sich ausbreitet und alle Systeme lahmlegt. (siehe auch [2] [3])

Gil Young, ein Netzwerkingenieur sagt zu dem Vorfall:

„Egal welches Betriebssystem, welchen Computer, welche Anwendung ich benutze – ich sollte eine Null eingeben können, ohne dass der Computer abstürzt.“ ([7])

Ron Redman, ein stellvertretender technischer Leiter bei der US Marine sagte:

„Unix ist das bessere System für die Kontrolle von Ausrüstung und Maschinen, NT hingegen für Datentransfer. NT ist nicht ganz ausgereift, es gab **einige Ausfälle** wegen des Systems.“ ([7])
(Hinweis auf frühere Ausfälle auch bei [4].)

Trotzdem übernimmt Microsoft keine Verantwortung für den Vorfall. Dies begründen die Redmonder damit, dass die Vermeidung solcher Fehler die Aufgabe der Programmierer und der Systemadministratoren der USS Yorktown wäre. (vgl. [7])
Ein Betriebssystem kann zwar einen „Divide by zero“–Error verhindern und einen Fehlercode an die Software schicken, aber die weitere Behandlung des Fehlers muss die Software selbst übernehmen, da das Betriebssystem den Kontext, in dem der Fehler aufgetreten ist, nicht kennt.
Außerdem ist es auch Aufgabe der Software zu verhindern, dass sich ein Fehler im gesamten Netzwerk des Schiffes ausbreitet und dieses lahmlegt.
Den Überlauf des temporären Speichers in den Hauptspeicher des Antriebssystems ist auch kein Fehler des Betriebssystems, da bei einer ausreichenden Trennung der Systeme in der Software, das Schreiben in dem Speicherbereich eines anderen Systems (Prozesses) verhindert worden wäre.

Die logische Schlussfolgerung ist daher, dass die Schuld bei der Applikation des Schiffes liegt.

4.3 Applikation

Es gibt mehrere Gründe, warum die Hauptschuld des Vorfalls bei der Applikation liegt:

- **keine Überprüfung der Eingabewerte**
  Die möglichen Einfabewerte hätten auf Korrektheit überprüft werden müssen, d.h. es muss an allen Stellen, wo ein User Eingaben machen kann, geprüft werden, ob die Eingabe korrekt ist, bzw. korrekt verwendet werden kann.

- **Einträge nicht vor Zugriff geschützt**
  Wenn es nicht vorgesehen ist, dass der User spezielle Einträge ändern kann, dann sollten diese Einträge auch vor Zugriffen durch den User geschützt werden.

- **schlechte Wartung**
  Bei einer vernünftigen Wartung wäre den Software-Entwicklern aufgefallen, dass die Offiziere die Software in einer Weise verwenden, wie sie eigentlich nicht vorgesehen war. Man hätte dies dann korrigiert, indem man entweder die Eingabewerte prüft oder sie vor Zugriff durch den User schützt.

- **schlechte Trennung der Systemkomponenten**
  Bei einer besseren Trennung der Systemkomponenten wäre es nie zu

einem Speicherüberlauf in den Speicherbereich des Antriebssystems gekommen.

- **Ausbreitung des Fehlers im gesamten Netzwerk**
  Es hätte verhindert werden müssen, dass aufgrund eines Fehler in einem Teil–System das gesamte Netzwerk des Schiffes zusammenbricht.

Dies sind alles Fehler der Applikation, da sie weder den Usern noch dem Betriebssystem zuzuordnen sind, wie es in den vorigen Abschnitten erklärt wurde.

Es stellt sich nur noch die Frage, wer bei der Software–Entwicklung die Schuld an dem Auftreten des Fehlers hat.

4.3.1 Management

Der wichtigste Faktor für den Fehler in der Applikation war, dass es offensichtlich zuwenige Software–Entwickler gab. Dies führte zu Versäumnissen bei

- der Planung
- der Implementierung
  und vor allem beim
- Testen bzw. Verifizieren
- Warten

der Software.

Die Personalplanung ist die Aufgabe des Managements. Aus diesem Grunde liegt die Hauptschuld des Vorfalls beim Management, welches das Umbauprojekt und die Software–Entwicklung für die Yorktown geplant hatte.

5 Fehlervermeidung

Jetzt, da wir die Ursache für den Fehler kennen, stellt man sich die Frage, wie man solche Fehler im Allgemeinen vermeiden kann, wenn man vorraussetzt, dass auch genügend Software–Entwickler vorhanden sind.

5.1 Wie kann man solche Fehler vermeiden?

Es gibt mehrere Möglichkeiten, wie man solche Fehler vermeiden kann.

5.1.1 In der Planung

Bei der Planung eines solch komplexen Systems hätte man mehr nach dem „Prinzip der Trennung der Belange“ handeln sollen und die unterschiedlichen Teil–Systeme der Yorktown so trennen müssen, dass ein Überlauf von einem Speicher eines Teil–Systems in den Speicherbereich eines anderen Teil–Systems verhindert wird. Dies hätte zumindestens einen totalen Ausfall des Gesamtsystems verhindert und der Antrieb wäre weiterhin funktionsfähig gewesen, wenn auch das Teil–System blockiert wäre.

5.1.2 bei der Implementierung

Bei der Implementierung hätte man darauf achten müssen, dass alle möglichen Einträge auf ihre Korrektheit überprüft werden. Wenn der Eintrag nicht korrekt ist muss eine Fehlerbehandlung erfolgen.

5.1.3 Testen

Ein weiterer Faktor bei der Verhinderung solcher Fehler ist das gründliche Testen. Testen ist immer sehr (zeit–)aufwendig, jedoch ist es für eine marktfähige Anwendung unverzichtbar. Und gerade bei Software für Kriegsschiffe müsste es genaue Richtlinien geben, damit ein Produkt (Software) freigegeben wird.
Bei gründlichem Testen hätte man an allen Stellen, an denen ein User Eingaben machen kann alle möglichen Eingabewerte oder zumindestens alle Sonderfälle der Eingabewerte durchtesten müssen. Es ist offensichtlich, dass durch dieses Testen ein solcher Fehler wie der „0“– Eintrag sofort aufgefallen wäre. Wenn man einen Fehler bemerkt kann man ihn beheben, oder eine zusätzliche Fehlerbehandlung integrieren.

5.1.4 Verifizieren

Das Verifizieren von Software ist extrem Aufwendig und eines der komplexesten Gebiete der Informatik. Man benötigt hierfür viele hochqualifizierte Personen. Allerdings sollte es die Regel sein, dass zumindestens besonders kritische Teil–Systeme verifiziert werden, da dies die sicherste Möglichkeit ist jegliches Fehlverhalten der Software ausschliessen zu können.

5.1.5 Warten

Warten von Software ist ein ständig ablaufender Prozess und erhöht somit auch die laufenden Kosten. Allerdings ist diese Erhöhung im Vergleich zu den

Einsparungen, die auf der USS Yorktown erzielt worden sind äußerst gering.

Bei einer guten Wartung wäre den dafür zuständigen Software–Entwicklern schnell aufgefallen, dass die Offiziere die Software in nicht dafür vorgesehener weise nutzen, indem sie die Daten direkt in der Datenbank ändern. Um das zu verbessern gibt es zwei Möglichkeiten:

- akzeptieren des Änderns der Werte
  Die Entwickler könnten das Ändern der Werte in der Datenbank durch einen User akzeptieren, müssten allerdings dann die Eingabewerte auf Korrektheit überprüfen und Fehlerbehandlungen integrieren.

  oder

- Zugriff unterbinden
  Die Entwickler könnten allerdings auch die Einträge vor dem Zugriff durch einen User schützen. Allerdings war dies ja anscheinend die einzige Möglichkeit für die Offiziere Fehler zu beheben und die Entwickler müssten daher auch alternative Möglichkeiten entwickeln und integrieren.

Außerdem könnten die Benutzer auch besser ausgebildet werden, um die Gefahren, die von solchen leichtsinnigen Änderungen im System ausgehen können, leichter zu erkennen.

5.2 Wie wird das Auftreten des Fehlers in Zukunft verhindert?

Die Navy will solche Fehler in Zukunft dadurch vermeiden, indem die Offiziere angewiesen werden an den bekannten Stellen keine „0“ einzugeben. Zusätzlich sollen schriftliche Aufzeichungen über alle Änderungen gemacht werden. Laut Navy kann somit bei einem Systemfehler das Schiff schnell wieder funktionsbereit gemacht werden. (siehe [5] [9])

Man muss sich allerdings im Klaren sein, dass es sich hierbei um ein bewaffnetes Kriegsschiff handelt. Aufgrund der unter Umständen herrschenden Stresssituationen kann dies keine vernünftige Lösung sein.
Die Navy geht allerdings mit solchen Vorfällen viel zu leichtsinnig um, denn aufgrund der Aussagen des Captains Richard Rushton gab es von 1995 bis 1997 schonmal zwei Ausfälle des Antriebs aufgrund von Buffer Overflows. Allerdings konnten in diesen Fällen die Fehler relativ schnell behoben werden,

im Gegensatz zu dem großen Vorfall, bei dem die Yorktown im Hafen noch zwei volle Tage repariert werden musste.

6 Ursachensuche

Es stellt sich auch noch die Frage, warum das Management nicht genügend Software–Entwickler für die Entwicklung der Systeme eingeplant hatten. Dies kann verschiedene Gründe haben:

- Unterschätzung des Aufwands der Systementwicklung
- Fehleinschätzung der Risiken
- Zeitdruck
- Kostendruck durch Regierung

Sehr wahrscheinlich war es eine Kombination aus all diesen Gründen, weshalb für dieses Projekt zuwenige Software–Entwickler eingestellt wurden. Allerdings lässt sich aus einigen Berichten herausinterpretieren, dass vor allem der Zeitdruck sehr groß war.

7 Smart Ship Concept

Trotz der Risiken und Fehler will die Navy auch weiterhin Schiffe im Rahmen des „Smart Ship Concepts“ bauen. Für neue Schiffe ist vorgesehen, dass sie bei gleicher Leistung höchstens noch ein drittel der bisher nötigen Besatzung benötigen.
Außerdem soll auf den Schiffen auch weiterhin Windows NT bzw. der Nachfolger Windows 2000 verwendet werden.

7.1 Vorteile

Die Vorteile des „Smart Ship Concepts“ sind offensichtlich.

- durch den geringeren Bedarf an Personal werden die laufenden Kosten deutlich gesenkt. Sie werden zwar durch die Kosten der Software–Wartung zwar wieder erhöht, aber diese Mehrkosten sind viel geringer als die Einsparungen.
  Große Einsparungen kann man auch bei den Instandhaltungskosten machen. So konnte z.B. bei der USS Yorktown von über 45.000 Arbeitsstunden pro Jahr für Instandhaltung mehr als 45 % eingespart werden.

- durch die Verwendung von „Standard Software“ wie z.B. Windows 2000 werden die Beschaffungskosten reduziert, denn die nötigen Lizenzen sind viel günstiger, als wenn man die benötigten Betriebssysteme selbst programmieren müsste. Allerdings kann es vorkommen, dass diese Betriebssysteme nicht ganz optimal für die Anforderungen geeignet sind.

Als weiteres Beispiel für die Vorteile des „Smart Ship Concepts“ dient auch der US Zerstörer USS McFaul. Er hat eine Besatzung von 350 Mann.
Die Navy hat für 2008 einen Nachfolger mit gleicher Leistung angekündigt. Dies soll das „Smart Ship“ USS Zumwalt sein, das nur noch 90 Mann Besatzung benötigen soll. (vgl. [7])

7.2 Risiken

Doch es gibt nicht nur positive Stimmen. So bemänglen Kritiker, dass auf den neuen Schiffen weiterhin standardisierte Software wie z.B. Windows NT / 2000 verwendet werden soll, obwohl es in der Vergangenheit wohl schon öfter zu Fehlern gekommen ist.
Sie argumentieren hauptsächlich damit, dass standardisierte Software oft gar nicht, oder zuwenig verifiziert und getestet sei. Man geht daher doch ein relativ hohes Risiko von Fehlfunktionen und Abstürzen ect. ein. Solche Fehlfunktionen könnten auf einem bewaffnetem Kriegsschiff auch verheerende Folgen haben. Mögliche Folgen wären z.B.:

- Bei einem Ausfall des Antriebs während eines Kampfeinsatzes würde das Schiff zur leichten Beute für die gegnerischen Streitkräfte werden. Außerdem könnten durch das manöverierunfähige Schiff auch die Formationen der eigenen Streitkräfte verändert werden und dadurch auch andere Schiffe gefährdet werden.

- Es ist auch denkbar, dass ein beliebiger Speicherüberlauf auch die Steuerung von Ventilen ect. beeinflussen kann. Die Folgen einer solchen unkontrollierten Steuerung könnten fatal sein. Vor allem aufgrund der Tatsache, dass das Schiff auch bewaffnet ist und somit auch über taktische Systeme verfügt. Falls diese auch beeinflusst werden könnten ist das Risiko noch sehr viel größer.

Außerdem wird bemängelt, dass Windows NT bzw. Windows 2000 gar nicht uneingeschränkt echtzeitfähig sei. Es gibt andere Betriebssysteme, die weitaus bessere Fähigkeiten bezüglich Determinismus, Ausfallsicherheit und Zuverlässigkeit bieten. Eine bessere Lösung wäre es Windows NT/2000 für den Desktop zu verwenden und für die direkten Echtzeitsteuerungen eine andere Software einzusetzen. (siehe [8]

Ein weiterer Punkt ist auch die höhere Anfälligkeit für Sabotage durch Hacker. Die Lücken in der Sicherheit von Standard–Software sind weit verbreitet und bekannt. Hacker könnten dies ausnutzen und somit erheblichen Schaden anrichten. Bei eigens für die Systeme entwickelter Software wäre ein solches Risiko viel kleiner, da Hacker erst Lücken im System finden müssen. Und das ist nicht einfach möglich, wenn die Hacker auch keinen Zugriff auf ein solches Software–System haben.

7.3 Fazit

Bei der Bewertung des „Smart Ship Concepts“ stehen die enormen Vorteile wie die Einsparung von großen Teilen der Besatzung und die stark verringerten laufenden Kosten, den doch evtl. großen Risiken gegenüber, die entstehen können, wenn die System nicht ordentlich entworfen und getestet bzw. verifiziert werden.
Man sollte also die Entwicklung zu den sogenannten „Smart Ships“ begrüßen und unterstützen, aber man muss sich der Risiken bewusst sein, die durch Fahrlässigkeit bei der Entwicklung der Systeme entstehen können.

Literatur

[1] *'USS Yorktown (CG 48)'*
`http://navysite.de/cg/cg48.html`
(Stand: 15. November 2003)

[2] Frank Meilinger,
'Große Ernüchterung bei der Umstellung von OS/2 auf NT'
`http://people.wiesbaden.netsurf.de/~meile/other\_sources/uss\_yorktown.html`
(Stand: 15. November 2003)

[3] Gregory Slabodkin, GCN (13.07.1998):
'Software glitches leave Navy Smart Ship dead in the water'
`http://www.gcn.com/archives/gcn/1998/july13/cov2.htm`
(Stand: 15. November 2003)

[4] Gregory Slabodkin, GCN (31.08.1998):
'Smart Ship inquiry a go'
`http://www.gcn.com/archives/gcn/1998/august31/1.htm`
(Stand: 15. November 2003)

[5] Gregory Slabodkin, GCN (09.11.1998):
'Navy: Calibration flaw crashed Yorktown LAN '
`http://www.gcn.com/archives/gcn/1998/november9/6.htm`
(Stand: 15. November 2003)

[6] Naval Vessel Register:
'USS Yorktown (CG48)'
`http://www.nvr.navy.mil/nvrships/details/CG48.htm`
(Stand: 15. November 2003)

[7] Konrad Lischka (5.10.2000):
'Windows-Navy im Einsatz'
`http://www.konradlischka.de/nhproben150.htm`
(Stand: 15. November 2003)

[8] Greg Bergsma (1998):
'Echtzeit-Erweiterungen fr Windows NT'
`http://www.swd.de/documents/papers/nt\_de.html`
(Stand: 15. November 2003)

[9] Justin Bague (17.11.2002):
'The USS Yorktown'
`http://www.ee.ualberta.ca/~musilek/cmpe510/USS_Yorktown.pdf`