

Will Software Failures Halt the Availability of Business Insurance?

Jeffrey Voas (jmvoas@RSTcorp.com)
Reliable Software Technologies, Sterling, VA USA

The Year 2000 problem has forced the insurance industry world-wide to reexamine the impact of software quality on business disruption insurance. Most large and medium-sized corporations that purchase this form of protection do so against threats such as natural disasters or other calamities. Without it, most corporations could not exist because of the overwhelming shareholder fear of bankruptcy.

As an example, whenever turmoil erupts in the Middle East, insurers raise cargo premiums for the oil tankers traveling through that area. In fact, during the recent Gulf War, military escort ships were needed to decrease the risk of attack on oil transport ships. The reason for this was that insurers were hesitant to continue to offer cargo insurance to oil tankers without military protection, and shipping companies refuse to transport cargo without insurance. If they shipped cargo without it, they would be responsible for the value of the cargo if it were lost.

Today, we are seeing similar events in the information technology arena. Public announcements of insurer unwillingness to cover corporations against the losses expected from Year 2000 have been made. This has occurred via insurers adding waivers to their standard business disruption policies that exclude Y2K-caused problems. Consider the recent case where Swedish insurer Trugg-Hansa made the following exclusion effective May 1, 1998 in the general conditions of their business insurance policies "The policy will not cover damage, cost, legal or other liability caused directly or indirectly or connected to time-related disturbance in computer functionality." This demonstrates the extreme, defensive posturing being seen as a result of the Y2000 problem. But of equal significance, it opens the door for non-time-related exclusions for other anomalous software behaviors. For example, exclusions might someday read like "The policy will not cover damage, cost, legal or other liability caused directly or indirectly or connected to disturbances in computer functionality." Such a waiver enables an insurer to avoid responsibility for all computer-related problems. Instead, the onus is placed on consumers to determine the qual-

ity of the computer systems that they employ. The consumer now bears his or her own liability, without access to an insurer to step in as their surrogate in case of a mishap. This represents a first in our industry *insurers are so concerned about software failures that they have started adding exclusions in the event of software failures in their policies.*

Software insurability refers to the degree of risk that an insurer is willing to take to indemnify or guarantee a software user against loss caused by software failure in exchange for an insurance premium. Note here that the insurer is not actually insuring the software, but is instead insuring the entity or processes that the software impacts. This might include an airplane, the operations of a bank, or the production rate of a manufacturing plant.

During the past 2 years, we started to see the first insurance offerings for information systems. Marsh and McLennan offer an insurance product called 2000 Secure related to problems with the Millennium bug. Premiums for 2000 Secure range from \$1M to \$10M. Network Risk Management Services (NRMS) offers a slew of insurance policies, offering coverage for "viruses, hacker attacks, overwhelmed communications, damage to data, network hostage situations, and key recovery issues or loss of income."

Most of these offerings protect against only certain kinds of threats and in fixed compensation amounts. For example, if an attack of type X were to occur and losses were to result, then you could expect to receive a claim for damages Y . If an attack outside of X occurred, you may or may not be covered. It will simply depend on the exceptions enumerated in the policy. Nonetheless, these policies are very attractive in today's digital age, and the market for them will continue to grow.

Before offering insurance for software-controlled entities, an insurer should understand (1) the worst-case scenarios that can result if the software were to fail, (2) what the dollar losses will be if that type of software failure were to occur, and (3) the predicted frequency of such events. These three pieces of information allow insurers to decide if they can profit under

such conditions. Recognize that insurers do not issue insurance to lose money or break even; they expect to profit in exchange for the risks they take.

Interestingly enough, a new US corporation has been formed to address the problem of determining software insurability on behalf of insurers the Software Testing Assurance Corporation (STAC). This company was founded in 1997 to provide independent certification when requested by insurers. This independent certification to attain insurance has limited availability. It is only available to corporations that have applied for *business disruption* insurance. Certification assessment will only be provided after written requests are received from an insurer.

The founding of this new company opens the door for additional software certification standards (as well the possibility of other companies entering into the certification marketplace) when business risks can be directly tied to the reliability, availability, and security of information systems. Now that one's ability to attain business insurance has been tied to the quality of the software that they employ, this opens a new chapter in what "is or is not" acceptable software quality.

Software users are now being made to bear the burden of bad software in a completely different way than simply lost productivity. They may have to fully bear the burden and consequences of business disruption. Hopefully this will encourage users to demand warranties of quality from vendors and software publishers. If this were to occur, this could greatly change how the risks and consequences of software failures are measured. That is, shouldn't we be measuring reliability as a function of the consequences of software failure as opposed to the frequency of failure?

The reason why the current events surrounding companies such as STAC and Trugg-Hansa are so significant for the Software Reliability Engineering (SRE) community is that for the first time we see a marketplace for *highly accurate* software reliability assessment. Previously, software reliability assessment has been an *ad hoc* activity usually done by a software publisher for the sole purpose of determining whether the software was ready for release. Because the accuracy of whatever model chosen was automatically brought into question, the model was used more as a heuristic than as an absolute measure. And because there is a plethora of software reliability models that can be employed as testing stoppage criteria, it is anyone's guess as to whether the the appropriate model (even if inaccurate) was employed. If the software was not ready for release, but was released because the model indicated such, the worst that could happen

would be an "egg on the vendor's face" and possibly some financial loss. This financial loss could come in the form of rework, a greater time-to-market, loss of reputation, and possibly legal action.

But here we are not concerned with protection of a software publisher's business or reputation. Instead, we are concerned with protecting (1) the software user's business against defective software, and (2) the insurance company as it attempts to indemnify the user. To accomplish these two goals requires accurate software quality assessment on behalf of the insurance company since they are assuming all risks.

This may not seem like a monumental task, but consider that software reliability models are assumption-filled, and even the same software system can cause terrible claims against the insurer when employed in one environment whereas if the system is used elsewhere it will cause no problems. Recall that the Ariane 4 software worked properly until it was reused in the Ariane 5. It probably would have been an "easy sell" convincing an insurer that the Ariane 4's software was a "sure bet" in the Ariane 5 rocket. After all, the development team of the Ariane 5 had fallen for that same myth because the historical data suggested such. But as we all now know, that small change in the environment between the Ariane 4 and Ariane 5 made such an enormous difference in the correctness of the trajectory software that the maiden flight of the Ariane 5 was a disaster.

What this suggests is that accurate software quality assessment will not even be as simple as just having a big repository of information that quantifies the many "ilities" for COTS products. Accurate risk assessment must also be with respect to the user's idiosyncracies. This makes independent certification of software systems necessary in the environment in which they will be insured. This does not mean, however, that certain risk analyses cannot be applied once and the results stored, but that may well not be enough for accurate risk assessment.

So in summary, there is now a marketplace for accurate Software Risk and Consequence Assessment (SRCA). That demand is being fostered by an insurance industry that is highly sensitive to potential losses that they will incur unless they exclude unforeseen software-related problems from their business disruption insurance. The software industry has failed to adequately police itself from producing defective software. Nor has the government policed the software industry. Probably the only other group with enough clout to force software quality to be taken seriously is the insurance industry. And they are doing so.