Chapter 1

Estimating Complex System Robustness from Dual System Architectures

Chad Foster
Massachusetts Institute of Technology, Cambridge, MA
crfoster@mit.edu

This paper investigates differences in robustness and reliability between hardware only and hardware-software products presented in the patent literature. Simplified models explore the reliability and robustness between these products.

In general the more complex hardware-software systems provided higher robustness but below average reliability. Complexity is a good indication of the decreased reliability and increased robustness.

Systems that require both high complexity and reliability require large investment, extensive testing and attention to system level failures. Some examples of this phenomena are presented from industry.

1.1 Introduction

Patent examples from a variety of industries are used to explore the issues of robustness and reliability. A short discussion is included about other aspects of the differences; reliability growth, the cost of changes, and development time frames.

The following discussion presents eight patents and the reasoning behind the mechanical or software aspects of each solution. Using information from

literature[8] the generic reliability was compared for each solution. A simplified robustness comparison was also made between the patent pairs.

For the purposes of this paper robustness is defined as the ability to deal with variance in the system. Reliability is defined as the generic time between failure for the system. All components in the system are considered to be of average reliability.

A general discussion of the results follows with some conclusions and directions for future work.

1.2 Patents

Patent number 6,991,280 shown in Fig. 1.1 describes "Airflow Control Devices Based on Active Materials." This device is based on an active material such as a piezoelectric material or a shape memory alloy. Included in this valve is a sensor and a controller to provide closed loop position control. Specific robustness and reliability claims include a reduction in maintenance and a reduction in the level of failure modes.

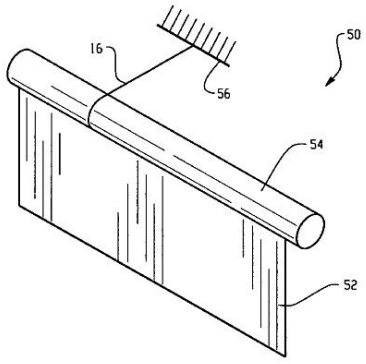

Figure 1.1: Patent 6,991,280

The comparison patent is 6,926,346 shown in Fig. 1.2 describes an "Adjustable Vehicular Airflow Control Device." This device is more conventional, using a belt or gear drive, to control a deflector. The implementation presented uses feedforward control that utilizes the vehicle speed sensor or a manual selection. The major innovation is adding a controllable airflow device where it has historically been fixed.

Patent 4,086,022 shown in Fig. 1.3 describes an "Improved Compressor Casing." This improvement is described in other papers[2] and includes the addition of a number of slots in the casing to delay the onset of stall or surge. This solution uses a mechanical change to increase the operating range of the system

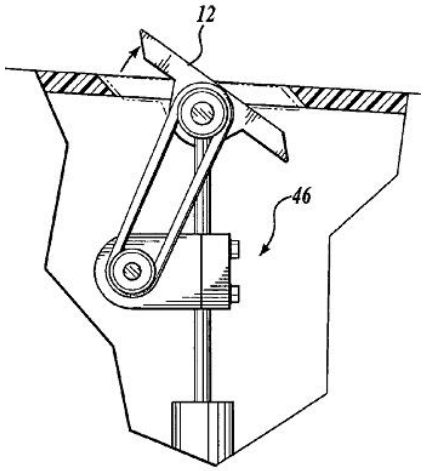

Figure 1.2: Patent 6,926,346

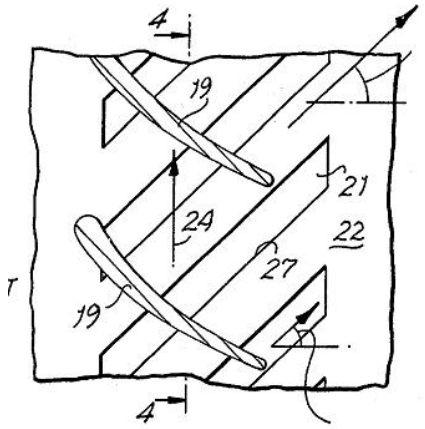

Figure 1.3: Patent 4,086,022

A comparison patent 6,354,602 uses "measurement and active operating limit line management" for surge avoidance. This is done completely in software using sensor measurements and transfer functions to actively control the air and fuel flows. The major improvement, in addition to surge, is the ability to control for compressor fouling.

The next example adds electronics to a current system. Patent 6,889,803 shown in Fig. 1.4 creates a "torsional active vibration control system." This device creates an adjustable torsional damper with at least one actuator that can adjust the absorption characteristics of the system.

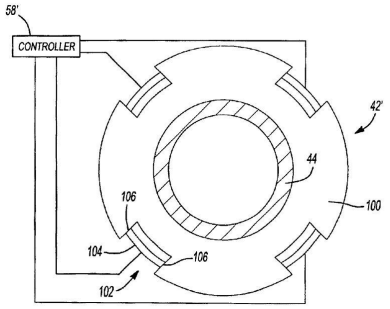

Figure 1.4: Patent 6,889,803

A comparative example without electronics is given in patent 6,371,857 shown in Fig. 1.5 it describes a torsional vibration damper with increased stability. The base structure of the two inventions is similar.

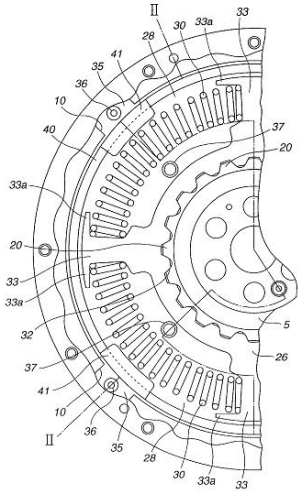

Figure 1.5: Patent 6,371,857

An example from the automotive literature exists in a locking differential. One traditional locking differential is called the Torsen differential and is explained in patent 2,859,641 shown in Fig. 1.6. This differential is completely mechanical with a number of gears providing complete function. Although this patent is 48 years old it is still used extensively in vehicles manufactured today.

A comparative example is a relatively new patent that uses electromagnetic clutches and a number of sensors to create a locking differential that can dynamically adapt to changing road conditions. This patent, 6,637,572, is shown

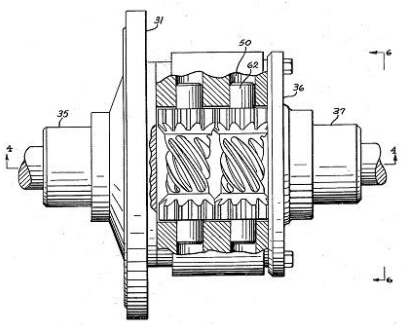

Figure 1.6: Patent 2,859,641

in Fig. 1.7. The flexibility is improved and this design lowers the theoretical system losses by reducing the number of meshing gears.

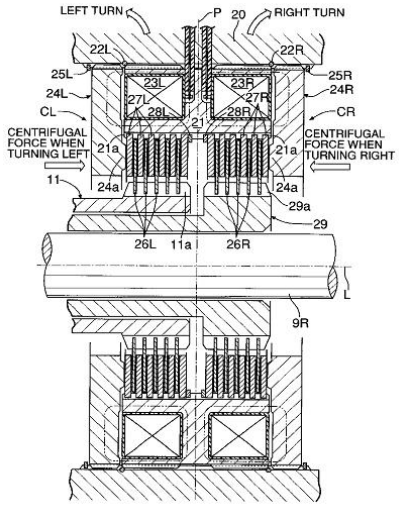

Figure 1.7: Patent 6,637,572

There are numerous other patent examples that demonstrate this dual architecture. Often new products compete with prior art that do not provide good comparative examples in the patent literature, and so they were excluded from the study.

1.3 Reliability

Patents were drawn from literature that were well known in the marketplace and offered typical comparisons. It is the author's belief that these are typical representations of dual patent architectures and not unique cases.

The reliability for each of these systems was calculated by using the failure data in published literature[8]. Two designs were designated as base designs and only the change in reliability was noted. The expected reliability of each design did not differ by more than 75 failures per million operating hours. An attempt was not made to add modification to the life equations, because similar in-use conditions were assumed.

The tabulated comparison between six patents of two differing designs is as follows:

Device	Failures/Million Hours	
	Mechanical	Software
Airflow	6,991,280	6,926,346
	44.5	119.6
Vibration Damper	6,371,857	6,889,803
	-	66.3
Surge Avoidance	4,086,022	6,354,602
	-	27.3
Transmission	2,859,641	6,637,572
	34.4	100.5

It is noted that in every comparison the reliability of the system decreases with the addition of electronics and software. The mechanical components were replaced by lower reliability electrical systems. There are examples of extremely complex mechanical systems or specific cases where this is not true and the software system has improved reliability by removing failure prone components.

In addition to this high-level calculation of reliability it should also be noted that the reliability growth differs between these two systems. It can be feasible to improve the reliability of the software in-situ, through software upgrades or bug-fixes. Mechanical systems are more difficult to upgrade and are very expensive to replace in the field. The most frequent upgrade to non-safety related items is to fix-as-failed. This practice replaces the poor previous design with a better design only when the product is brought in for a repair (reactive maintenance).

The models of this growth show a lower dollar requirement for the software system to repair after release. Taking the rule of thumb that the cost of changing mechanical systems increase by an order of magnitude for every development stage, and taking some field data on software development, the lesser cost of software is shown in Fig. 1.8.

[6]

The reliability models of the software systems do not include the actual software, only the hardware components supporting that software. There is

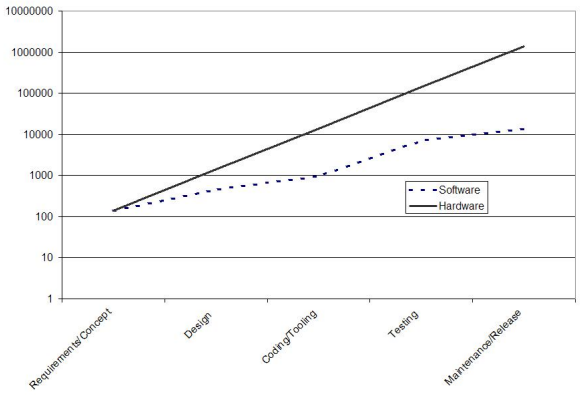

Figure 1.8: Cost of Making changes in Mechanical and Software Systems

no accepted method for measuring software reliability[11]. The best estimate used by researchers is based on the number of lines of code[6]. The addition of software only reduces the overall system reliability, and can reduce it quite severely and in unexpected ways.

Although these outline generic trends it is conceivable that for any particular product these differences may be insignificant or even reversed. On average, though, the software based product will have a lower reliability and could exhibit peculiar failures.

Software can be improved with additional redundancy, in hardware and software, reliable software practices, simpler software and adding more reliability testing. The most reliable systems avoid off-the-shelf components by creating a unique reliable solution, this avoids including problematic code. Reused mechanical components have the opposite effect and are usually preferred due to their demonstrated robustness.

Investigating specific reliability cases could determine more causal effects for the differences in reliability between hardware and software systems and highlight pitfalls the practitioner should avoid.

1.3.1 Failure Severity

Failure severity is defined as the number of failures that would completely prevent the system from completing its function. Ideally this is zero and any failure would still allow the system to perform its function (even with limited performance). This is interpreted in the software domain as fault tolerance.

The primary catastrophic mechanical failure is fatigue. These failures are normally abrupt without obvious warning. The primary prevention technique is expensive on-site monitoring, or adequate mechanical design. This failure can be correlated to the loading, endurance limit and yield stress. This is a straightforward second order relationship[10].

In software there are often numerous catastrophic failure modes with nonlinear relationships. The complexity of software systems stems from interdependencies, assumptions, and baggage that are difficult to test. Replacing mechanical systems with software systems requires changing the development process to account for this complexity and testing difficulty.

1.4 Robustness

The robustness definition used in this paper is the ability for a system to react to noisy input parameters with little performance degradation. Simple robustness comparisons are used that compare the different embodiments. For each of the previous patents, a relative ± comparison is made about the robustness.

The Airflow device's functional requirements concerning robustness and the expectation to those requirements are laid out in the following chart. Note the comparisons were only made '+' and '-' because of the high-level nature of this comparison; further comparisons would use the approximate distributions of these inputs and mathematical or simulation models of the outputs.

Requirement	Mech(6,691,280)	Soft(6,926,346)
Variance in Air	0	0
Unstable Current	0	+
Temperature Change	-	+

The difficulty of controlling the active materials severely limits their use. Temperature, current, temperature and other affects need to be actively controlled and so decreases the base robustness of this solution.

The two methods to avoid surge were to first change the casing and the other is to use sensors and a control system. Here is the comparison of their noise factors.

Requirement	Mech(4,086,022)	Soft(6,354,602)
Pressure Ratio	+	0
Soot Loading	0	+
Temperature	0	+

The added software gives the system tremendous flexibility to operate closer to the surge limit and thus improve operating efficiency. The addition of mechanical slots adds manufacturing complexity and provides some losses during normal operation.

The torsional vibration damper added an electronic controller with the intent of increasing the damped frequency range. A comparison chart of the noise factors:

Requirement	Mech(6,371,857)	Soft(6,889,803)
Frequency Variance	0	+
Wear	0	+

Comparing the locking transmissions also shows greater robustness in the software based solution.

Requirement	Mech(2,859,641)	Soft(6,637,572)
Friction Loss	-	+
Driving Speed	0	+

The robustness of these functional requirements were improved when performed with the software based system over the mechanical based system. In general the addition of the software addressed a failure and gave a more consistent output for the range of inputs.

1.5 Complexity

The systems so far have been assumed to follow a linear law of requisite variety.[1]. The amount of control that is generated by the system is directly dependent on the number of independent variables. The question arises about the coordination between the variables. It would be expected that the percentage of problems due to the software would increase by a power law as the amount of software in vehicles increases. This is not observed in the recall data available through the National Highway Transportation Safety Administration (NHTSA)[9]. Motor vehicles seem to have the same percentage and severity of recalls with software as with hardware.

Although there are numerous examples of poor software, and newsworthy failures (Ariane 5 missile, AT&T switching system, Therac-25, Osprey Helicopter)[6][5] the added complexity does not create disproportional numbers of failures. This does not indicate that the systems are not more complex, they are, there is just no evidence of a power law relationship. The decrease in reliability for the software products can be seen from consumer reports data in Fig. 1.10.

One of the big software development differences is the cost to find and fix any errors, it is reported that 80% of the software development costs are in finding and fixing defects[6]. During the development of the Space Shuttle the independent validation and verification group was was almost as large as the development group. The Space Shuttle software is considered the state of the art for reliable code at 0.1 faults per 1000 executable lines of code(KXLOC). (Windows 2000 is rated at 2-4 per KXLOC)[6]

There are also variations in software quality not seen in the more standardized, and simpler, mechanical design. For example, in an experiment[7] 27 versions of the same algorithm were developed in Pascal the most reliable did not fail 1 in a million trials and the least reliable failed 10,000 in a million trials.

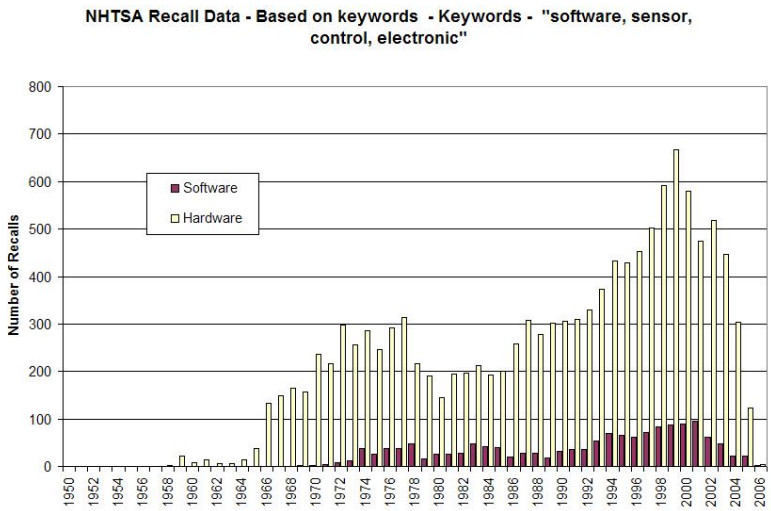

Figure 1.9: Number of automotive recalls related to software

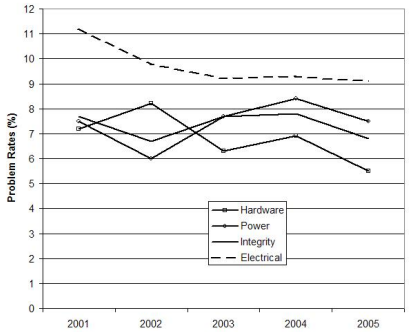

Figure 1.10: Automotive Reliability by area[3]

There are unique requirements in both mechanical and software systems. Making the decision between the two systems requires complex tradeoffs in development resources, testing, and final end use.

1.6 Conclusion

The move to more complex systems and specifically software systems is not without tradeoffs. The addition of components of greater complexity reduces reliability at the betefit of increased robustness.

Initially, it was believed that the additional components would underestimate industry failures, and complexity would play a role, this turns out to be incorrect.

The current increases to complexity has been dealt with sufficiently. There is potential for further increases in complexity to lead towards these cascading, unexpected failures. This was not seen in the aircraft industry[4] and is not expected in the automotive industry.

Future work is focused on aiding the designer to make the decision between using software based or mechanical based solutions. This will allow a more knowledgable choice between system options.

Bibliography

[1] Bar-Yam, Yaneer, "Multiscale variety in complex systems", *Complexity* **9**, 4 (2004), 35–45.

[2] Clausing, Don, and Daniel D. Frey, "Improving system reliability by failure-mode avoidance including four concept design strategies", *Systems Engineering* **8, No. 3** (2005), 245–261.

[3] Consumers Union, "Consumer reports website", http://www.consumerreports.org/.

[4] Frey, Daniel D., John Sullivan, Joseph Palladino, and Malvern Atherton, "Part count and design of robust systems", *Submitted to INCOSE* (2006).

[5] Gage, Deborah, and John McCormick, "We did nothing wrong: Why software quality matters", *Baseline* (2004).

[6] Hatton, Les, "Software failures, follies and fallacies", *IEEE Review* (1997).

[7] Knight, J. C., and N.G. Leveson, "An experimental evaluation of the assumption of independence in multi-version programming", *IEEE Transactions on Software Engineering* **12** (1986), 96–109.

[8] Moss, T. R., *The Reliability Data Handbook*, ASME Press (2005).

[9] of Transportation, United States Department, "National highway traffic safety administration", http://www.nhtsa.dot.gov/.

[10] Shigley, Joseph Edward, and Charles R. Mischke, *Mechanical Engineering Design* 6th ed., McGraw-Hill (2001).

[11] Smith, David J., *Reliability, Maintainability and Risk - Practical Methods for Engineeris*, Elsevier (2005).