System Issues in Software Development-Problems, Effects and Solutions

Confidence Idim, Sayeed Sajal
Department of Computer Science
Minot State University
500 University Ave W, Minot, ND, 58707
Sayeed.Sajal@minotstateu.edu

Abstract

This paper talks about some system issues that could occur when creating a system or product. The problem that is discussed in this paper are software and hardware tradeoff and increase in hardware-software complexity. Also, the effects of system issues mentioned above with popular examples are to be discussed in this paper. Furthermore, possible solutions like hardware and software codesign that is used to reduce some of these system issues are also going to be part of the paper. In conclusion, this paper is going to be about problems, effects, and solutions to system issues.

I. Introduction

In the early 1900s, the concept of having a complex system that integrated hardware and software components of the system together was a theoretical concept. At that time, most computers were single-purposed as they were designed for one task and did not support the use of software applications. However, this concept motivated researchers to come up with models that implemented this integration. Hardware-software integration encourages organization, flexibility, cost-effectiveness, and efficiency of the whole system. Evidence of this integration is existent in a lot of important aspects of our lives. The remaining part of this paper is divided into five sections. Section II of this paper discusses system issues in software development while section III and IV discuss the effect and solutions of problems listed in section II. Lastly, section V concludes the paper.

II. Problems

A. Increased Complexity in Hardware and Software Component

Smartphones, cars, and computers are heavily used daily in our present world. These devices make strenuous tasks easy, and as a result of its importance in our lives, it is subject to constant improvements to satisfy the need of customers. Improvements in these devices lead to an increase in the efficiency and functionality but it affects the hardware complexity of the system negatively. An increase in the hardware complexity is a necessity because it is the only way the system can meet up with the requirements of the customer demand. This increase in complexity tends to solidify the predictions of Moore's Law, which states that the number of transistors in an integrated circuit doubles about every two years. Software developers would have to always keep changing certain features or change the whole design of the software to meet the requirements of the constant change in the architecture of the hardware. This has led to the increase in software complexity which is as rapid as that of the hardware. According to David Garlan et al, "the design problem goes beyond the algorithms and data structures computation: designing and specifying the overall system structure emerges as a new kind of problem" [6]. Complexity in both the software and hardware components of the system is subject to an increase as their respective design have to be changed constantly.

B. Hardware Software Tradeoff

Due to the increase in codesign complexity, the hardware-software tradeoff problem emerges, and it is defined as the decision on system structure (that is whether to sacrifice hardware for software implementation or vice versa). A lot of factors are taken into consideration when making decisions about whether to go for software design or hardware design. Factors such as cost, speed, efficiency, and size are common topics in design decision making. It is discussed in [1] that "the actual requirements for the DSPs differ for the various fields for application: For hand-held devices, e.g. meeting the signal processing demands of one data channel while minimizing cost and power consumption is critical." This situation is inefficient as one part of the system is neglected. With our present technology, this possesses a big problem as the need for the hardware and software component of the system is of high demand.

III. Effects

As explained above, the reason for the need for an increase in software and hardware complexity is to meet the demands of the users. This increase in complexity influences the cost and time taken to complete the project. Several cost estimation techniques have been designed to tackle the effect of costs. However, this cost estimation techniques have some limitations because the cost estimation is made based on the functionality of the software. This turns out to be underfunded or overfunded, causing a lapse in the production of the software. Furthermore, project failure is another massive occurrence when there are system issues. Project failure caused by system issues has fatal repercussions if there are not handled meticulously. As discussed in [4] “The $170 million VCF system, a searchable database intended to allow agents to “connect the dots” and follow up on disparate pieces of intelligence, instead ended five months ago without any system being deployed.” Below are solutions that solve problems caused by system issues listed in section II.

IV. Solutions

A. Codesign:

Codesign solves the problem of integrating the hardware and software components of the system. As discussed in [5] “Codesign is a concurrent and cooperative design approach that includes a fundamental component of the capability to explore hardware/software tradeoff.” This design encourages efficiency as possible problems that could occur in the development of the software can be detected due to concurrency in codesign. Furthermore, as described above, the complexity of the present technology affects every aspect of that and so does codesign. In codesign, hardware and software designers work hand in hand to make sure that the final product is a flawless one. Codesign also explores and solves the issue of complexity of the system as explained in [3] "codesign techniques are mainly driven by the complexity of today’s electronic system designs and serves as a means (or at least try) to close the well-known design gap to produce correctly working and highly optimized (e.g., with respect to cost, power, performance) system implementations.”

B. Exploring Hardware Software Tradeoff:

Hardware-software tradeoff when explored meticulously can be used to produce solutions to some system issues. Several techniques and approaches can be applied to the architecture of the hardware to improve the performance of the system. Approaches such as simple and fast instructions, load and store architecture and condition codes and control flow can be implemented. As described in [2], simple and fast instructions offer three advantages which are:

a. increase in execution time because the instruction set is not complex.
b. the workload of the compiler is minimized because of the absence of sophisticated instruction set.
c. The performance improvement from machines and compilers are significant.

Furthermore, the use of load and store architecture is an efficient way to explore hardware software tradeoff. All these advantages tend to increase the performance of the system, so if there is a tradeoff, the effect on the performance of the system would be minimal or unnoticeable.

V. Conclusion

Hardware-software integration is a necessity with the rapid advancement in technology. However, this integration is heavily flawed with system issues. Hardware-software tradeoff and the increase in hardware-software complexity are the major flaws of the integration talked about in this paper. Furthermore, the effects of these issues are discussed in the paper. Effects such as cost and time are taken to complete the project was talked about in this paper. Also, project failure as an effect of the system issues was discussed in this paper. Lastly, codesigns and exploration in hardware and software tradeoff is an effective solution to system issues presented above. These solutions do not completely solve the system issues but are efficient in minimizing system issues to little or no effect.

References

[1] H. Michel, A. Worm, M. Munch, and N. Wehn, "Hardware/software trade-offs for advanced 3G channel coding," *Proceedings 2002 Design, Automation and Test in Europe Conference and Exhibition*, Paris, France, 2002, pp. 396-401. doi:10.1109/DATE.2002.998304

[2] J. Hennessy, N. Jouppi, F. Baskett, T. Gross, and J. Gill, “Hardware/software tradeoffs for increased performance,” ACM SIGPLAN Notices, vol. 17, no. 4, pp. 2–11, Apr. 1982. doi:10.1145/960120.801820

[3] J. Teich, "Hardware/Software Codesign: The Past, the Present, and Predicting the Future," in Proceedings of the IEEE, vol. 100, no. Special Centennial Issue, pp. 1411-1430, 13 May 2012. doi:10.1109/JPROC.2011.2182009

[4] R. N. Charette, "Why software fails [software failure]," in *IEEE Spectrum*, vol. 42, no. 9, pp. 42-49, Sept. 2005. doi:10.1109/MSPEC.2005.1502528

[5] S. Kuman, B. W. Johnson, J. H. Aylor, and W. A. Wulf, "A Framework for Hardware/Software Codesign," in Computer, vol. 26, no., pp. 39-45, 1993. doi:10.1109/2.247650

[6] D. Garlan and M. Shaw, “An introduction to software architecture,” in advances in Software Engineering and Knowledge Engineering, World Scientific, 113, pp. 1 -39. doi: 10.1142/9789812798039_0001