Hardware and Software: How Can We Establish Concurrency between the Two?

Shuichi Fukuda

Stanford University

Abstract. Today, most of our products are combinations of hardware and software. we must remember software works on hardware. But the function of hardware is fixed, while that of software is evolving throughout its life cycle.

Hardware is an individual living. Once they are created, they start to degrade. Therefore, maintenance is a very important task with respect to hardware. But hardware functions are fixed so that it is relatively easy because the objective of maintenance is to restore the degraded quality to its original designed level.

Software, however, is a species. Each software, once born, grows in a different way to adapt to the situation. And many new portions are added on. Therefore, decommissioning becomes very difficult for software.

Since most of our products are combinations of hardware and software, we should pay attention to how we should work them together effectively and how we can decommission the system which is composed of both hardware and software.

This paper discusses this issue and suggests the possible solution.

Keywords. Hardware, Fixed function, Individual living, Software, Evolving function, Species, Coordination

1 Introduction

Today, most of our products are combinations of hardware and software. we must remember software works on hardware. But the function of hardware is fixed, while that of software is evolving throughout its life cycle.

Shuichi Fukuda
Stanford University, Consulting Professor
3-29-5, Kichijoji-Higashicho, Musashino, Tokyo, 180-0002, Japan
Phone: +81-422-21-1508 Fax; +81-422-21-8260
Email: shufukuda@aol.com

Hardware is an individual living. Once they are created, they start to degrade. Therefore, maintenance is a very important task with respect to hardware. But hardware functions are fixed so that it is relatively easy because the objective of maintenance is to restore the degraded quality to its original designed level.

Software, however, is a species. Each software, once born, grows in a different way to adapt to the situation. And many new portions are added on. Therefore, decommissioning becomes very difficult for software.

Since most of our products are combinations of hardware and software, we should pay attention to how we should work them together effectively and how we can decommission the system which is composed of both hardware and software.

This paper discusses this issue and suggests a possible solution.

2 Hardware: What characterizes it

Hardware is produced with fixed funtions as shown in Figure 1. But once delivered, it will start degrading. The majority of our efforts is to restore its degrading functions to their original level. Thus, maintenance is very important task in hardware.

But hardware is an individual living. The life prediction of hardware is very much like predicting life for each of us. Although it is very difficult to predict its life due to wide variability, it is fundamentally in the same framework as the prediction of our life. In fact, it is well known that hardware failure rates and our death rates exhibit the same bath tub curve.

3 Software: What characterizes it.

Sotware used to be produced in the same way as hardware. In fact, there were companies in Japan who called their software division “Software Factory.”. There were no distinctions between hardware and software at that time.

Software engineers did their best to comply with the design requirements and to deliver software products which satisfy them. But it became soon clear that it is impossible to completely debug and with the increasing diversification, these efforts do not really answer to the needs of customers. Customers would like to have software with more adapatablity and flexibilty than one with fixed functions without any bugs.

Around 1980, knowledge engineering was proposed and what knowledge engineering brought to software sector was quite revolutionary. The most important impact it brought to us is the concept of continual prototyping.

Since then, software development was changed completely. Software engineers started to provide us with a baby software (Beta version) and this baby grows with us. Software was no more “fixed function” product.

But as we come closer to the end of 20^{th} century, around 1990, diversification grew more and more and situations come to change more frequently and widely.

To respond to these changes, software changed once again. It development philosophy was to grow a product. But now it changed from an individual living to a specimen. Many new pieces are coming to be added on so that software products come to “evolve” as species. The discussion based on an indivual living or entity does not hold any more. We have to look at software as “evolving system” or “evolving species”.(Figure 2).

Since software is a species, we cannot easily predict its life. It evolves forever.

4 Contradictions between Hardware and Software

As discussed above, there are two major contradictions between hardware and software development. Hardware is produced with fixed functions and it is an individual living, while software evolves with time and it is a species.

Although most of our products are combinations of hardware and software, there are very few discussion on how we can reconcile them and work them together effectively.

This becomes a very important problem when we have to decommission a system which is composed of hardware and software. It must be noted that sometimes the degradation of hardware is “remedied” by software. Therefore, life prediction of a system composed of hardware and software combined becomes increasingly difficult.

Further, we should note that in practical applications, software does not evolve forever. In most cases, hardware on which software runs are replaced with new one and this decommissioning of hardware really defines the life of a combined system.

But currently hardware engineers and software engineers do not work together so well. Rather, they work independently from each other. They need more collaboration to really produce a better combined system. This is the point I should like to emphasize in this paper.

5 What Solutions May be Possible?

Then, what solutions may be possible? We have to remember that hardware is also changing. IC tip technology brought us a big change. Hardware comes to evolve as a system. Too.

Therefore, we have to consider a hardware- and software-combined system as system of systems.

But what is a great advantage of hardware is that it is “a thing in the physical world”. With too rapid progress of software, we tend to forget that we are living in a physical world. We sometimes misunderstand that we are completely living in a non-physical world. But where a human lives is a physical world.

Thus the problem of coordinating hardware and software is nothing other than the one how we can coordinate our physical world and our non-physical world.

Our physical world is bounded in many ways. Its space is limited and its resources limited as is well known that our world resources are running out., etc.

But our non-physical world is boundless. It can expand to infinity. Thus, the issue is how we can compromise the finite and bounded world and the infinite and unbounded one.

But we are living in a physical world and our life is finite. For example, an organization or a company pursues to proper forever. Survival is their ultimate objective. But the people who work there changes after about 30 years. Generations change.

Then, would it be really worthwhile to develop software that works forever? If the generations change, their ways of thinking or their ways of solving the problem would change. Therefore, it would be far wiser to develop software that works best for this period of time at the maximum. Of course, the situation changes are more frequent so that we have to add on more new pieces to increase adaptability. If the people in an organization changes widely, then it would be time for decommissioning the system. Software is the collection of the brains of these people but if these people are no more with the organization, there will be no more people who can make adequate judgement about the system outcomes.

In fact, people can not understand software with too many add-ons. If they cannot understand it, it is the time for decommissioning.

But what happens if hardware degrades much faster than software. In fact, in most cases, software is thrown away because hardware is replaced. Thus, the problem will be how we can decommission them at the same time. This could be possible if we change our hardware development from the fixed function products to evolving function ones. In short, we will develop hardware as a system. There have been hardware systems. But these systems are composed so to speak in a parallel way. There was no communication between the parts. But we have to change its design so that each part communicates with others to evolve. Some parts could be decommissioned earlier if other parts could take over their functions. Thus, new hardware system works on a complementary basis. Each part or component communicate with other ones so that they can maintain functions as a whole system. The function may not evolve, but it certainly adds adaptability and flexibility to the system. The hardware system survives in changing situations and maintains its original desired functions. Thus, we should change our hardware design to be more system-oriented and adaptive. Hardware elements will serve for the designed purpose as a system to interact with a physical world.

We have to remember again that we are living in a physical world. So if such a hardware system does not work anymore, it is the time to replace the whole software- and hardware-combined system. We can add any adaptability and flexibility as software, but it is within a non-physical world. If we come back to our basics that we are living in a physical world, the system that would not fit anymore with the physical world would lose its meaning. And efforts to evolve software would not be worthwhile.

6 Hardware, Software and Humanware: Integrating Them into One

Hardware and software are contradictory. Hardware is produced with fixed functions and degrades with time. And it is something like an individual living. Software evolves with time and it is something like a species. So how we can work together effectively will pose a big issue because more and more products are produced by combing them.

The problem of decommissioning a hardware- and software-combined system would be the following. If we come back to the basics that we are living in a physical world, then it would be an adequate decision to decommission a system, when a physical part or hardware part of a system would not adapt to the situation anymore. This does not mean the present framework of hardware systems. In the new framework, hardware elements or components will communicate to complement with each other in order to maintain adaptability and flexibility. Software and IC tip technology will help a great deal for this purpose.

It should be noted that in this new framework, software is not just software and works alone. Its very important new role is to help hardware elements communicate with each other to maintain the desired functions as a whole. So in the new framework, the whole system works as shown in Figure 3 by integrating software, hardware and humanware into one. Each subset, i.e., software, hardware and humanware, works for the whole system (intersection) and when the whole system would not adapt to the surrounding environments, we will announce it is the end of life of the whole system.

7 Reference

[1] Shuichi Fukuda, editor, Feature Issue. Frontiers in Reliability, Trans. Institute of Electrical, Communication and Information Engineers, Dec. 2006 (in Japanese)

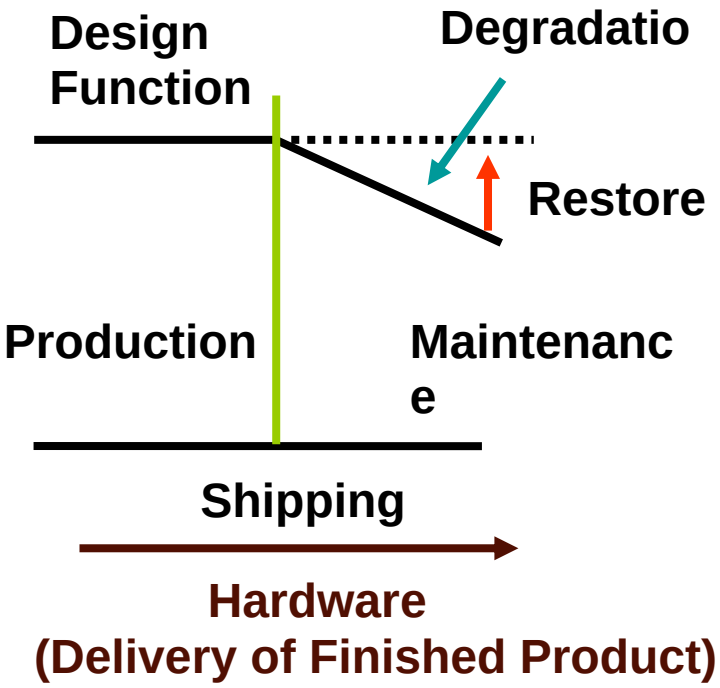

Figure 1 Hardware development

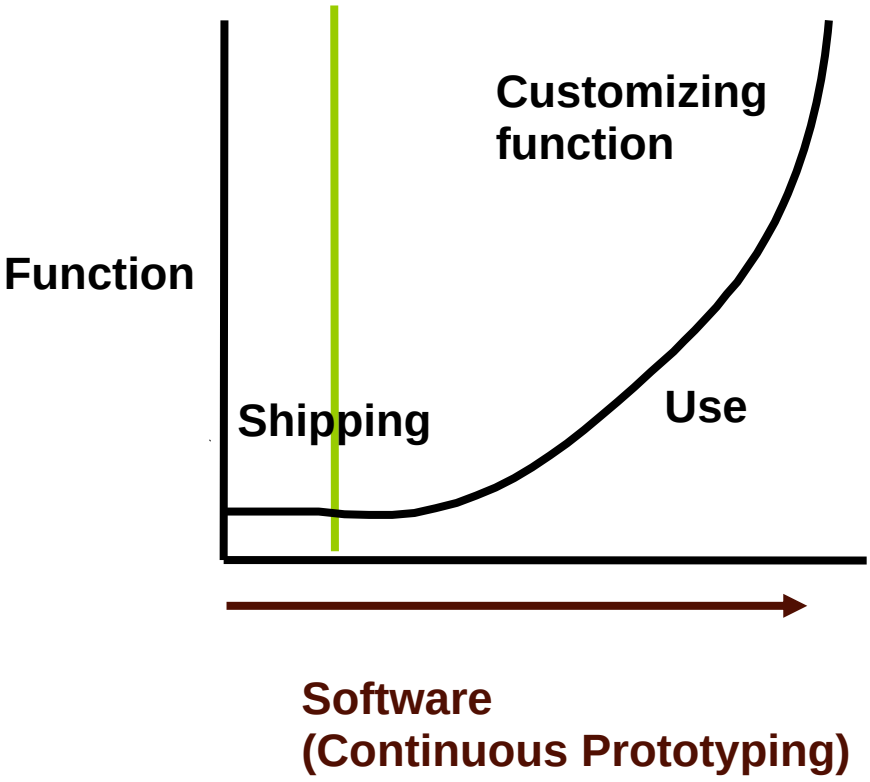

Figure2 Software development

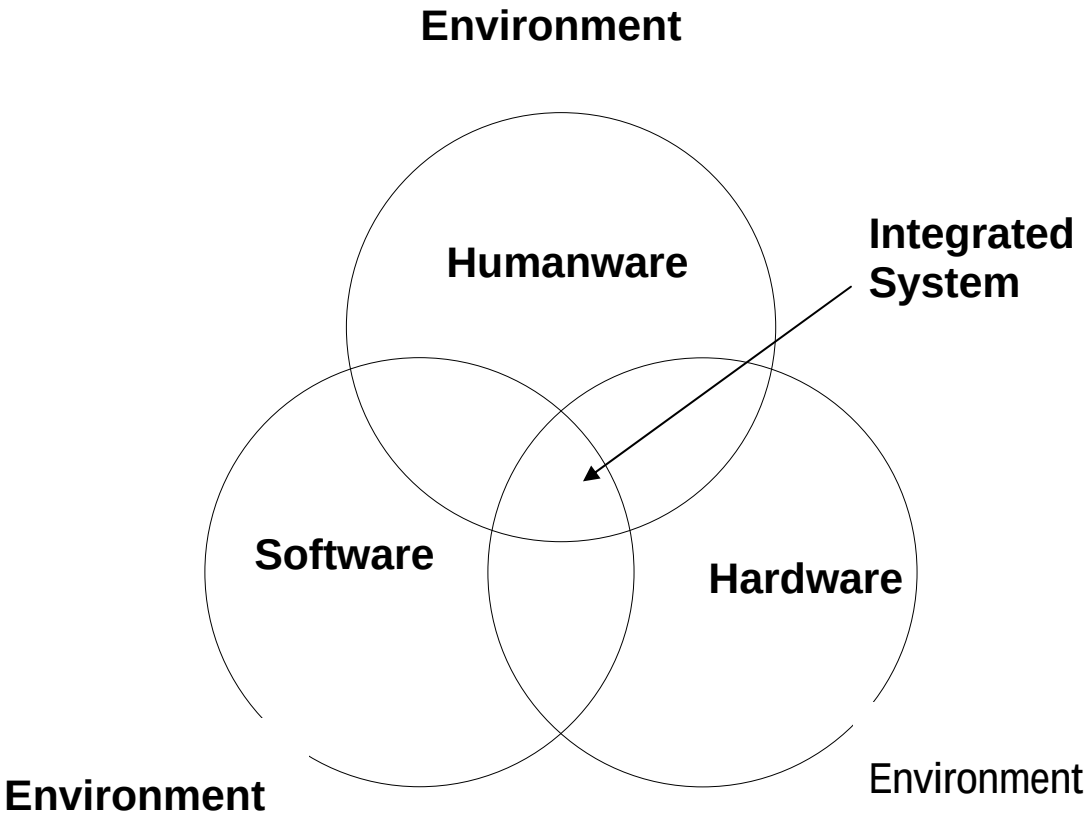

Figure 3 Hardware, software and humanware integrated into one system