

An Evolutionary and Comparative Perspective on the Japanese Enterprise Software Industry*

Revised and Final Version 21 Dec. 8, 2009

Robert E. Cole

Visiting Fellow, ITEC, Doshisha University
Professor Emeritus, Haas School of Business, UC, Berkeley

Fushimi Shinya
General Manager, Strategic IT Business Planning
Mitsubishi Electric Corporation

*We are indebted to Takai Shinji, Doshisha University, Yamagishi Hideyuki, Oracle, Japan and Josh Greenbaum, Principal at Enterprise Applications Consulting, Consulting and Oyama Gen and Sekiguchi Kazuhiro of Mitsubishi Research Institute for their views and suggestions. We alone are responsible, however, for the analysis included in this article. This paper benefitted from research support from the Institute for Technology, Enterprise, and Competitiveness, Doshisha University, Kyoto, Japan and the Center for Japanese Studies, UC Berkeley.

** To appear in Yoshifumi Nakata (ed.) Has the Japanese Enterprise Changed, MacMillan's Asian Business Series (British).

Software is the main driver of information technology and is critically linked to communications and computers. Hardware increasingly depends on software for its functionality and software needs hardware to express its capabilities. Software is critical both as an industry itself but also as infrastructure enabling all other industries. It drives the spread of information and communications industries and the Internet in increasingly knowledge based economies. How the two largest economies, Japan and United States, have organized their enterprise software activities is central to understanding the dynamism of both economies.

Contemporary observations on the U.S. and Japanese software industries reveal strong differences in firm strategy, organizational structure and processes. To be sure, there are those who stress similarities among leading users of IT (Rapp, 2002) but the dominant image is one of differences. What are some of the differences to which we refer? They include, but are not limited to: a large packaged software sector in the U.S. and a weak one in Japan, more horizontal markets for software products in the U.S. versus more vertical markets in Japan, a greater role for firms producing software products in the U.S. versus services in Japan, a greater role for startups in the U.S. versus the continuing dominance of large incumbents in Japan, a stronger emphasis on quality in the software development process in Japan versus innovation and time to market in the U.S., a more consolidated market for system integrators in Japan than in the U.S. How do we account for these differences.

These differences, many of which are linked, are all the more remarkable, since at the start of the computer revolution some 50 years ago, there were some important similarities in strategy, structure and processes. Nevertheless, first mover advantages, different institutional practices, and the interests and strategies of powerful actors moved evolution of the two industries in quite different directions. Our task in this paper is to trace what became increasingly divergent evolutionary paths, given some common starting points and to provide some accounting for these differences. At the same time, we identify punctuated pressures for convergence over the last 60 years. Dosi and Kogut (1993: 249-260) point us toward understanding these contradictory movements by describing a cycle of divergence in the organizational structure and processes of common national industries due to the introduction of new organizing heuristics, followed by gradual convergence resulting from the diffusion of these new heuristics across national borders, with the process repeating itself over time.

Because of the breadth of the software industry, we have chosen to focus on two linked segments in Japan, those computer manufacturers and their subsidiaries that deliver *system integration services* for *enterprise software* to their corporate clients. The strong need for

system integration services is driven in part the heavy use of customized enterprise software especially in Japan. Most Japanese corporate software is still for mainframes and midrange computers and large firms account for most software use. Customized software for corporate users accounts for 85% of the total Japanese software market (Ministry of Economy, Trade and Industry (METI: 2007; see also Rapp: 2002:48). We aim to show that the prominence of this sector and its functions are key to understanding the lack of a strong packaged software industry in Japan as well as the persistence of incumbents over time and the corresponding relative absence of strong venture capital/start up activities.

We will proceed as follows: Section 1 provides an introductory description of the system integration market in the U.S. and Japan. Section 2 reviews the emergence of the computer industries and subsequent rise of the software sector in both Japan and the U.S., aiming to capture both differences and similarities. In section 3, we turn to the rise of factory models of software development showing their differing impact in Japan and the U.S.

Section 4 describes the transformative developments around the turn of the century that imposed strong pressures for Japan's convergence with global practices. These pressures arise from the emergence of open architecture and open networks. Section 5 takes up the striking different ways U.S. and Japanese firms build their information systems, Japanese firms have a much higher propensity to develop their own software or to order custom software compared to U.S. firms which are more open to packaged solutions.

Section 6 details how system integration firms deliver customized solutions to their customers, highlighting the continued importance of keiretsu boundaries in Japan. In section 7, we analyze the impact of the dominance in Japan of customized software for international software trade and examine why it is so difficult to grow the packaged software market in Japan. Finally, we explore the pressures on Japanese firms to move more towards packaged and semi-custom solutions in the future.

The System Integration Market (1)

By system integrators, we refer to those firms which specialize in planning, design, implementation, and project management to join together component subsystems into a whole and makes certain that these subsystems work together in a seamless fashion. In the IT world, this often involves customization of innovative software applications from different vendors. System integration may involve the process of configuring newly acquired applications software to fit into the technology infrastructure and business processes of the company. It may be as basic as filling out the Bill of Materials (BOM) for Enterprise Resource Planning (ERP) or as complex as the integration of an ERP solution into a firm's existing IT system.

As mainframe computing of large organizations in the 1980's grew more complex, corporate IT capabilities came under increasing pressure technologically and from rising costs.

Internal development capabilities, and in particular, the ability to keep pace with fast moving technology, were called into question. As a result, U.S. firms began to explore external support. To meet this need, established companies delivering system solutions like Computer Sciences Corporation and EDS as well as four of the big eight accountancy firms came to offer system integration services. Delivery of these services requires close cooperation with the client firm (Steinmueller, 1996:39-40).

In the case of Japan, the major computer/software vendors in the 1970's, adopted and continue to follow to this day, the business model of the IBM mainframe era. Initially, this was to sell both hardware and software to client companies. Later, this evolved to selling them custom application systems, wrapped in hardware and software and then to add on to this initial contract, consecutive deals for maintenance, administration and upgrades for as long as possible. This makes them primarily information service rather than product companies. These firms (including units within large firms), in the 1990s, came to be called system integrators. Japanese information service firms totaled under 1,000 in 1985; that number jumped to 7,000 by 1990. The number of firms then declined and flattened out for a number of years after that, rising again starting in 1997 and coming to total a little short of 8,000 by 2002. Sales of information service firms, however, showed a steady growth from 1985 rising about 800% above the 1985 level by 2002. System integration firms accounted for about 43% of total information service revenues in this same year (JPSA, 2005:19-20).

Table 1 shows the top 10 worldwide system integrators in terms of market share. When we compare the top two U.S. firms, Accenture and IBM, (ranked one and two respectively) with the top two Japanese firms (ranked three and six respectively), we find that the bulk of sales of the two Japanese firms were in Japan while IBM and Accenture focused more on global markets.

Table 1

Top 10 Worldwide System Integrator Market Share Leaders in 2005

Rank	Vendor	Headquarters of Vendor
1	Accenture	U.S.A.
2	IBM	U.S.A.
3	Fujitsu	Japan
4	Lockheed Martin	U.S.A.
5	CSC	U.S.A.
6	Hitachi	Japan
7	SAIC	U.S.A.
8	SAP	Germany
9	Capgemini	France
10	Northrup Grumman	U.S.A.

Source: Torto, 2006.

Table 2 reports the top 10 consulting and system integrators' revenues and market share in computing in the Japanese market for 2006-2007. Fujitsu scores the highest revenue at

4.9 billion dollars in 2007 and a market share of 15.8%. By comparison, Accenture, the leading U.S. system integrator, had global system integration revenues of 6.3 billion dollars in 2005. There has been growing concentration over the last 15 years in the global system integration market (OECD, 2002:122) The Japanese system integration market, remarkably, is the most consolidated of all the worldwide regional markets with an estimated 73% of total system integration market share held by the top 10 providers (Karlsson, 2008). Compared to other world regions, potential new entrants to the Japanese system integration market face larger barriers to entry (OECD, 2002:122).

Table 2 also provides estimates of the total market revenue for the Japanese system integration sector for 2006 and 2007. For 2006, it is estimated to be 29.7 billion dollars. To get a sense of the role of this sector in the total IT service industry, we can draw on data from METI's Annual Survey of the Condition of IT Services Industry, 2006 (METI, 2007a). The entire industry reported revenues of 143.8 billion dollars in 2006. Twenty eight percent of that total is accounted for, not by software development revenues but, by revenues from data processing services (METI, 2007a). If we eliminate this latter portion, then system integration services impressively accounts for roughly 29% of all software development revenues.

Table 2

Japan: Top 10 Consulting and & System Integrator Providers' Revenue and Market Share in Computing: 2006-2007

(in millions of dollars)

Rank	Vendor	Revenues 2006	Revenues 2007	Growth (%) 2006-2007	Market Share (%) 2007
1	Fujitsu	4,849	4,938	1.8	15.8
2	NEC	4,146	4,311	4.0	13.8
3	Hitachi	3,668	3,913	6.7	12.5
4	IBM	3,466	3,555	2.6	11.4
5	Nomura Research Institute	1,067	1,220	14.3	3.9
6	NTT Data	1,021	1,052	3.0	3.4
7	Toshiba	987	1,047	6.1	3.3
8	Fujitsu	986	965	-2.2	3.1
9	Mitsubishi Electric	866	882	1.9	2.8
10	Nihon Unisys	657	858	30.7	2.7
	Other Service Vendors	8,007	8,547	6.8	27.3
	Total Market	29,720	31,289	5.3	100.0

Source: Karlsson, 2008

Early Developments (2)

We turn now to the early stages of the commercial computer industry in Japan and the U.S. The origins of the American computer industry during and immediately after World War II lay in military and scientific activities. By the 1950s, interest turned toward business objectives. In the period from 1950 to 1965, software was not a separate business for computer companies

but rather a service provided by these companies to sell their hardware. Consistent with this approach, software was a product whose costs were bundled with computer prices (Steinmueller, 1996: 19). Many of the new companies in this field, IBM, Burroughs, Remington Rand and NCR originated as business service firms and they came to understand that more and better software sold more computers.

The early development of the Japanese computer industry followed a similar pattern, minus the military origins, and with a lag. Strong interest in the new computer industry by commercial companies did not take place until the mid- to late 1950s by what were to become the three major players: Fujitsu, NEC and Hitachi (see Fransman, 1995: 130-137). These were all telecommunication companies and dedicated suppliers to NTT, the telephone monopoly. As such, they had secure businesses that made them somewhat slow, especially Fujitsu, to venture into this new world of computing with its new way of thinking about information collection, analysis and deployment.

As a dominant first mover, IBM (70% of the global value of computer sales in 1956 and 62% as late as 1971) was both the model to be followed and the target for Japanese hardware manufacturers. This induced important commonalities between the two industries in this mainframe era. Particularly with its creation and release of the IBM 360\370 series in 1964, IBM product architecture became the emergent global industry's de facto architecture. It delivered a family of compatible computers with backward and upward compatibility for the first time in the new industry. It minimized the need to develop new software for each new computer. Firms throughout the world were now playing by IBM rules. That meant they were making devices, writing software, manufacturing clones, running time-share centers, all within the computing environment that IBM defined (Ferguson and Morris, 1993: 15). The launch of the System 360 family of computers in 1964 established the first broad installed base of computers with a single operating system, thereby enabling independent software vendors to market the same product to a variety of users.

A third factor stimulating software sales by independent software vendors was IBM's decision to unbundle software from hardware pricing. Following a 1956 consent decree, IBM, fearing further anti-trust action by the U.S. Dept. of Justice, decided to unbundle software from hardware pricing. This 1968 announcement led to a momentous shift that firmly launched an already nascent independent software industry. Initially there was rapid growth of new and existing companies producing IBM compatible software, especially application software. Early on, computer vendors provided tools for applications development. There was consequently an enormous growth of user developed application software in the U.S. in the 1960s. Those firms developing their own software for their own proprietary uses, however, had little incentive to sell that software to others. Consequently, there also arose in the U.S., strong independent software vendors (ISV) for whom software was a business. It is between 1965-70 that we witness the real beginning of the U.S. independent software vendor (ISV) industry (Steinmueller, 1996:17-24).

Gradually, the U.S. computer hardware companies came to believe that their ability to understand and solve user problems was limited and that they should specialize in hardware. The 1980s witnessed a sharp decline in software and services revenues as a percentage of total revenues for computer firms. A number of computer companies like DEC abandoned their vertical market strategies which had established them as leaders in providing integrated hardware and software for selected industries. Instead, they concluded they could gain more by concentrating on hardware innovation and, leaving software development to others. IBM was, however an exception to this development in that it did not receive a diminishing share of their revenues from software production. IBM focused on those software products that strongly raised the level of hardware demand such as those products that stimulated the sales of computational or mass storage hardware (Steinmueller, 1996: 17,30-35,45).

The Japanese were just as much prisoners of IBM's architecture as their American competitors. The Japanese struggled mightily and eventually rather successfully to catch up with IBM's performance by the late 1970s. Inspired by IBM's global success, which was attributed to economies of scale, scope and vertical integration, the leading Japanese computer vendors organized as vertically integrated firms (Fransman, 1995:148-149). Consistent with IBM, but unlike other U.S. computer firms, the Japanese computer vendors sought to keep software central to their mission, either through internal software divisions and/or captive subsidiaries. How they came to do that, however, was quite distinctive.

Belatedly following IBM, Japanese computer vendors formally unbundled software from hardware in 1977. This spurred the creation of spinoffs from user companies which were early leaders in new uses of computing (steel industry spinoffs were an early example followed in the early 1990s by financial sector spinoffs). These user firms spun off their management information systems (MIS) divisions because they were becoming too expensive for the parent company. The new subsidiaries continued to do work for the parent company, hire new engineers for programming and other tasks but pay lower wages than the parent firm. Starting in the 1970s, Japanese computer vendors took an active role in developing software for users, quite the opposite of the overall trend in the U.S. This required the employment of large numbers of system engineers who were often dispatched to user firms' data processing departments for extended periods. These user firms became dependent on their services. As demand increased, those dispatched employees were assigned to separate internal divisions of the computer vendors or spun-off as external captive subsidiaries (Baba, Takai, Mizuta, 1996: 116).

The captive software firms of the major computer vendors were invariably members of their parent firm's vertical keiretsu. Some computer vendors like NEC and Fujitsu preferred to organize much of their software development capabilities in internal divisions (though they eventually established software subsidiaries as well) while others like Hitachi emphasized creating captive subsidiaries. In both cases, their target customers were typically their parent firm, other group members, and their customers and supplier. This led to the formation of vertical markets. The links between computer vendors, their software development activities (either by internal divisions or captive subsidiaries) and user firms tended to be long term and

stable, especially in the mainframe era. Ronald Dore named these distinctive contractual links, “relational management” (Dore, 1987: 173-192).

The computer maker spinoffs and user spinoffs grew expert and quite accustomed to churning out code to strict contractor specifications. They choose low profit but secure business relationships in the keiretsu family to the alternative of more risky, but potentially high profit activities, developing innovative software products for the broad corporate or consumer market. Of course, since they were often under the control of their corporate parents, they had little choice (cf., Hamilton, 1993:B4).

What were the initial origins of these vertical divisions? In the early 1960s, MITI sponsored multiple alliances between the selected fledgling Japanese companies and overseas producers as a way for Japanese firms to catch up technologically. As a result, each of these emergent computer firms had a different source for their operating systems. Hitachi became allied with RCA and then when RCA exited the market in 1971 became IBM compatible; NEC became allied with Honeywell and was non-IBM compatible, Toshiba with RCA and then General Electric, Mitsubishi Electric with TRW and Fujitsu was on its own and then later became IBM compatible (Rapp, 2002:49).

The operating systems of IBM compatible producers were able to run a new customer’s existing IBM programs. However, the operating systems of even IBM compatible producers were sufficiently different that the application programs written for one system would not run on the operating system of other IBM-compatible competitors, much less non-IBM compatible producers. The computer firms made little effort to insure interoperability, given their objective of insuring that customers stayed locked in for maintenance, upgrades, application development and system integration. It was these relationships that were to later solidify into the current vertical divisions of the computer and software markets that locked major corporations and their vertical keiretsu into selected proprietary hardware and software systems.

Factory Models of Software Development (3)

Initially, both IBM as the dominant U.S. player and the large Japanese mainframe companies were confronted with the challenge of organizing software development. By the 1970s, considerable discontent developed among U.S. producers and users over the handicraft nature of the industry and the problems this created for software development including lack of standardization, poor productivity and quality, cost overruns and failures to meet new product launch dates. We draw heavily on Cusumano’s analysis (1991; 2004) of responses to these challenges.

The software factory model was based on efforts to apply to software the strategy and tactics and practices used in hardware manufacturing. Software development in the early years of the industry utilized programmers operating more as creative artists than as disciplined

production workers engaged in a standardized development process with standardized tools and methods. As a result, costs were high and quality low.

In U.S, firms like IBM, General Electric and AT&T in the 1960's made notable efforts to apply to software, concepts and practices developed in manufacturing. IBM, as the most successful firm engaging in software development, faced the challenges of managing software development earlier than most. Yet, as late as 1970, it lacked a firm-wide coordinated system for development, quality assurance and maintenance. With a lack of centralization, each of the world-wide centers was often free to develop its own tools and methods defeating standardization efforts.

Gradually, these problems came into focus and IBM sought to address them using much of the language of what was to be called (not by IBM) the software factory. For example efforts were made in the late '60s and '70s for developing general purpose tools rather than product specific tools, and efforts in the late '70s were directed toward organizational and technological integration. A variety of process innovations were instituted as well such as top-down hierarchical design (Cusumano, 2004:101). Further efforts were made in the late 1980s to standardize hardware and software internal architectures and interfaces thereby enabling greater reusability of systems and tools as well as whole programs. Japanese software managers often claimed to have been inspired by IBM's efforts to impose structure and discipline in the software development process.

The System Development Corporation (SDC) made the most concerted effort to impose the factory model on software in the U.S.; it devoted several years to planning and experimentation in the 1970s as they sought to put their software factory into practice. SDC opened its software factory in Dec 1976 with the experiment coming to an end after just three years. The reasons given for its failure are many. In particular, according to Cusumano (1991:136,157-160), it fell short in managing requirements specifications and changes, creating design and verification tools, and enabling the reuse of software more systematically. Underlying these weaknesses was the lack of fit between its product line emphasizing innovation for a variety of customers and the standardization focus befitting the software factory. This made it difficult for them to focus on process improvement. The emphasis of U.S. firms on product and Japanese firms on process innovation is a patterned difference which has long been observed in comparisons of U.S. and Japanese manufacturing (Mansfield, 1988).

Notwithstanding, there were important positive legacies to these efforts in design, quality assurance and project management as manifested in particular in the U.S. defense and space programs. A well known offshoot of the software factory concept emerged in the founding of the Software Engineering Institute (SEI) at Carnegie Mellon University in 1984 by the U.S. Dept. of Defense. The Institute's objective was to codify and disseminate best practices in software development. Cusumano notes that most of the practices recommended by SEI arose from IBM's experiences at creating operating systems and complex applications. SEI became particularly well known for its delineation of the stages of maturity for software development practices.

In spite of these legacies, the Americans fell well short of the fully implemented software factory. The reasons seem to lie primarily in their emphasis on product innovation and time to market. These emphases eventually served the U.S. producers well, leading to the creation of a vibrant ISV sector specialized in hugely successful packaged software businesses. This strategy was enabled in the software industry because, unlike in manufacturing, process improvement, while important in the software development process, is a non-issue in software “production.” That is to say, unlike hardware, software is not manufactured. That is, the costs of producing the first copy of an information good can be substantial but the cost of replication is minimal (cf., Shapiro and Varian, 1999:3).

The Japanese began their efforts at creating the software factory just a few years after IBM had begun to organize its own software operations. The major Japanese computer manufacturers-Fujitsu, NEC, Toshiba and Hitachi and Mitsubishi Electric, all made long term commitments to building the software factory. The overriding objective of these Japanese software factories was not to make products sold on the open market but to customize software for internal divisions or corporate parents who then sold the output as complete hardware and software solutions to corporate end users.

Creating software factories for them meant, above all, the strategic management and integration of software development along with achieving economies of scope among related projects (e.g., through reuse of software modules). On a tactical level, it meant emphasizing incremental innovation in feature design, a strong focus on getting user needs built into the initial design specifications, standardized development techniques, common training programs, reusable component libraries, computer-aided support tools and adherence to strong quality control policies and practices driven by a “quality first mentality,” (cf. Cusumano, 2004:8).

Most importantly, the Japanese showed a commitment to creating the software factory that went well beyond American efforts in terms of resources allocated to these efforts. Consequently, Japanese software factories made considerable progress as measured by their strong quality and productivity performances relative to international competitors (Cusumano, 2003).

The leading Japanese captive software firms and the software divisions of corporate vendors were largely organized into “software factories” whose mission was to develop custom IBM or IBM-compatible mainframe software for the large computer hardware vendors and their large end user corporate customers. These “products” met the needs of these corporate customers for software systems and customized applications which supported their existing organizational practices. In prioritizing these objectives, enterprise software in Japan came to be seen not as an independent industry, but more exclusively as infrastructure support for large corporations. Software came to be treated as more of a service than a product. Just like IBM, the objective of large Japanese software producers was to use the creation of software to sell hardware (mainframes) and services (e.g., maintenance). Stability and performance of core functions, rather than innovation, are the hallmark of their software systems. Notwithstanding, even the Japanese fell well short of creating the software factory, in large part because of the

difficulty of confirming in advance software specifications in a dynamic market with rapidly changing customer needs and technology.

Rise of Open Architecture and Its Impact on the Corporate Enterprise Market (4)

At the turn of the century, the major challenge for Japanese software providers arose from emergent technologies based upon open architecture. This challenge was twofold: First there was open architecture in hardware, leading to downsizing from mainframe computing and reducing the total cost of system development. Secondly, there was open architecture in software, leading more companies to enter into the IT service market and enabling users to mix and match products. This is seen in the impact of Sun Microsystems' object-oriented software, JAVA, starting in the early 2000s. Java had a huge impact on software development; it eliminated the risk of vendor lock-in, enabling customers to assemble "best-of-breed" solutions. More recently open architecture's impact can be seen in the accelerating adoption of Open Source Systems (OSS) including Linux, Apache and PostgreSQL.

The problem for established Japanese software leaders was that their dominance in the mainframe era was based on long term vertical partnerships rooted in closed proprietary systems among business partners, often including even retailers. The mainframe vendors followed an "enclosure" strategy whereby they sought to have exclusive ownership of the entire value chain for their particular customer base. Their machines had proprietary hardware and software, proprietary interfaces with peripheral equipment and the computer vendors and their subsidiaries sought to provide customized applications for their customers (Kokuryo, 1997). These tight linkages among business partners were in fundamental ways incompatible with the opportunities opened up by the emergent open network era. Yet, understandably, dominant firms were slow to abandon the basis for their past success to embrace the uncertainty of open platform systems. Practically speaking, open architecture now allowed system integration companies to use a large variety of software and hardware to design a system based on each customer's specific needs. This was an enormous change from the mainframe era with its proprietary systems based on technological incompatibility.

Open architecture enables any software company, having the requisite design capabilities, to enter the market. This development helped break "vendor-lock in," and accelerated the pace of outsourcing. The proprietary IT technology held inside a given company came not to contribute as much to a firm's competitive advantage as it did in the mainframe era. This led to severe competition in the system integration market, thereby contributing to more equal

relationships between system integrators and end users, and leading to cost reductions (Cole and Matsumiya, 2006:9). Under these circumstances, users gradually found it easier to change vendors for a given layer or even all of their IT system, when they became dissatisfied with existing services. To be sure, these changes came much more slowly to Japan than the U.S. They have, by now, however, worked their way through a good part of the Japanese economy. Consider the following case of a major manufacturing firm's current deployment of its IT system:

Table 4

IT Deployment and Migration in a Major Japanese Manufacturing Firm

Business Function	Vendor(s)
R&D	IBM -> UGS + Fujitsu
Manufacturing Design	Unisys -> UGS + Fujitsu
Manufacturing System	Hitachi -> Partially went to NEC
Finance	IBM + SAP*
Sales Overseas (Main Markets)	IBM

*IBM was responsible for implementing and continues to maintain the SAP system

Source: E-mail communication from Yamagishi Hideyuki, Oracle, Japan Corporation, Oct. 16, 2008

We see in Table 4 the outcomes from replacement of mainframes and legacy applications with open servers and migrations to different vendors for different specialized applications. This is a big change from the mainframe era where end user firms were likely to have only one hardware and software vendor which served as a long term partner. Now many firms have a number of specialized long term partners. If the advantages of a competitor become large enough, they are increasingly prepared to change partners. To be sure, many firms still depend on one single system integrator vendor to build and maintain core and mission critical systems (e.g., mega banks). These firms may use other system integrator vendors to develop relatively "light systems" (e.g., e-mail, web, payroll and human resources management).

Ways of Building Information Systems (5)

We can broaden this discussion to address fundamental ways of building information systems. We begin with some definitions. Packaged software, narrowly defined, are those software products aimed at the mass market which require very little after sales support.

Network effects tend to be strong with those firms developing and distributing such products operating on a “printing press” model. If they can develop hit products, the rewards are very large. Packaged software is tradable software licensed to users by the software creator and holder of the copyright. The industry early in its history, produced a “shrink wrapped” product but it evolved and firms now use a variety of new distribution methods and channels. More specifically, it moved from a “bunch of tapes,” to CDs, then DVDs and now it is often downloaded from the Internet.

The narrow definition of packaged software, however, largely grows out of observations of the consumer mass market and doesn’t do justice to actual practices associated with packaged software in the enterprise market. For the enterprise market, we might rather see packaged software as a software product in which the majority of the functionality is pre-built, say 80%. That leaves another 20% which need to be customized during implementation. Packaged software is generally cheaper per user than highly customized software and much easier and less expensive to maintain and upgrade. When it comes time to upgrade, the vendor needs to write the change once and distribute it en masse to all its customers. The economies of scale are enormous and they are one reason why charging high maintenance fees can be so profitable.

Customized software, by contrast, involves an end user firm entrusting customized software development to a partner (often a system integration company), which develops the customized product based on the client’s needs. In a pure world of customization, the vendor would do something different for every customer. Unlike in packaged software, where the copyright is retained by the developer, in the case of customized software, the copyright for the developed software is often agreed to be attributed to the client firm (JPISA, 2005:6). While more expensive than packaged software, customization can deliver exactly the functionality desired by the end user.

How do the two nations compare in the distribution of these activities? First, we examine corporate end user data from 2004 commissioned by the Ministry of International Affairs and Communications as reported in Table 5. The most striking difference seen in this data is the large number of Japanese firms (54%) reporting that they develop their own or order custom software versus only 16% of the U.S. firms surveyed. This represents a huge difference in business practices. Alternatively, 43% of Japanese firms used some form of packaged software versus 78% of U.S. firms.

Table 5

Ways of Building Information Systems			(in percent)
	Packaged Software With Little Customization	Customized Packaged Software	Developed own or Ordered Custom Software
Japan	17	26	54
United States	29	49	16

Source: Ministry of International Affairs and Communications, 2004: Appendix: 340.

Data for 2006 from the METI Annual Survey of the IT Services Industry (Tokutei Sabisu Sangyou Jitai Chousa), as reported by JISA, reports only Japanese software firm activity in which software is the primary business activity. If we exclude data processing activities and firm's self-development of software, we find that 86.3% of sales are accounted for by customized software development versus 13.6% (a 6.3 to 1 ratio) by software product sales (JISA, 2008:4). We may deduce from these data that there is a heavy reliance on customization in the Japanese market and that packaged software products play a much more minor role.

Customization (6)

How do the leading firms conducting software activities organize their activities in the customized software arena and who are their customers? What follows first is a prototypical example of the organizational structure, practices and rationale of the "Big Five" Japanese enterprise software players based on our field observations. These players, as noted earlier, are: Hitachi, NEC, Fujitsu, Mitsubishi Electric and Toshiba. This example will not resemble exactly any one of the five firms but represents rather an "ideal type."¹ As one software specialist confirmed, "major Japanese software firms resemble one another in structure and practices."² As in many Japanese industries, the leading software firms have a full range of matched product offerings. The extensive use of the term "sogo" (general or full line) in Japanese captures these strong tendencies (Kokuryo, 1997:6). That is to say, there is relatively modest specialization at the firm level of software product offerings among the Big Five. Such comparable coverage by competitors is not absent in the U.S. in such fields as human

¹ Ideal type is used here in the Weberian sense. An ideal type is based on the characteristics and elements of a given phenomena such as organizational structure but it is not meant to correspond to all of the characteristics of any one particular case. The term is not meant to refer to a perfected version of the phenomena in question.

² E-mail communication from Prof. Takai Shinji, Doshisha University, September 3, 2008.

resources and financial software where regulatory requirements induce a similarity in offerings but the phenomenon is more pervasive in Japanese industry.

We begin our treatment of the prototypical Company X with a discussion of its Product Group A. Group A is a software “product center” for the system integration business. Although it is often formally organized and called a business unit, its “products” (such as database management software, application development tools, and job management software) are supposed to be used only in Company X’s system integration projects. Group A is ostensibly a profit center, but in reality it is managed as a cost center of the system integration business unit to which it reports.

The rationale for the above mentioned reporting relationship is that the enterprise software developed by Company X is plugged into large complex custom solutions for specific customers. These inputs are coordinated and combined by the system integration business unit. The solutions in question are typically vertically integrated solutions (e.g., Operating System + Database + Applications + Hardware). In this sense, the business application “product” developed by Group A is just a component (independent module) in this larger complex solution. Nevertheless, they are referred as products because “products” reference critical components of the system to be developed that have been already “productized.” To say that they have already been “productized” means that it has already been operationalized and used elsewhere and consequently there will be shorter development time and lower cost for building that module. In addition, these components are referred to as products because the firm wants to have the same set of “products” that IBM and other competitors provide. They also want to follow current fashion by saying, for example, that “our version 1.xx supports the Service Oriented Architecture concept.”

Put differently, to make an ideal product at low cost in this “software factory” approach, firms avail themselves of commonly used software parts as independent modules which they call “products.” These modules are then deployed for use in multiple end user products to provide economies of scope. It is critical, that these components are smoothly integrated with the larger custom solution.

The contrast with IBM is quite striking. IBM’s software group has brought to market many innovative packaged software products, tools and middleware that it passes over to its Global Business Services (GBS) group to be used as a factor input to which GBS adds a range of value added services associated with their customization work for end users. These products, like its DB2 database, compete directly with other enterprise software firm products such as those from Oracle and Microsoft. Yet, the GBS group also partners with these same enterprise software companies as part of its public commitment to provide customers the best possible solution to meet their needs. That is to say, IBM’s GBS unit offers competitor, as well as IBM products, to end users in their effort to deliver to customers the best solution. As a result, the

software group can not be sure of sales from global services unless they have cutting edge products.³

Company X's Group A, on the other hand, is primarily oriented toward meeting the needs of its System Integration Business Unit which, in turn, is responsible for integrating the total complex custom solution. So the System Integration Business Unit is their primary and sometimes exclusive customer. As a result, business applications, which in the U.S. may be tradable stand alone products such as an order entry module in an ERP system or application development software tools or even whole ERP and Database systems, tend to be seen quite differently in Japan. For the Japanese firms producing them, they typically are non-tradable components aimed at contributing to the whole solution package which will be sold to the end user. The system integration business unit provide a vertically integrated solution. As a consequence, the "products" created by Group A, customized as they are for a specific end user, typically are not marketed to other customers. Indeed, Group A doesn't have a strong marketing arm nor an aggressive sales force aimed at the external market. Software groups, operating in this fashion, have few incentives to produce innovative software which would be competitive on the open market. That said, there are differences among the Big Five. For example, Hitachi and Fujitsu's equivalents to Product Group A, are notably more aggressive in marketing its products to other system integrators.

There is also a software Group B at Company X. Group B's revenues are (typically) much smaller than Group A's. Group B is a true software product business unit, and operates as a profit center with relative autonomy. In some system integration firms, Groups A and B are in the same business unit. While a profit center, Group B does not, however, market its products to the external market nor to the consumer mass market. Rather it markets its products, enterprise software, to the Group A's of other major system integrators. The unstated assumption is that it is futile to try to sell directly to end user corporations. This is in striking contrast to much software sales activity in the U.S. Group B does have some marketing and sales capabilities but they are not strong. By contrast, U.S. companies like Oracle making innovative business applications, have succeeded in the U.S. and Japan by prioritizing the sales function.

The huge challenge facing Group B is that they are trying to sell their products to system integrators at other companies which typically offer a full range of their own Company X products. So unless Group B can develop a new breakthrough product or greatly improve an existing one, it is very hard to make sales to other system integrators. These other system integrators are their customers and these customers have every incentive to use their own products for their end user customers. Moreover, even if Group B can develop a successful product, the other system integrators will quickly act to "cover its hit." They also will do

³ Email communication with Josh Greenbaum, Enterprise Applications Consulting, Oct 24, 2008.

everything they can to persuade the end user to buy a related product from them, even if its functionality is more limited, or to wait until an equivalent product becomes available.

This brings us to an interesting feature of this system from a Western perspective. The pivotal role of system integrators in Japan, is such that software vendors rarely succeed, as discussed above, in marketing enterprise software directly to large corporate end users. While successful cases do occur, the end user corporations are usually heavily dependent on their long term system integrator partners to guide them in their IT selection choices. Heads of IT are on average relatively weak in Japan as seen in a comparison of the proportion of firms with a full time Chief Information Officer (CIO) post. It would be hard to find a firm in the U.S. with revenues over \$500 million in 2008 which did not have a CIO.⁴ The Mitsubishi Research Institute software specialists estimate that some 30-40% of Japanese large firms in 2008 had full time Chief Information Officers.⁵ In a 2005 publication, the Japan Personal Computer Software Association (JPSA) refers to “the absence or inadequacy of Japanese CIOs” in terms of their overdependence on system integration vendors when it comes to decision making (JPSA, 2005:33). In still another survey, it was reported that while 51% of American CIOs state that they participate in designing corporate strategy, only 13% of Japanese CIOs could make a similar claim (Ministry of International Affairs and Communication, 2003:Appendix: 340). Rapp claims that, in superior performing Japanese firms, CIO’s play a critical role in seeing that IT is fully integrated into business strategy (Rapp, 2002:27). That may or may not be the case in superior performing firms, but the data suggest it is hardly the norm among large firms.

Because most Japanese top managers don’t have confidence in their ability to evaluate IT choices, they rely on their system integrator(s) which knows every detail of their business processes through a long term relationship, and can guarantee a smooth implementation process without disruptions.⁶ From the system integrator side, therefore, they have a strong incentive to look for software solutions which prioritize stability over innovation (i.e., software systems which insure few quality problems and disruptions).

In summary, we can say that in Japan, customization work is the central activity of system integration firms done in collaboration with in-house teams and in some cases an external software vendor. More than in the U.S., there is an overwhelming bias among corporate managers that large scale business applications like ERP will require extensive customization, despite the added costs this imposes. From a Japanese system integrator vendor perspective, the ultimate form of a product (hardware or software) is one which is fully customized for a specific customer and that is exactly what customers have come to expect and demand. Software producers, inextricably tied to hardware firms, seem to have fully internalized the user mentality epitomized by Japanese hardware managers.

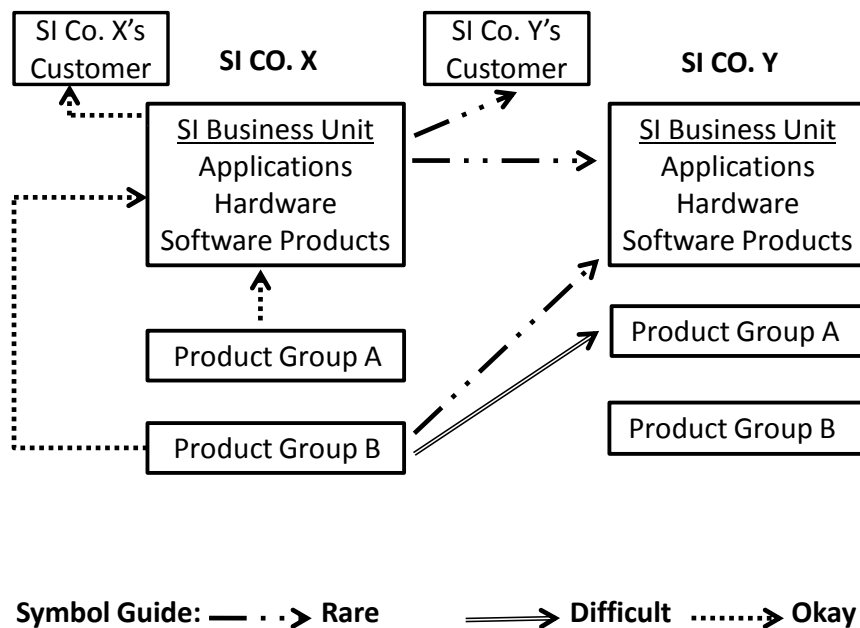
⁴ Ibid.

⁵ Interview with Oyama Gen and Sekiguchi Kazuhiro of Mitsubishi Research Institute conducted 6/18/08 by Fushimi Shinya.

⁶ Ibid.

Figure 1 summarizes the relationships we have described in terms of sales results achieved. System Integrator Firm X represents the prototypical system integration firm. Its Product Group A is focused primarily on supplying software components to its System Integration Business Unit which in turn integrates them into complete solutions for its customers. In relatively rare cases, efforts are made to sell directly to Company Y's system integration unit. Success is not common but when successful, Company X's system integration unit proceeds as follows. Instead of making direct sales efforts to Y's system integration business unit, it rather tries to influence Y's end user customers to use enterprise application z. If successful, the end users come to prefer it and tell System Integrator Y to use it in their system integration projects; the sale itself is to System Integrator Y. A much smaller Product Group B mostly sells its software products to Company X's system integration business unit but it also tries to accomplish the difficult task of selling its outstanding products to other Product Group A's at other system integrators.

Figure 1
Software Sales Between
System Integrator (SI) Firms X & Y
and Their Customers



We add a few additional observations regarding customization of enterprise software from an American perspective. As noted above, customizing a business package like ERP is often very expensive and complicated. Customization usually involves altering the code to meet

a specific company need. Some business packages have very generic features, such that a great deal of customization occurs in most implementations. Alternatively, all U.S. enterprise products, like Oracle's ERP, have increased configuration options which obviates the need for customization in some areas. Rather than changing the source code, exercising configuration options is more about settings tabs or parameters to meet specific customer needs such as setting up of payment terms or the length of the customer record.

Creation of a best practice capability embedded in enterprise software systems is based on modeling specific business processes among "lead users" and then creating a standardized best practice model for such things as order entry systems. A common view among U.S. software vendors is that customization is a main cause of cost overruns in software development and leads to increased costs in long-term maintenance and upgrades. In this view, all too often customization is made to satisfy a perceived need for company-specific functionality, despite that a more than adequate method/process already exists in the standard software routines. Indeed, a number of researchers in the U.S. cite avoiding customization as one of the critical success factors in successful ERP implementation (Lorenzo, Kawalek and Ramdani, 2009: 160). While long term relationships with system integrators certainly are in evidence, it is also the case that in the U.S., customization work is often undertaken on a time and materials basis rather than as an activity in a long term partnership.

In summary, much research over the last decade documents the declining significance of keiretsu in the Japanese economy, using as outcomes measures the diminished role of interlocking stock ownership and board of directors (Lincoln, 2004). In this volume, Lincoln reports that in recent years, strategic technology alliances in the electronics industry are less reliant on keiretsu ties. In the world of enterprise software, however, keiretsu networks continue to powerfully shape trading relationships among enterprises.

Packaged Software (7)

The OECD's 2002 examination of the global market for packaged software includes three types of packaged software: applications software (roughly 50% of the worldwide market), system infrastructure software (32%) and software development tools (19%). The leading software products in each category as measured by market share are for applications: cross-industry business Enterprise Resource Planning (ERP) systems; for system infrastructure software: operating systems and; for software development tools: database management systems.

Packaged software as tradable software dispensed with a license, by definition, is commercialized and capable of being extensively replicated and exported. There are a variety of quite complicated issues regarding how we measure software imports and exports. Nevertheless, the official data of business associations and governments can give us a rough idea of Japan's global competitiveness in software. In the year 2000, according to data from a leading industry association, the value of measurable exported application software (broad

definition) from Japan was a mere 5.1 billion yen. This compared to application software imports for the same year totaling 297.5 billion yen (JPSA, 2005:23). In the year 2004, the leading ERP supplier in Japan was the German firm, SAP accounting for 56% of the large firm corporate market in Japan; Microsoft dominated the operating system market while Oracle, IBM, and Microsoft had about 75% of the database market in 2005. Overall, foreign firms are dominant in the lower layer (i.e., operating systems, databases and other middleware) and in some parts of the upper layer (e.g., ERP).⁷ While the domestic Japanese markets of most advanced manufacturing industries are dominated by Japanese firms, we see that this is not the case in software.

Steinmueller suggests that the U.S. software industry's advantage in packaged software, not only over Japan but over the rest of the world, lies in the U.S.'s first mover advantage. It was an advantage enabled by government R&D policy and the early development of computer science education at leading U.S. universities. The importance of software for national defense resulted in generous U.S. government support for basic and applied research in software as well as for support for reconfiguring computer education at the university level (Steinmueller, 1996:42; Langlois and Mowery, 1996:53-85). The U.S. advantage was also greatly facilitated by existing institutional arrangements including a vibrant venture capital industry, and universalistic applications of contract law, securities law, employment and intellectual property law. All the associated institutional practices came with a set of uncommitted professional "outsiders" who stood ready to help new software entrepreneurs. This gave birth to a vast array of innovative start up firms supported by a very active venture capital community (Rtischev and Cole, 2003:146-148).

The contrast with Japan is stark. In the mainframe era, added value was overwhelmingly contributed by hardware; the government gave generous support to hardware vendors. As Japan entered the era of open systems, more and more value added came to be contributed by software. Nevertheless, government bureaucrats continue to see Japan's success as rooted in manufacturing hardware. Consequently, government support policies have remained fixed on supporting hardware development in large incumbent firms. In 2004, the budget for information technology related policies of METI totaled roughly 150 billion yen with software related items accounting for only 5% of that total. By contrast, hardware related R&D (especially device related) took up 75% of the budget (JPSA, 2005:16).

In Japan there are very few computer science departments in the national universities with most not established until the 1980s. The quality of their instructors was recognized to be well below U.S and European standards. The relatively few computer science engineers have not been well used and valued in an environment in which the primary activities in software engineering revolve around customization (see Baba et.al., 1996:122). Finally, unlike in the

⁷ Interview with Oyama Gen and Sekiguchi Kazuhiro of Mitsubishi Research Institute conducted 6/18/08 by Fushimi Shinya

U.S., there were few uncommitted professional outsiders and resources. Instead, we see a strong adherence to incumbent organizations and networks, buttressed by structural barriers inhibiting new, competing organizations and networks (Rtischev and Cole, 2003).

Beyond these macro observations, we can still ask why it is so difficult to grow the packaged software market in Japan given the seeming cost advantages over customization. Above all, with successful packaged software products, firms can amortize development costs over a large base of users? That large base of users combined with minimal reproduction costs is a recipe for high profits among the winners. Nevertheless, the data suggest that the incentives for developing packaged software in Japan are quite low. We can examine the matter first from the perspective of corporate users.

According to survey results accumulated by Mitsubishi Research Institute, corporations are reluctant to adopt packaged software in the applications software layer primarily for the following three reasons: most emphasized is the perceived inconsistency between the process routines required of packaged software applications and existing business processes. The second most common reason firms gave was insufficient quality. The third was loss of control. The latter is a reference to the fact that these products don't come with their source code and thereby limit uninhibited change possibilities.⁸

With regard to the perceived inconsistency between the packaged software required process routines and the existing business processes, managers fear additional costs after purchasing packaged software. These arise from the perceived need to customize the product to conform to existing organizational practices rather than changing organizational practices to conform to these software requirements. Nezu points to the fierce resistance encountered by those seeking to introduce organizational change to accommodate routines required by software packages (Nezu, 2002:21-22). In a study of U.S. and Japanese executives, conducted by NTT Data, 65 percent of U.S. respondents indicated a willingness to reform organizational practices to comply with IT system operations whereas only 42 of comparable Japanese executives were willing to do so (Cole, 2006:120).

Rapp (2002: 26) argues that in superior performing firms, in Japan and the United States, organizational adjustment flows from strategic decisions and improvement activities, not from IT deployment. That is, change decisions are made for their own sake rather than because of IT created opportunities (Rapp, 2002:26). This is a rather formulaic and limited view of how and why changes occur or don't occur in organizations. Surely, both can and should be potential drivers of change; firms benefit from "needs pull" adoption as well as "technology push"

⁸ Ibid.

adoption of new organizational practices. To rely on just one source limits the drivers of change and innovation. In earlier field research of a large successful manufacturing firm, we found that the Japanese managers often assumed their current organizational processes conferred competitive advantage to the firm. Specifically, these managers uncritically assumed that all current practices were conferring competitive advantage rather than critically examining which ones did and which ones did not (Cole, 2006).

Consistent with Nezu's observations cited above, managers are often unwilling to consider the possibility that some of the required process routines embedded in software applications could actually improve productivity. Decisions about whether to adopt as is, or to customize are often left to lower level managers, many of whom have strong incentives to maintain the status quo and/or to optimize local organization (genba) at the expense of the larger organization (Cole, 2006:105-126). It is quite possible that many U.S. firms suffer from the exact opposite problem, uncritically assuming that new software required process routines are best practice.

We can also examine barriers to growth of the packaged software sector from the software developer's perspective. Research among Japan Personal Computer Software Association (JPSA) members cite weakness of distribution networks (including difficulty of selling directly to large end user companies), difficulty of developing markets, difficulty of understanding and accommodating diverse user needs, speed of technological changes and difficulty of establishing competitive pricing (JPSA, 2005: 29).

There appears to be insufficient investment incentives for the production of high quality packaged software in Japan among larger firms, judging by their modest R&D expenditures as a percentage of sales revenues (JPSA, 2005:30). In discussing this matter, the JPSA notes that the investment of capital in software related venture companies in Japan is 1/30 of the U.S. and 1/14 of Europe. (JPSA, 2007:30). A reasonable interpretation of this outcome lies in the overwhelming bias of Japanese end users for customized software and a further bias toward working with long term partners rather than upstart newcomers. While the latter bias is far from absent in the U.S. it takes on even more power in Japan's vertically organized society (Rtischev and Cole, 2003). In understanding these outcomes, we must remember our earlier conclusion that the dominant Big Five software vendors have relatively little incentive to invest in uncertain new packaged software development when they have a secure revenue stream from customizing, integrating, maintaining, and upgrading services for their long term corporate customers.

Conclusion (8)

What is apparent from this discussion is that software as a product, or more to the point software as a business, is not that well developed in Japan relative to the size and strength of

its overall economy. To be sure, there are long term trends indicating the growth of packaged software and there are pockets of success, particularly in recent years, among producers of packaged software for small and medium sized firms. But for large firms, vertical markets and customized software still dominate. Given this dominance, we can say that enterprise software in Japan is primarily an infrastructure support industry. While there are high costs associated with that approach, there are also many pluses, in particular, solutions exactly tailored to end user needs. For software developers, however, these markets are low return markets compared to the increasing returns that can be generated by successful semi-custom or packaged software firms. Large software firms have chosen instead a low risk model with low but predictable returns from the custom software business based on successive deals for maintenance, administration, and updates. They are the kind of “gifts” that keep on giving.

What about the future? In a 2006 web survey of 1,078 IT professionals at large firms concerning enterprise applications, some 61% reported they were currently developing their software from scratch (fully customized) versus 19% who reported using commercial off the shelf packaged software (COTS) while 54% reported using semi-customized package software (multiple answers were allowed). When asked about future expectations, however, and allowing only a single response, only 31% expected to develop their software from scratch in the future and only 10% reported that they expected to rely on packaged software. Instead 55% expected to use as their main system, a semi-customized packaged software solution. That might well point the way to the future (MIJS, 2007).

The current pressures for this new direction arise from firms trying to minimize growth of their IT investments. Aggregate investment in IT has been relatively flat in recent years. Moving away from the costly approach of building software solutions from scratch can make a big contribution to holding down IT investment. Still another solution on the horizon is “cloud computing” which, by potentially giving final users direct access to software stored on the web, could diminish the size of IT departments and the role of system integrators. Long term growth of semi customized packaged software solutions will depend on whether firms become more willing to reevaluate the utility of current business processes. The unwillingness of many managers to question current business processes and instead assume they all confer competitive advantage is a major barrier to moving more strongly toward semi-customized package software solutions.

That said, despite initial divergence at the birth of the software industry, resulting largely from the U.S.’s first mover advantages, various technological developments over the last 60 years has created strong pressures for convergence. Three key such developments were: the development of IBM architecture, introduction of the Windows standard and more recently, the adoption of open architecture and open networks. Using IBM as a model, however, later put the Japanese software industry out of step with the evolution of a strong product oriented new venture sector in the U.S.; a sector composed of firms which targeted horizontal markets. Large Japanese system integration firms focused instead on providing services and products (modules) aimed at their narrow base of customers. Heavy customization, despite its high costs, gave large Japanese end user firms exactly the functionality they desired. Moreover, powerful

institutional forces in Japan led computer firms to adapt the IBM model of developing their own software, to the keiretsu system, thereby furthering divergence. Yet, more recently in the U.S., in a historic reversal that brings the U.S software industry closer to the Japanese model, computer firms such as Hewlett Packard have greatly increased their focus on software and services; Hewlett Packard now calls itself the 6th largest software company in the world. Conversely, Oracle **has acquired** Sun Microsystems, a hardware company. In short, consistent with Dosi and Kogut's theorizing, we see continued pressures for both divergence and convergence and it is not always Japanese firms which are moving toward U.S. practices.

References:

Baba Yasunori, Shinji Takai and Yuji Mizuta, 1996. "The User-Driven Evolution of the Japanese Software Industry: The Case of Customized Software for Mainframes," in David Mowery (ed.), The International Computer Software Industry. New York: Oxford University Press.

Cole, Robert, 2006. "Software's Hidden Challenges," In D. Hugh Whittaker and Robert E. Cole (eds.) Recovering From Success. New York: Oxford University Press.

-----, Tsuyoshi Matsumiya, 2006 "Capturing Value from Technology and Challenges to Past Successes: The Case of NTT DATA," ITEC Working Paper Series 06-01, Doshisha University. (March): 1-39.

Cusumano, Michael 1991. Japan's Software Factories. New York: Oxford University Press.

----- et. al., 2003. "Software Development Worldwide: The State of the Practice." IEEE Software (November/December):2-8.

-----, 2004. The Business of Software. New York: The Free Press.

Dore, Ronald, 1987. Taking Japan Seriously. Stanford, CA., Stanford University Press.

Dosi, Giovanni and Bruce Kogut, 1993 "National Specificities and the Context of Change: The Coevolution of Organization and Technology," in Bruce Kogut (ed.), Country Competitiveness, New York: Oxford University Press.

Fransman, 1995. Japan's Computer and Communications Industry. New York: Oxford University Press.

Hamilton, David, 1993. "U.S. Companies Rush in to Fill Japanese Software Void," Wall St. Journal, 7 (May):B4.

JISA, 2008. IT Services Industry in Japan, 2006, Tokyo: The Association.

JPSA, 2005. Package Software Market Expansion Research Forum Activity: Fiscal Year 2004 Report (Heisei 16 Nendo Pākeji Sofutowae Shijou Kakudai Kenkyūkai Katsudou Hokokushou). Tokyo: Japan Personal Computer Software Association.

Karlsson, Susanne, 2008: DATAQUEST INSIGHT: Top 10 Consulting and System Integration Providers, Market Share in Computing, Worldwide, 2007. Stamford, Ct.; Gartner.

Kokuryo, Jiro, 1997. "Information Technologies and the Transformation of Japanese Industry," Paper presented at the Pacific-Asia Conference on Information Systems. Brisbane: 1-15.

Langois, Richard and David Mowery, 1996. "The Federal Government Role in the Development of the U.S. Software Industry," In David Mowery (ed.) The International Computer Software Industry. New York: Oxford University Press.

Lincoln, James and Michael Gerlach, 2004. Japan's Network Economy. Cambridge University Press, Cambridge, UK.

Lorenzo, Oswaldo, Peter Kawalek, and Boumediene Ramdani, 2009. "The Long Conversation: Learning How to Master Enterprise Systems," California Management Review," 52 (Fall 2009):166.

Mansfield, E. (1988). "Industrial R&D in Japan and the United States: A Comparative Study," American Economic Review Papers and Proceedings, 78, (May): 223-228.

METI, 2007. 2006 Survey on Selected Service Industries.
<http://www.meti.go.jp/english/statistics/tyo/tokusabizi/index.html>.

METI, 2007a. 2006 Annual Survey of the Condition of the IT Services Industry (Heisei 18 Nen Tokutei Sabisu Sangyou Jitai Chousa). Tokyo: The Ministry.

MIJS, 2007. MIJS Research Report: Questionnaire Concerning Enterprise Applications. Tokyo: Made in Japan Software Consortium.

Ministry of International Affairs and Communications (2004). Kigyō keiei ni okeru IT katsuyō chōsa (IT Practical Survey of Company Management). Tokyo: The Ministry.

Mowery, David (ed.), 1996. The International Computer Software Industry. New York: Oxford University Press.

Nezu, Risaburo, 2002. "Perspectives and Strategies for Japanese Industry," Paper presented at the Conference on Prospects for Core Industries in Japan and Germany. Berlin: Fujitsu Research Institute and German Institute for Economic Research in Berlin.

OECD, 2002. OECD Information Technology Outlook. Paris: The Organisation for Economic Co-operation and Development.

Rapp, William, 2002. Information Technology Strategies. New York: Oxford University Press.

Rtischev, Dimitri and Robert E. Cole, 2003. "Social and Structural Barriers to the IT Revolution in High-Tech Industries," In Jane Bachnik (ed.) Roadblocks on the Information Highway. New York: Lexington Books.

Shapiro, Carl and Hal Varian, 1999. Information Rules. Boston: Harvard Business School Press.

Steinmueller, "The U.S. Software Industry: An Analysis and Interpretive History," in David Mowery (ed.) The International Computer Software Industry. New York: Oxford University Press.

Torto, Stephanie, 2006, Worldwide Systems Integration 2005 Vendor Shares: Top 10 Vendors and Industry for 2005. Stamford, CT.: Gartner.