

Introduction

<https://www.linkedin.com/pulse/why-japanese-software-industry-lack-global-ripe-ryo-shimizu/>

NOTES:

Origin of Japanese Software Industry

- Origin lie in government intervention and involvement
 - During era of economic growth in 60s
 - MPT and METI
 - Pushing protectionist regime in an effort to grow japan electronic and IT industry
 - Foreign competition out
 - Nippon Telegraph and Telephone Public Corporation (NTT)
 - Gov and state run corporation granted contracts
 - Giving contracts who select companies
 - Initially this aided software industry of japan
 - Kept employment rate high
 - Provided stability for those in electronics and IT sector
 - But had long term side-effects
 - Like the rise of galapagos phones in NTT docomo
 - Essentially unable to compete globally
- Japanese software industry with Construction company comparison
 - The government intervention on Electronics and IT industry had a overall ipact on health of the software industry
 - IT General Contractor
 - Share structural similarity to construction companies
 - Major IT contractors win big contracts and outsources the project to subcontractors
 - Business biddings for software contracts from MPT/METI regime took similar step to this
 - This process however, introduced adverse side effects
- Practices of labor-intensive industry
 - Business bidding for governments contracts meant
 - Winning large contracts, keeping labor cost low,
 - General contractors began colluding during bidding cycle and rotate the contracts
 - IT general contractors practiced backroom deals to avoid competition
- Waterfall methodology
 - IT General contractor model is most suited with waterfall
 - Prime vendor conduct hearing with customer

- Obtain their requirements and produce specific documentation
 - Hands off specification to subcontractors for implementation and finally deliver the project to client after testing
- Labor and time intensive
 - Projects that consume more labor and more time equate to larger profit for IT general contractor
 - This method stuck
- Reduced global competitiveness
 - Lacking of global competition due to 1st and 2nd factor
 - Since they avoid competition - they cant face real competition and not incentivised to be nimble nor fast deliver projects
 - Lack of competitiveness in software adverse impact electronics as software became a feeder industry
 - As software vendors started incorporating software features in product lines to distinguish from competitor Japanese software houses with waterfall methodology couldn't keep pace in terms of cost and time to market
- Environment not conducive for VC backed companies
 - Difficult to launch venture backed software business in Japan due to few IT general contractors sitting in the software industry food chain
 - Impossible to do business without their involvement
 - Keeping start ups out
- Reduced status of Software engineers
 - Software engineers reduced to manual labor status
 - Deviated system of secondment of engineers
 - Being assigned to different projects

- Introduction
 - Most software products are for domestic market
 - Little to no supply to foreign market
 - Tomoo Matsubara
 - Japanese software engineer lack creative idea for product
 - Maegawa
 - Productivity of software development hampered by multi subcontracting system that prevents quality and productivity
 - Few venture companies
- Reasons for lack of international competitiveness
 - Lack of strategy in industry
 - 1990s
 - Large Japanese computer manufacturer developed both hardware and software for computer and sold them
 - Software was treated as additional service of hardware
 - Many software firms were subcontractors of large computer manufacturers
 - Software is not priced
 - Client lack of awareness of software
 - Management personnel do not understand complexity of software
 - Most Japanese firms do not have development teams within companies, and cannot evaluate value of software
 - Software companies
 - Lack attitude in trying to create new creative systems
 - Lack of strategy is caused by software as an extension of hardware
 - Lack of talent
 - Evaluation or lack thereof has affected human capital management of small mid sized software firms
 - College students with dedicated IT or Computer science work for high salary firms like banks, brokerage, trading or even large IT firms
 - SME software firm rely on non traditional workers with no academic background to IT or those from vocational
 - Japanese software industry takes advantage of this situation
 - Use multi subcontracting system where small software firm exists
 - SME are subcontractor of large software industry and develop software on cheaper wages, leading to lack of international competitiveness of Japanese software industry
 - This system hampers improvement of quality of software and productivity in developing software
 - Those firms do not attempt to reduce number of working hours

- Industrial structure
 - This multi subcontracting system also prevents adopting new project development methods
 - Waterfall method is used
 - Requirements definition, design, coding, testing, operations and maintenance
 - Little to no feedback mechanism
 - But japanese clients do not always decide all systems at once
 - Gradually make decisions in the process
 - Hence development is modified and lengthening and increase in development costs
 - Leading to overtime work and poor negative impact on recruitment of potential workers in the japanese software industry
 - Methods of prototyping is also used
 - Agile development does not need requirement definition and design in the beginning
 - Divided into small functions, in one development process, only one function is produced
 - Instead of making minute design the teams decide the gradually
 - Putting priority on adjustment to change in plan
 - In japan waterfall method is still used
 - Why:
 - First fits industrial structure of japanese software industry
 - Original contractor implements the requirement and design and subcontractor implement coding and test making it suitable for waterfall method
 - Second bureaucratic attitude to japanese software companies
 - Role of design and coding is divided, when developed software is different from original design, blame is on coders
 - Contractors prefer waterfall
 - Third when team makes similar type of software, previous experiences id leveraged

https://www.heinz.cmu.edu/faculty-research/profiles/branstetter-lee/_files/branstetter_going_soft-1.pdf

- 3 Facts of this paper
 - 1. Innovation became software intensive
 - 2. Japanese firm rely less on software knowledge in IT hardware invention (produce fewer software inventions)
 - 3. Innovation performance of japan IT firm is lagging in software-intensive sectors
- Prevention fro japanese firms from using software advances
 - 1. Resource constraint argument, US firm have access to larger pull of talent
 - And Japan unable to offshore intensive software R&D
 - 2. Rooted in japanese manager to understand the nature of IT
 - Couldn't fully grasp ithe importance of software in IT hardware product
 - Unable to access foreign talent due to immigration laws,
- Flow of IT labor pool was 68% larger than Japan
 - Inflow became 3 times by 2001
- Import of workers and software offshoring became a huge advantage for US increase
- Alternatively
 - IT manager in japan failed to appreciate software

- Patent Study findings
 - Japanese firm file fewer software patents than US counterparts
 - That difference grown steadily since late-1980s and after the mid 1990s
 - IT has become more software intensive
 - Japanese IT firms are less-software-intensive sectors
 - Two reason for declines
 - Lack of Talent Pool
 - Slowness of IT firms to recognize the transformational nature of software
- Education system
 - CS field
 - US benefitted first mover advantage in government, R&D and early adoption of CS as a educational field
 - Most CS PHD level holders are those pursued by professional academics
 - Lack of demand by Japanese industry for engineering PHD overall and lack of software startups
 - Most CS phds from US are hired by industry and government
 - PhD contribution led to innovation and startups like Adobe, Google, Synopsys, Qualcomm etc
 - Leading to excelling in the software industry
 - Lack of Japanese education integrating state of the art CS professor and education into the STEM field
 - Out of date knowledge in CS field
 - Particular in the PC age, in the US, vs the mainframe hardware mindset influenced by IBM
 - Modest number of startup open to updated knowledge led to decline
 - Expansion of IT departments in 1980s
 - Hired former managers and executives of major companies including NEC, Fujitsu, Hitachi etc
 - But they were grounded in mainframe knowledge of hardware and software despite being increasingly replaced by PC, work stations, new languages etc
 - Not competent to teach university students state of the art software
 - Moreover CS degrees or education is less reputable compared to on job training in mid-1990
- Institutional and Structural Perspective
 - Weak corporate incentives to create software products
 - Annual revenue of private sector in 2007 for Japan was 1.03 while the US was 4.3 around the same timeframe

- Gartner report suggested that US firm spent 3 to 5 percent on revenue towards IT in the manufacturing sector while the number wasn't even 1 percent in Japan cases
- Japanese firms have focused on competing IT on operational effectiveness point and not on innovation or new revenue growth
 - Consequence of that in an age of IT - software, customers eagerly pay for high valued innovation, leading to modestly trained IT personnel and not highly trained software professionals
- Business application of IT in Japan is 17 years

Institutional and Structural Perspective

(Why incentives, not just talent, constrained Japanese software innovation)

- **Core argument: the problem was not absolute labor shortage, but weak institutional demand for software innovation**
 - Japanese IT leaders did not perceive a severe or binding shortage of software labor; instead, the industry lacked **demand for high-end software architects and product-oriented engineers**
 - Without strong corporate incentives to create scalable software products, firms had little reason to invest in advanced software talent or experimentation
Cole_Robert_The_Japanese_Softwa...
- **Software treated primarily as a cost center, not a strategic asset**
 - Japanese firms historically viewed IT investment as a means of improving **operational efficiency and cost control**, rather than as a driver of new revenue, products, or business models
 - This contrasts with U.S. firms, which used IT to reshape products, services, and entire business models
Cole_Robert_The_Japanese_Softwa...
- **Extreme reliance on customized software development**
 - Around **90% of Japanese software investment** was directed toward customized development (outsourced or in-house), with only a small fraction devoted to packaged or scalable software products
 - Customization reinforced one-off project logic, making reuse, platform thinking, and product ecosystems economically unattractive

- **Dominance of system integrators and “software factory” structures**

- Roughly **75% of Japanese IT engineers** were employed by system integrators and subcontracting firms rather than product companies
- These environments prioritize delivery schedules, quality compliance, and cost targets—**not experimentation or innovation**
- Multi-tier subcontracting diluted architectural ownership and discouraged radical design choices

- **Thinning of in-house IT capabilities in large firms during the 1990s**

- Many Japanese corporations spun off IT departments or outsourced them to reduce costs
- This weakened internal capacity to:
 - Identify new IT opportunities
 - Challenge system integrators
 - Align software with long-term strategy
- As a result, firms became passive consumers of IT rather than active innovators

- **Limited strategic authority of CIOs**

- A relatively small share of Japanese firms had full-time CIOs, and even when the title existed, it often carried little decision-making power
- This reinforced the perception of IT as a support function rather than a core strategic capability

- **Low contribution of software to productivity growth**

- Empirical evidence shows software contributed **significantly to total factor productivity in the U.S.**, but had **near-zero or negative impact in Japan**

during comparable periods

- This reflects misallocation of IT investment toward maintenance and customization rather than innovation
Cole_Robert_The_Japanese_Softwa...

- **Weak startup ecosystem and venture capital support**

- Software startups played only a marginal role in Japan, receiving far less venture funding than in the U.S.
- Without startups as external innovation engines, large firms lacked pressure and models for rapid software-driven change
Cole_Robert_The_Japanese_Softwa...

Path Dependency and Hardware Centricity

(How past manufacturing success locked Japan into the wrong trajectory)

- **Path dependency rooted in Japan's manufacturing success**

- Japanese electronics firms were founded and matured during an era when **hardware excellence was the primary source of competitive advantage**
- Organizational routines, promotion systems, and evaluation metrics became deeply aligned with hardware innovation
Cole_Robert_The_Japanese_Softwa...

- **Increasing returns reinforced hardware dominance**

- Long-term success in hardware created:
 - Heavy sunk investments in hardware skills
 - Organizational confidence in hardware-first strategies
- These increasing returns made it rational—but ultimately harmful—to continue prioritizing hardware over software
Cole_Robert_The_Japanese_Softwa...

- **Software positioned as a secondary “supporting” function**
 - Embedded software was typically introduced **late in product development**, after hardware architecture had already been fixed
 - This structurally limited software engineers’ ability to influence core design decisions
Cole_Robert_The_Japanese_Softwa...
- **Seniority-based promotion systems reinforced bias**
 - Older hardware engineers occupied top management positions, while younger software engineers lacked authority
 - Managers tended to promote people with similar backgrounds, reinforcing hardware-centric leadership over time
Cole_Robert_The_Japanese_Softwa...
- **Self-fulfilling perception of software as non-innovative**
 - Because software teams were brought in late and under time pressure:
 - Their work focused on implementation rather than innovation
 - Software appeared “routine” rather than strategic
 - This reinforced the belief that software was inherently less valuable than hardware
Cole_Robert_The_Japanese_Softwa...
- **Confusion between software quality and software innovation**
 - Japanese firms excelled at producing **high-quality, low-defect software**, especially embedded systems
 - However, emphasis on defect elimination and perfection often:
 - Slowed time-to-market
 - Reduced tolerance for experimentation
 - Suppressed radical innovation
Cole_Robert_The_Japanese_Softwa...

- **Monozukuri ideology institutionalized hardware bias**

- Government policy and corporate culture elevated *monozukuri* (craftsmanship in manufacturing) as a national strategy
- Software was framed as a facilitator of hardware excellence, not as an independent source of customer value
- Even in the 2010s, software received minimal attention in official manufacturing policy documents
Cole_Robert_The_Japanese_Softwa...

- **Lack of disruptive entrants preserved legacy practices**

- Unlike the U.S., where startups regularly displaced incumbents, Japan's electronics sector saw little firm turnover
- New divisions and spin-offs were staffed by insiders, reproducing existing routines rather than breaking from them