Part I

The fundamentals

1 The offshore landscape

Offshoring IT work is an important milestone in the history of global economics. Why has this happened now?

There is no *one* factor that brought about this phenomenon, but rather, six. Six principal forces converged, as depicted in Figure 1.1. The first of these forces is well known: the *globalization* of trade and, more recently, the globalization of trade in services, which is now approaching 2 trillion USD annually. Borders began opening in the 1980s as market-based solutions gained broad acceptance. The collapse of the Soviet bloc spurred this process even more.

Nations that were once hostile to business, or at best indifferent, are now competing with one another to attract foreign investment and spur their software sectors, creating a *business-friendly climate*. Nations are offering tax incentives and are easing government regulations. They are building technology parks to make it easy to set up and run business

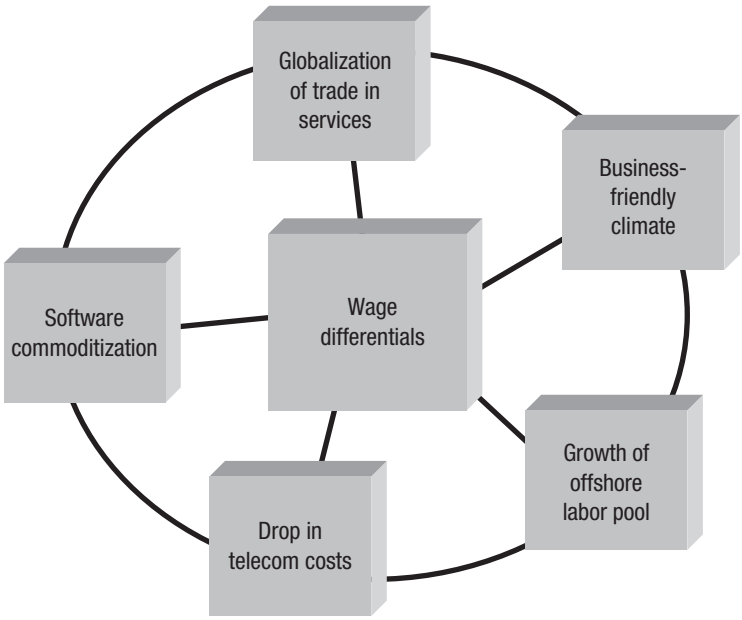


Figure 1.1 The principal economic, business, and technology forces of offshoring.

operations. India and the Philippines are dotted with such technology parks. China has established 15 software parks and 53 technology parks (more on the country "menu" will come in Chapter 4).

Meanwhile, the number of engineers pouring out of universities and technical schools in India, China, and other nations has surged. China, alone, graduates four times as many engineers as the US every year. While the quality of these programs was once inferior to those in industrialized nations, the gap has narrowed. The elite of the offshore labor pool – the talent that is now being directed at higher-end software activities (e.g. research and development (R&D)) – was always there. But, not long ago, this talent would emigrate to the industrialized nations or find other jobs. Today, global technology firms tap these talented engineers and scientists wherever they may be.

In the course of just a decade *communications costs* have decreased to almost zero for nearly unlimited usage. This has brought about a remarkable outcome – that it is almost as easy to work with someone across the ocean as across town (though not equally the same, see Chapters 8 and 9). Between the late 1990s and the early 2000s the benchmark international calling rates have fallen by 80–90%; that is, for those who still use standard rates. Many software workers use voice over IP at zero marginal cost. Equally important for software, the bandwidth has expanded by orders of magnitudes, from almost zero in the 1990s, reaching 4 gigabits per second to India alone in 2004 (with up to 9 terabits per second of system capacity). It was only in 1994 that one of the pioneering project managers offshoring to India had his team copy the weekly software "build" onto tape every Friday just in time for the FedEx pick-up that would fly the tape across the ocean.

Software commoditization is not as well understood by those outside the software industry. It is the standardization of software development practices and tools. For the first time, in software's 50-year history, some software tasks are sufficiently routinized and automated that they have been "commoditized." These tasks are nearly undifferentiated by producer, like a barrel of oil or a bushel of wheat. Once some tasks are commoditized they can be produced by the lowest-cost, most-productive bidder. As one manager commented to us "these are the skills that you can shop for on the Internet."

Finally, and make no mistake about it, the dominant force in offshoring is the *wage differential* between low- and high-wage nations. Hence this force appears in the center of Figure 1.1. The wage differentials lead to lower costs. Some managers will utter other politically acceptable reasons for offshoring that seem less offensive than simply slashing costs, but these are often secondary considerations voiced for appearances. The cost pressures have made offshoring a strategic necessity for some firms (see Chapter 5, Offshore Strategy). Not only are corporate executives stressing cost savings, but American venture capital firms, in an effort to reduce their own capital investments in young firms, have pushed technology startups to perform their R&D offshore from the outset. Until the 1990s, technology firms looked largely at labor pools in high- and middle-wage

nations: the G7 nations,[1] Switzerland, Israel, Brazil, and several others. This has permanently changed.

To be sure, there are other, secondary, forces that helped spur offshoring, such as the emergence of sophisticated IT firms offshore, especially in India; and the advantages, in isolated cases, of working around the clock. Additionally, market access has been a factor for large technology companies. Large global firms need to, or are forced to, invest in operations in important nations. China is the premier example of this. No important technology firm can sell to China today without having some R&D or manufacturing operations in-country. For example, in 2003 China mandated its own cryptographic standard for wireless local area networks (LANs). Foreign firms who wanted to access this market were forced to collaborate in software R&D with local companies.

Historical context and lessons for the future

Offshoring is not new. The principal consumers of offshore software work, the industrialized nations (e.g. USA, UK, and Germany), have already witnessed many manufacturing industries shift offshore as they have matured. These industrial migrations accelerated since World War II: steel, shipbuilding, automobiles, manufacturing, textiles and apparel, consumer electronics, tool making, semiconductors, and others. In the automobile industry, for example, during the period of accelerated decline of the North American industry to Japan, the market share of US firms declined dramatically from 85% in 1974 to 56% in 1991. The common denominator of these historical migrations is that, until recently, offshoring occurred in physical goods; offshoring has taken a new turn in that it is now taking place in *services*.

A useful way to understand the context of these offshoring waves is via Vernon's classic model called the *international product cycle*.[2] The model has three stages. In Stage I, a new product begins with highly skilled entrepreneurial activities, typically in industrialized nations. In Stage II, production begins to shift offshore via investments in low-wage nations. In Stage III, as the product standardizes, it is mass produced with cheap, low-skilled labor. The model seems to describe software offshoring fairly well, helping to explain the recent accelerated pace of software offshoring. Interestingly, while some software segments have now entered the third and "final" stage of the international product cycle, other software segments are still in Stage I or II.

Offshoring is still a small portion of the global software and IT services marketplace – comprising at most 5% of expenditures. A UN report dramatically labeled the new offshoring trend as a "new international division of labor" and emphasized that it is still at its early stages.[3] Where will it go? How far will it go?

Possible future trajectories are plotted in the graph of Figure 1.2 using flat "S curves." One trajectory is becoming visible: a split between design activities (high-level) and

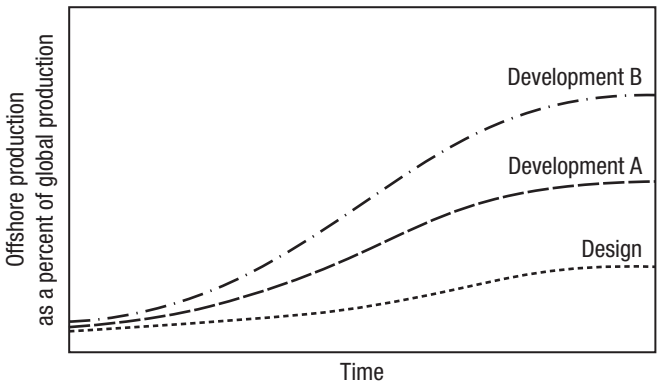


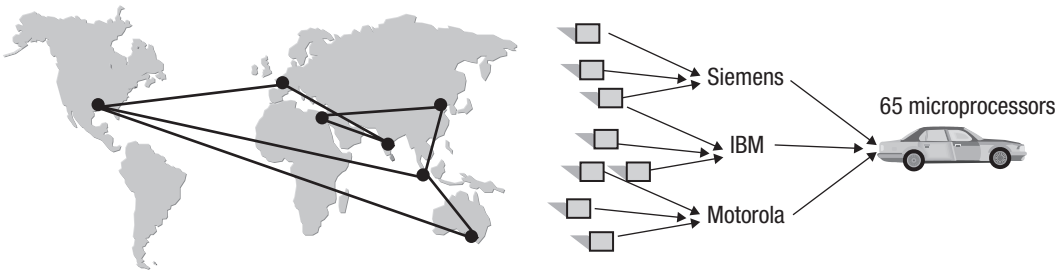
Figure 1.2 Scenarios of offshore migration.

development activities (low-level) that are migrating offshore at different rates. Offshoring will reach some plateau (though we cannot say when). The plateau may be at a lower relative level (as in "Development A") or at a higher relative level (as in "Development B"). While we cannot predict the future, we can draw six lessons from previous offshoring waves that we detail below.

1 To reiterate, in previous offshoring waves, production shifted to lower-wage nations. Some of these waves were gradual increases over time, some had an inflection point where offshoring accelerated, others moved through some kind of "S curve" reaching a new plateau (e.g. automobiles), as in the two upper lines of Figure 1.2. In consumer electronics, production – and much of the design – moved first to Japan and then shifted to the newly industrialized countries (NICs: South Korea, Taiwan, Singapore, and Hong Kong), and then to other Asian countries (China, Thailand, and Malaysia) with even lower wages. This migration pattern is referred to by international economists as the *Flying Geese Formation*, where the lead goose is the US, followed by Japan, and so on. These flying geese are beginning to appear in the software industry, as well. After the US lead, the first geese were the three Is: India, Israel, and Ireland. The second tier of geese appeared when Indian firms began to move some software work to another tier of low-wage nations: Vietnam and China.
2 In previous offshoring waves, it was not just production that moved offshore but the *know-how* about that production, as was the case in the steel industry or, separately, the semiconductor-manufacturing industry. In software offshoring, production know-how transfer is most evident in the quality standards such as Capability Maturity Model (CMM) and International Standards Organization (ISO) (which are introduced later in this chapter). The Indian organizations, and later software firms in other nations, embraced these standards and are now global leaders in their application.

3 Offshoring is a significant industry tremor leading to massive restructuring, namely: acquisitions, consolidation, job displacement, and the emergence of global giants with a broad presence in major markets. Since 2000 this tremor and its after-shocks has been evident in the IT services industry where the distinctions between the major IT services organizations in the US, Europe, and India have begun to blur. American firms in this segment increasingly resemble Indian firms in their offshore offerings, while the large Indian firms are vying for the largest contracts just like the American firms.
4 In previous offshoring waves there was often a corresponding rise in the industry's productivity in the home countries, due to a rise in R&D investments, automation, and production efficiencies. In parallel there were significant changes in the design approaches used in each industry. Charles Simonyi,[4] one of Microsoft's first software architects, argues that offshoring is but a prelude to software automation and mechanization. There is evidence that this is already taking place as software service companies scramble to automate labor-intensive tasks in data centers, software customization, translation, web site hosting, and reuse of code.
5 In some industrial offshoring waves there was a split between higher-level design activities and lower-level production activities, as in the distinction between *design* and *development* of Figure 1.2. Indeed, in the case of software, one of the forces of offshoring is standardization, allowing some factory-like approaches in software production. This is a departure from the practice of many decades in which software was practiced largely as a craft. Standardization is less evident in higher-level (design) tasks, which are more creative tasks, and which are usually the sources of a company's competitive advantage.
6 The political dynamics surrounding previous offshoring waves suggests that protectionist policies, such as import barriers, can help to slow offshore migration for some periods, but do not seem to be effective in the long term. This is an interesting lesson for industrialized nations struggling to deal with the ramifications of offshoring (see Chapter 12, Offshore Politics). The political dimensions have also changed in this offshoring wave. In the case of the US, the political constituencies of business and labor have diverged. Large corporations were vocal when the competitive threats came from Japan in the 1980s. However, in the software offshoring debate of post-2000, US firms continue to dominate the global marketplace. Not only do they not lobby for protection, to the contrary, they lobby against protectionism. The other political constituency is software labor, which in the US is largely non-unionized.

We offer a final observation in our look into the future. Offshoring will likely accelerate the formation of two industry configurations: networks and supply chains (see Figure 1.3). On the one hand, offshoring has created truly global networks of software activities, similar to the well-known network structure of the Internet. A network is set of connected nodes with each node connecting to many other nodes. It is not unusual anymore



(a) Network of software activities

(b) Supply chain of software activities depicting embedded software that is used in a typical GM car.

Figure 1.3 Future structures of global software activities: network and supply chain.

to find a network of collaborating teams, as in the case of an EDS project that had such network collaboration between Mexico, Australia, Egypt, and Brazil.

On the other hand, borrowing from another business area, the software industry is beginning to resemble the auto industry in that there is a "global supply chain" of software producers, where each producer adds value as the software is transformed and then passed from one phase to the next. We see this illustrated in the auto industry itself, in the embedded software that goes into today's cars. A typical GM vehicle has about 65 specially built microprocessors, each with its own embedded software (in fact, together, these 65 microprocessors are now more expensive than the costs of all of the other raw materials that go into the car). GM writes little of this software in-house, with the exception of the microprocessor for the power train. Instead, it contracts with three major suppliers, Siemens, IBM, and Motorola, who in turn, source from a network of American, European, and Asian software centers. In short, a global supply chain of software that goes into your car.

The Offshore Stage Model: progression and diffusion

We now turn to look at companies that are offshoring in order to understand the progression and diffusion of this phenomenon. The Offshore Stage Model, first described in an article by Carmel and Agarwal,[5] helps us to tell this story.

Companies tend to move through four offshoring stages depicted in Figure 1.4. Companies that do not offshore are in Stage 1, "Offshore Bystander," in which they metaphorically watch the others. In fact, as we later discuss, most companies, whether large or small, are still in Stage 1.

Stage 2, "Experimental," is a transition stage in which companies test the offshoring waters for a year or more. For large corporations this stage's expenditures could be as large as 10–20 million USD per year. Experimentation is a wise approach for organizational learning and risk reduction because of the many difficulties in offshoring. Savvy

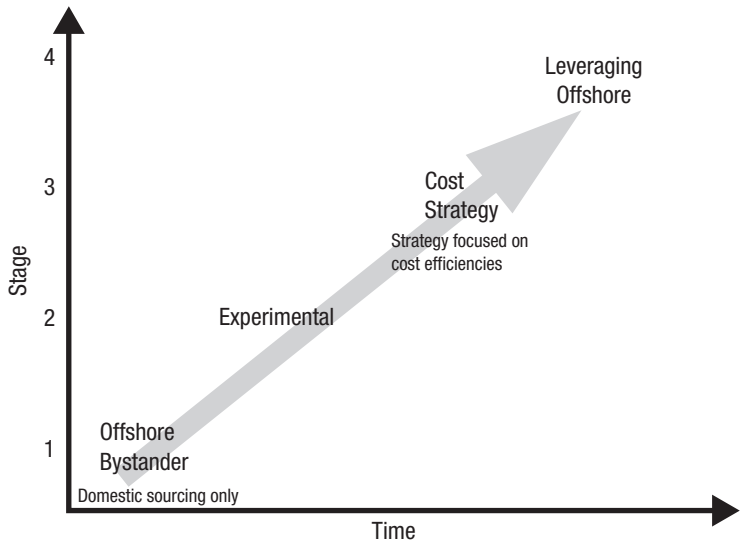


Figure 1.4 Offshore Stage Model.

managers experiment to the point where they see measurable, positive results, and only then do they grow to the next stage. Some call this the "Start-Small" strategy and, according to one study, 63% of companies are using this approach.[6]

In Stage 3, "Cost Strategy," companies begin to experience significant and consistent cost savings in their IT work. By this stage, firms have corrected some early missteps and have expanded their offshore activity as measured by number of projects, staff, or budget. There have been hundreds of firms, if not thousands, large and small, which claim cost savings in their software related activities driven by the low wages in offshore nations. Various studies have tried to determine just how much offshoring saves. The composite of studies indicate that the cost savings ranges from 15% to 40%[7] for companies offshoring at least a year (this is discussed in Chapter 2, Offshore Economics and Offshore Risks).

Experienced companies move to Stage 4, the highest stage, where they truly leverage offshoring. In this stage companies move beyond mere cost savings derived from wage differentials and benefit from other strategic advantages. Here, offshoring is used to drive innovation, speed, flexibility, and new revenues.

The Offshore Stage Model is also useful to measure offshoring *diffusion*. Since it was introduced in 2002, it has been used to estimate the ratio of large companies at each stage of the offshore progression, as shown in Table 1.1. The rough estimates in this table, made by two American research companies, indicate that only 10% of the largest US corporations were active in offshoring in 2003–2004 (i.e. they were in either Stage 3 or 4). Furthermore, about half of the largest American firms do not offshore at all. In spite of the enormous attention to offshoring in the US in the early 2000s, offshoring was still rather limited.

Table 1.1 Offshore stages of US Fortune 1000 firms

Stages	Percent of 1000 largest US firms in this stage (2003–2004)		Percent of all software work which is offshored for a typical firm in this stage (%)
	Meta Group (%)	Forrester (%)	
Stage 1	55	50–60	0
Stage 2	33	25–30	5
Stage 3	8	5–10	10–30
Stage 4	4	<5	40–50

Source: Estimates by Meta Group[8] and Forrester[9]

The stages can also be used to anticipate offshore diffusion for large firms. If, assuming conservatively, only 20 firms a year move out of Stage 1 and into Stage 2, and the annual advance from stage to stage is just 10% of the firms in that category, then by 2010, nearly one-third of US "Fortune 1000" firms will be active offshore users in Stages 3 and 4.

Strategic advantages

IT offshoring has been driven primarily by executives' desire to lower operational cost. This is the Cost Strategy of the Offshore Stage Model. Lowering operational costs does not necessarily translate into a company's strategic advantage, just as saving money on a new office lease is not a strategic advantage, but merely the relentless day-to-day effort of any company to reduce its operating costs.

However, in some industries, IT offshoring is beginning to be viewed as a *strategic necessity.* Some call it "offshore or die." When one company's cost efficiencies allow it to lower prices or expand its competitive options, then other companies must match their competitor's strategy, or fail. Offshoring is becoming part of the larger context of *hyper-competition*: companies are swept into faster and faster cycles of competitive responses and reactions in order to remain financially viable and cost competitive. *Not* offshoring may well become a strategic peril. Such was the case of one of America's largest television manufacturers, Zenith Electronics, which resisted offshoring for decades, while slowly shrinking, before it disappeared completely.

While cost reduction is the primary strategic focus of most companies that are offshoring, it is not the only strategic advantage to offshoring. The fourth and final stage in the Offshore Stage Model is labeled "Leveraging Offshore." As we saw in the estimates of Table 1.1, there are relatively few companies that have reached this stage. Those that have progressed to this stage have moved beyond mere cost reduction and benefit from innovation, speed, flexibility, and new revenues. We discuss these benefits in greater detail in Chapter 5, Offshore Strategy. Here we introduce the two most important of these additional strategic goals: attaining speed and accessing talented labor.

The first strategic lever is the increase in *speed, agility, and flexibility*. This means that companies that offshore can rapidly ramp-up (by reducing the time to get the project started) and reduce project duration (time-to-completion). The abundant supply of labor offshore gives companies greater agility: to assign a large number of engineers to a problem; to forge ahead in several directions instead of just one; to ramp-up (scale-up) and respond to a business need within days instead of months.

Companies that develop software products benefit from the second strategic advantage: *accessing talent.* For these companies, their success stems from innovation and their innovation capabilities come from their talent – their most brilliant and creative engineers. Firms that expand abroad to tap this talent are called "knowledge seekers" [10] and tend to behave somewhat differently than those seeking mainly lower wage rates. In previous decades technology companies would tap foreign talent by going to other high-wage, industrialized nations. In the 1990s, they began turning to Israel, India, and later to China. For example, by 2003, 77 global software product firms established direct R&D subsidiaries in India.[11] Many others perform contract R&D on an out-tasking basis in India.

Follow-the-sun

Stories about offshoring often mention *follow-the-sun*, also known as *round-the-clock.* Along with low costs, follow-the-sun is another allure of offshoring. It is often mentioned by those who seek to make offshoring sound unique. Follow-the-sun, as the name hints at, exploits time zone differences to speed up project work. For example, a team in America can hand off its work at the end of its day to team members in India or China, who can then continue the work while the US team members sleep.

This has undeniable appeal. If software work can be coordinated properly, then project duration can be reduced by a factor of two. Moreover, if three teams are correctly positioned across time zones, then a theoretical threefold duration reduction is possible. This is much like a factory running three shifts, 24 hours per day, producing three times the volume. Using follow-the-sun development, a company may be able to save months from the development cycle and release a product earlier, thus giving it a competitive advantage. This is an enormous potential benefit of offshoring.

However, coordination in follow-the-sun must be flawless in order to reduce project duration. One miscommunication can delay the entire day's worth of work. In practice, few globally dispersed software efforts have been able to fully capitalize on the theoretical advantages of follow-the-sun. Daily follow-the-sun coordination is simply too difficult for software teams. An IBM team, described in Carmel's 1999 book,[12] was set up to capitalize on follow-the-sun. However, fairly quickly, the global team discovered that daily handoffs were too difficult to coordinate.

Nevertheless, follow-the-sun can be effective for some activities and for certain phases in software work. Startups in Silicon Valley have been excited about rapid prototyping